Maestría en:

INTELIGENCIA ARTIFICIAL APLICADA

Trabajo previo a la obtención de título de Magister en Inteligencia Artificial Aplicada

AUTORES:

Burneo Aguilera Richard Patricio

Cevallos Apolo Richard David

Changalombo Trávez Adrián Alexander

Cortez Osejo Christian Mateo

Salinas Herrera David Fernando

TUTORES:

Alejandro Cortes

Karla Mora

TEMA:

Sistema de Trascripción de Voz a Texto Enfocado en el Tono Emocional.

Quito - Ecuador

marzo – 2026

REINVENTEMOS EL FUTURO

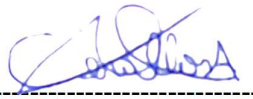
Certificación de autoría

Nosotros, **Richard Patricio Burneo Aguilera, Richard David Cevallos Apolo, Adrián Alexander Changalombo Trávez, Christian Mateo Cortez Osejo, David Fernando Salinas Herrera,** declaramos bajo juramento que el trabajo aquí descrito es de nuestra autoría; que no ha sido presentado anteriormente para ningún grado o calificación profesional y que se ha consultado la bibliografía detallada.

Cedemos nuestros derechos de propiedad intelectual a la Universidad Internacional del Ecuador (UIDE), para que sea publicado y divulgado en internet, según lo establecido en la Ley de Propiedad Intelectual, su reglamento y demás disposiciones legales.

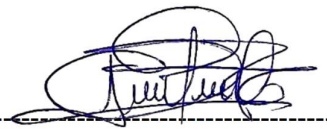
Firma

Richard Patricio Burneo Aguilera



Firma

Richard David Cevallos Apolo



Firma

Adrián Alexander Changalombo Trávez

Firma

Christian Mateo Cortez Osejo

Firma

David Fernando Salinas Herrera

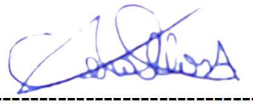
Autorización de Derechos de Propiedad Intelectual

Nosotros, **Richard Patricio Burneo Aguilera, Richard David Cevallos Apolo, Adrián Alexander Changalombo Trávez, Christian Mateo Cortez Osejo, David Fernando Salinas Herrera,** en calidad de autores del trabajo de investigación titulado ***Sistema de Trascripción de Voz a Texto Enfocado en el Tono Emocional***, autorizamos a la Universidad Internacional del Ecuador (UIDE) para hacer uso de todos los contenidos que nos pertenecen o de parte de los que contiene esta obra, con fines estrictamente académicos o de investigación. Los derechos que como autores nos corresponden, lo establecido en los artículos 5, 6, 8, 19 y demás pertinentes de la Ley de Propiedad Intelectual y su Reglamento en Ecuador.

D. M. Quito, (mes año)

Firma

Richard Patricio Burneo Aguilera

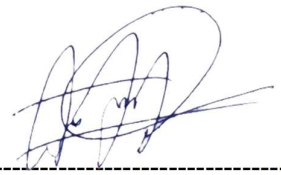


Firma

Richard David Cevallos Apolo

Firma

Adrián Alexander Changalombo Trávez



Firma

Christian Mateo Cortez Osejo

Firma

David Fernando Salinas

ACUERDO DE CONFIDENCIALIDAD

La Biblioteca de la Universidad Internacional del Ecuador se compromete a:

1. No divulgar, utilizar ni revelar a otros la información confidencial obtenida en el presente trabajo, ya sea intencionalmente o por falta de cuidado en su manejo, en forma personal o bien a través de sus empleados.

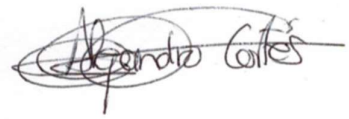
2. Manejar la información confidencial de la misma manera en que se maneja la información propia de carácter confidencial, la cual bajo ninguna circunstancia podrá estar por debajo de los estándares aceptables de debida diligencia y prudencia.

Paulina Vizcaíno
Directora Académica

Gabriela Fernández
Gestora Cultural

Aprobación de dirección y coordinación del programa

Nosotros, **Alejandro Cortés director EIG y Karla Mora Coordinadora UIDE,** declaramos que: **Richard Patricio Burneo Aguilera, Richard David Cevallos Apolo, Adrián Alexander Changalombo Trávez, Christian Mateo Cortez Osejo, David Fernando Salinas Herrera,** son los autores exclusivos de la presente investigación y que ésta es original, auténtica y personal de ellos.



---------- ----------

Alejandro Cortés
Director de la
Maestría en Inteligencia Artificial Aplicada

Karla Mora
Coordinadora de la
Maestría en Inteligencia Artificial Aplicada

DEDICATORIA

Dedicamos este trabajo a Dios, por acompañarnos en cada etapa de este camino, darnos la fortaleza para seguir adelante y permitirnos llegar a la culminación de esta meta académica.

A nuestras familias, por su paciencia, apoyo constante y comprensión durante este proceso. Su confianza y aliento fueron fundamentales para no rendirnos y continuar avanzando, incluso en los momentos más exigentes.

AGRADECIMIENTO

Agradecemos a la universidad y a los docentes del programa de maestría por compartir sus conocimientos y guiarnos con compromiso y profesionalismo a lo largo de nuestra formación académica.

Expresamos un agradecimiento especial a nuestro/a tutor/a, por su orientación, disposición y acompañamiento durante el desarrollo de este trabajo, así como por sus valiosos aportes, que contribuyeron significativamente a su culminación.

Finalmente, agradecemos a nuestras familias y a todas las personas que estuvieron presentes de una u otra manera, brindándonos apoyo, ánimo y comprensión durante este proceso. Su respaldo fue clave para alcanzar este logro.

RESUMEN

En este proyecto se desarrolló un sistema de traducción de voz a texto enfocado en el tono emocional. Para lograr esto, se realizó una investigación de las técnicas y herramientas que pueden ser utilizadas para desarrollar el sistema donde se encontraron limitaciones en modelos actuales, sobre todo para la detección de emociones de texto en español, por lo que se decidió realizar un ajuste fino de un modelo existente.

La arquitectura del sistema consta de 2 capas principales: el reconocimiento automático de voz (ASR) y la detección de emociones en el texto. Para el ASR se utilizó el modelo WHISPER mediante el api de OpenAI y el modelo VOSK para el mismo propósito, en donde se obtuvieron resultados positivos en general, ya que estos modelos están optimizados y funcionan de manera estable. Para la detección de emociones se intentaron varias técnicas. La primera fue utilizar modelos existentes de HuggingFace en donde el modelo base fue emotion-english-distilroberta-base con siete emociones principales: neutral, miedo, ira, tristeza, alegría, sorpresa, asco. Este modelo funciona de manera correcta en audios en inglés, pero en español no se obtuvieron resultados favorables. Por este motivo se realizó un ajuste fino sobre este modelo.

Palabras Clave: *ASR, Whisper, transcripción, OpenAI, Hugging Face, Fine Tuning.*

ABSTRACT

This project developed a speech-to-text translation system focused on emotional tone. To achieve this, research was conducted on the techniques and tools that could be used to develop the system. Limitations were found in current models, especially for emotion detection in Spanish text. Therefore, it was decided to fine-tune an existing model.

The system architecture consists of two main layers: Automatic Speech Recognition (ASR) and emotion detection in text. For ASR, the WHISPER model was used via the OpenAI API, yielding generally positive results, as this model is optimized and operates stably. Several techniques were attempted for emotion detection. The first was to use existing face-hugging models, with the base model being emotion-english-distilroberta-base, containing seven main emotions: neutral, fear, anger, sadness, joy, surprise, and disgust. This model works correctly with English audio, but did not produce favorable results with Spanish. For this reason, a fine-tuning of this model was performed.

Keywords: *ASR, Whisper, transcription, OpenAI, Hugging Face, Fine Tuning.*

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

CAPITULO 1

1. INTRODUCCIÓN

1.1. Introducción al proyecto

En el Ecuador, el Instituto Nacional de Estadística y Censos (INEC) estimo en el año 2022 que un 48,6% de los negocios registrados se encuentran en el sector de servicios, y un 36,4% en el sector de comercio, con una notable porción de estos con áreas dedicadas a la atención al cliente y Call Centers para la promoción, notificación y comunicación con cliente y contactos requeridos para las empresas, esto conlleva a que sea del interés de las empresas el monitoreo de la calidad de servicio prestada por la empresa y proteger los interese de esta y sus empleados.

1.2. Análisis de la problemática

En la actualidad existe una problemática real en el área del contact center y este es los análisis masivos de audio considerando las configuraciones que suelen usar para la gestión como lo puede ser el uso de un CRM para controlar estados y datos de clientes como lo puede ser HubSpot y para el almacenamiento de grabaciones una opción es Twilio, sin embargo, a pesar de que estas herramientas son muy modernas y facilitan la gestión aún quedaría el control de calidad como una área de suma importancia que presenta varios desafíos. Los agentes de contact centers frecuentemente se encuentran con el malestar de los clientes que tienen problemas con productos o servicios prestados, lo cual puede generar problemas a la hora de cerrar una venta.

También existe la posibilidad de que clientes sin atención apropiada o resoluciones a tiempo decida llevar su negocio a la competencia. Negligencias por parte de los operarios del call center que cometan descuidos como no mencionar que la llamada está siendo grabada, una negativa al tratamiento o usos de sus datos, conflictos e incorrecta comunicación por parte del agente o el cliente, errores de tipeo de datos, descuidos al comunicar características términos y condiciones de productos o servicios , demora en la atención de requerimientos y consultas, etc. puede generar pérdidas económicas para la empresa o una posible retención de clientela en el futuro, este problema por lo general no

puede ser atendido en el día a día debido a la cantidad de datos que habría que revisar de manera manual, por lo cual se deberían identificar los audios en los que se presenten conflictos o problemas de comunicación lo cual si bien se puede buscar en la bases de datos si es un cliente frecuente o que se ha comunicado por otros requerimientos podría complicar el filtrado por emoción o entender cuál fue el problema que este presento.

Los informes de incidentes y la investigación de estos requieren una transcripción del audio, el motivo de inconformidad, identificación del agente y si tiene si se presentan reincidencias de incidentes pasados o se trata de un caso aislado. Este tipo de investigaciones pueden llegar a tomar varias horas al día lo cual complica la gestión y la calidad del servicio prestado.

1.3. Justificación

Con el análisis de emociones se puede obtener información relevante en distintos escenarios. En la atención al cliente, tener definido un proceso para calificar la experiencia del usuario puede ser útil para las empresas, ya que de esta podrían medir si en general los clientes están satisfechos con los productos o servicios brindados. Asimismo, con estas herramientas no solo podemos obtener resultados generales como comentarios positivos, neutros o negativos, sino que podemos determinar en las conversaciones emociones como alegría, ira, miedo, o tristeza.

Ante esta situación, este proyecto de investigación presenta una propuesta de implementación un sistema de transcripción y reconocimiento de emociones basado en Inteligencia Artificial, en el cual utilizaremos librerías de Python y modelo multilingües que permitirán el procesamiento de audio para reconocer la vocalización de palabras y transcribirlas para el usuario, mientras que a través de modelos de detección de emociones a través del análisis de prosa, intensidad del volumen vocal, la prosa y características lingüísticas del interlocutor para etiquetar secciones de la transcripción con las emociones detectadas.

Estas transcripciones se almacenarán para su análisis histórico para el estudio de calidad de servicio y evaluación de desempeño y profesionalismo.

1.4. Relevancia del análisis emocional en interacciones orales

En escenarios donde las emociones juegan un papel crucial, como lo es en aquellos ámbitos que como parte core de negocio requieren atención al cliente, educación virtual, salud o procesos legales, la interpretación del contenido semántico no es suficiente. Es ahí donde la capacidad de un sistema para identificar emociones como frustración, ansiedad, enojo o entusiasmo aporta contexto valioso para evaluar la calidad de la interacción.

En tal contexto, el sistema prototipo desarrollado permite analizar no solo lo que se dijo, sino cómo se dijo, proporcionando una transcripción enriquecida con emociones que puede mejorar significativamente la manera de evaluar la experiencia del interlocutor en una interacción determinada ya que diversos estudios han confirmado que la importancia del tomo emocional juega un papel crucial cuando se trata de establecer sistemas predictores de satisfacción y bienestar en la atención a requerimientos de manera verbal (Acharya et al., 2021; Cowen et al., 2019).

1.4.1. Automatización de la transcripción emocional

El sistema implementado incorpora dos motores de reconocimiento de voz, siendo estos Whisper que es un modelo neuronal de OpenAI altamente preciso e ideal para servidores con GPU y Vosk que es un modelo ligero, versátil y capaz de ser ejecutado offline. Esta dualidad permite mantener funcionalidades tanto en entornos de altas prestaciones donde no se escatima el poder de cómputo, así como en escenarios con capacidades limitadas de hardware (Radford et al., 2022).

Además, la integración de la librería Resenblyzer permite detectar, segmentar e identificar distintos interlocutores para poderlos segmentar en cada tramo del audio, lo que mejora significativamente el análisis emocional por hablante. A partir de estas transcripciones, se aplican modelos como DistilRoBERTa y XLM’RoBERTa para inferir emociones a nivel de oración y segmento (Radford et al, 2022).

1.4.2. Aplicaciones prácticas y adaptabilidad regional

La arquitectura modular propuesta e implementada en FastAPI facilita la integración del porotito vía API REST con otras plataformas como lo son interfaces gráficas y

microservicios que hagan uso del mismo. Dada la naturaleza FastAPI, permite aprovechar herramientas clave como lo el PyTorch para el procesamiento con modelos pre entrenados y fine-tuning, mientras que la presentación visual de resultados emocionales se lo realiza a través de dashboard minimalistas que no requieren instalación de marcos de trabajo adicionales.

Cabe mencionar que una de las cualidades principales del prototipo es su capacidad de adaptación a entornos hispanohablantes, un área poco cubierta por los servicios comerciales más populares del mercado actual. Esto, claramente, gracias a la incorporación de modelos como pysentimiento y herramientas de traducción automática como Helsinki-NLP (Barbieri et al., 2022).

1.4.3. Impacto académico, social y tecnológico

El proyecto, además de su funcionalidad técnica, representa una herramienta de investigación y aprendizaje continuo, permitiendo que estudiantes y profesionales profundicen en temas de alto impacto como lo son las redes neuronales, procesamiento de voz, y análisis de emociones. Esto, y su enfoque abierto a herramientas open source como Hugging Face, OpenAI Whisper, Vosk y Resemblyzer, lo dotan de total transparencia y posibilidad de adaptación local.

Como consecuencia, desde una perspectiva social, el proyecto puede servir en iniciativas de salud mental, seguimiento emocional en educación remota, y estudios sobre comunicación en el ámbito ecuatoriano, promoviendo así el desarrollo de IA ética, de utilidad y accesible para el entorno latinoamericano.

1.5. Componentes del Sistema de Transcripción y Detección de Emociones

1.5.1. Investigación

1.5.1.1. Análisis del contexto nacional. En el Ecuador, los servicios de atención al cliente y call centers se rigen por regulaciones que incluyen leyes de protección de datos, regulaciones de telecomunicaciones y estándares internacionales utilizados con frecuencia por la industria local, estos deberán tomarse en cuenta para el uso de la herramienta en cumplimiento de las leyes y regulaciones nacionales del Ecuador.

Ley Orgánica de Protección de Datos Personales: Se exige el informar a los usuarios respecto a que se realizara la grabación y el objetivo para el cual se registrara el audio de la llamada y que este de su consentimiento. La voz se considera un dato biométrico por lo que la transcripción debe seguir protocolos de seguridad que impidan la filtración o vinculación indebida de la identidad del cliente sin medidas de protección. Así mismo, las empresas están en la obligación de entregar las transcripciones al cliente si este lo solicitase, en un formato que sea legible y estructurado.

La Agencia de Regulación y Control de las Telecomunicaciones ARCOTEL y Entidades regulatorias y de control: Según el sector en el que se haga uso de esta herramienta, deben seguirse normas específicas sobre la retención de transcripciones e identificación. En empresas de telecomunicaciones, se establecen normas sobre la identificación del remitente para la transparencia, utilizando etiquetas identificativas como Finanzas, Turismo, o Ventas. Para el sector financiero, los estándares de recolección de información indica que las instituciones financieras deberán conservar los registros de audio por un mínimo de 6 meses, si existiera un reclamo en curso entonces la información como audios y transcripciones deberán guardarse hasta que se agotaran las instancias legales. Las leyes de Defensa del Consumidor indica prohibiciones al hostigamiento a través de llamadas en estrictos horarios de 08:00 a 20:00 y en días de lunes a viernes de la semana, por lo que deberán incluirse etiquetas de tiempo para detectar infracciones en tiempo real.

Normas Internacionales ISO: promovidas por el Servicio Ecuatoriano de Normalización (INEN), estos estándares son adoptados por empresas de Outsourcing en Ecuador para certificarse, entre las más notables tenemos:

- ISO 18295-1: Esta normativa define los requisitos para el centro de contacto como infraestructura, gestión de datos operativos, KPIs, etc.
- ISO 18295-2: La normativa define los requisitos para la organización que contratara el call center.

1.5.2. Componentes para el Desarrollo

El componente principal del proyecto será un API de transcripción automática,

desarrollada con el framework FastAPI, ejecutada en un entorno ASGI con Uvicorn, e integrada con el modelo Whisper de OpenAI basándose en redes neuronales profundas para el reconocimiento multilingüe de voz. Este servicio permitirá recibir archivos de audio en formato wav o mp3 a través de peticiones HTTP, para que, tras su procesamiento en el servidor, genere la transcripción correspondiente y retorne como respuesta el texto resultante (OpenAI, 2023; Tiangolo, 2022).

La arquitectura planteada busca ser modular, escalable y mantenible, de tal manera que a medida que avanza el proyecto se pueda integrar fácilmente nuevos módulos, tales como el de la detección emocional de voz, utilizado modelos pre entrenados en bases de datos populares como RAVDESS, CREMA-D o Emo-DB, y que permitirá identificar emociones básicas (alegría, enojo, tristeza, miedo o neutralidad) a partir de características acústicas como el tono de la voz, la energía en la pronunciación y ritmo de la vocalización (Zhang et al., 2021). Los resultados se integrarán en la transcripción mediate etiquetas que identificarán los segmentos del diálogo donde se detectaron dichas emociones.

1.5.3. Definición del proyecto

La propuesta presentada a continuación consiste en el desarrollo de una herramienta de transcripción de voz a texto que reconocerá patrones de habla para determinar el estado emocional del interlocutor aplicado para su uso en llamadas de servicio al cliente de Call Centers. Esta herramienta permitirá, además de producir registros textuales de audios de llamadas de call centers utilizando modelos de Voz a Texto (STT), generar etiquetas en la transcripción que indicaran las emociones detectadas en las porciones del audio procesado a través del análisis de características acústicas como la entonación, ritmo, intensidad y duración para la identificación de las emociones presentes en el audio que es parte integral de la comunicación humana, de esta manera se podrá entender de mejor maneras de las interacciones entre los interlocutores y el factor emocional en la conversación, lo cual podría recontextualizar interacciones que de otra manera no serían evidentes en transcripciones textuales típicas. Se utilizarán librerías

Python de análisis acústicos y procesadores multilenguaje que permitirán en análisis de pistas de audio y la generación de transcripciones.

1.6. Alcance

El presente proyecto tiene por alcance el desarrollo de una herramienta de transcripción con detección de patrones de voz para call centers en español, con foco en post-call análisis de archivos de audio obtenidos de llamadas previamente atendidas por agentes de soporte y ventas.

Cabe mencionar que, en esta primera fase del proyecto, no se integrará el enrutamiento en tiempo real con centrales telefónicas, sino que el sistema procesará grabaciones de ejemplo o simuladas.

- La propuesta tendrá como foco el construir un prototipo funcional que permita automatizar la transcripción de llamadas mediante técnicas de reconocimiento automático del habla (ASR) y su posterior procesamiento para el análisis de emociones en la voz del interlocutor.
- Enriquecimiento de las transcripciones generadas con etiquetas emocionales.
- Validación del prototipo mediante pruebas de concepto con grabaciones reales o simuladas.

1.7. Objetivos

1.7.1. Objetivo general

Desarrollar una herramienta de transcripción automática de voz a texto con detección de patrones emocionales aplicables call centers, que permita analizar grabaciones post-llamada y generar transcripciones enriquecidas con tags emocionales, con el fin de mejorar el monitoreo de calidad de atención al cliente.

1.7.2. Objetivo especifico

- Analizar las técnicas actuales de reconocimiento automático de voz (ASR).
- Integrar la analítica de detección emocional con el sistema de traducción automática.

- Desarrollar un prototipo con una aplicación web para mostrar la funcionalidad del reconocimiento de emociones en voz.

CAPITULO 2:

2. REVISIÓN DE LITERATURA

2.1. Estado del arte

Históricamente, la transcripción de grabaciones de voz se dividía en diferentes pasos definidos (los modelos acústicos, modelos de lenguaje, modelos de léxico). Con la implementación y mejoras de modelos End-2-End muchas de las veces estos procesos se realizan en una sola red neuronal. En los últimos años la implementación de transcripción y análisis acústicos han avanzado de manera acelerada con la popularización de los modelos basados en modelos Deep Learning y las arquitecturas Transformer, estos avances también han crecido con la colaboración continua de comunidades Open Source y a APIs comerciales que han permitido la construcción de varios pipelines multimodales.

2.2. Marco teórico

- **Introducción a los Sistemas de Procesamiento de Audio**. Un sistema con el objetivo de traducir voz y capturar el tono emocional mediante una transcripción requiere, en general, la integración de varios componentes especializados en varios aspectos como la lingüística, la transcripción de voz a texto, y la capacidad de correlacionar los matices afectivos. La implementación de estas herramientas de transcripción y análisis emocional se ha convertido en una parte critica en la operación de múltiples industrias y áreas.
- **Reconocimiento automático del habla (ASR).** El avance significativo más recientes del ASR es Whisper de OpenAI, un modelo multilingüe basado en Transformer entrenado con grandes volúmenes de audio, lo que se demuestra en su robustez en la distinción de los matices procedentes de acentos, el ruido, y la variabilidad del habla, así como su capacidad de realizar transcripciones en múltiples idiomas (OpenAI, 2022).

En los últimos años han surgido extensiones basadas en Whisper que están orientadas a resolver limitaciones en la longitud de los audios. Un ejemplo de esto es la versión WhisperX, que ha sido optimiza para la transcripción en grabaciones extensas

mediante segmentación con Detección de Actividad de Voz (VAD) y alineación forzada a nivel de las palabras, lo que permitiría obtener timestamps con una mayor precisión y reduciría los errores acumulativos que son típicos en los audios continuos y extensos. Por otra parte, una alternativa relevante es la herramienta Vosk, el cual está orientado al reconocimiento de voz en modo offline y pensado para ser ejecutado también en dispositivos que disponen de bajos recursos.

De esta manera su principal bondad es su notable viabilidad para el despliegue local con modelos que son relativamente pequeños, lo cual lo hace una herramienta con una precisión si bien un tanto inferior a los modelos previamente mencionados, puede realizar tareas de transcripción de manera rápida y sin requerir muchos recursos, lo cual permite su potabilidad a un mayor número de usuarios con equipos con capacidad limitada (Alpha Cephei, 2020). Además, han surgido modelos multilingües a gran escala, como NLLB-200, diseñado para cubrir idiomas con pocos recursos, y modelos multimodales, como SeamlessM4T, que integran la traducción de texto y voz en una arquitectura unificada (Barrault et al., 2023; Costa-Jussa et al., 2022).

Aunque varios de estos modelos tienen mayores exigencias computacionales, representarían una clara tendencia hacia los sistemas de traducción multilingües completos. A los efectos de esta tesis, un enfoque modular (ASR + análisis emocional + NMT) permite evaluar de forma independiente la influencia del tono emocional en la traducción, un aspecto que los modelos unificados aún no controlan explícitamente.

2.2.1. Enfoques arquitectónicos y técnicas de soporte

2.2.1.1. Enfoques arquitectónicos. En la actualidad, los modelos tradicionales han sido reemplazados por modelos End-to-End (E2E), los cuales son capaces de mapear los audios directamente a textos haciendo el uso de una sola red neuronal, a continuación, nombraremos algunas técnicas.

2.2.1.1.1. Clasificación Temporal Conexionista (CTC). Es una técnica clásica aun en vigencia en la actualidad, la cual permite que la red neuronal realiza predicciones de características o de palabras sin la necesidad de que el audio este en alineación con el

texto durante el entrenamiento, un ejemplo de estas son los modelos ligeros como Vosk o las primeras versiones de Baidu DeepSpeech.

2.2.1.1.2. Modelo Codificador-Decodificador Basado en Atención (AED). Son arquitecturas de las redes neuronales que procesan secuencias de entrada; como serian audios de voz o textos; y generara secuencias de salida, utilizando mecanismos de atención que enfocaran dinámicamente en partes relevantes de la entrada al generar cada sección de la salida, lo cual mejoraría de manera enorme tareas como la traducción automática, resúmenes y reconocimiento de voz. Un ejemplo bastante popular es OpenAI Whisper.

2.2.1.1.3. Transductor o RNN-Transducer (RNN-T). Esta arquitectura de redes neuronales se denomina de secuencia a secuencia, diseñado principalmente para el reconocimiento automático de voz de extremo a extremo en tiempo real, el cual es capaz de realizar el modelado en conjunto de dependencias acústicas y del lenguaje.

2.2.1.2. Técnicas de procesamiento y refinamiento. La transcripción de un audio de voz dependerá de una serie de procesos que ocurrirán antes, durante y después de que la red neuronal “escuche” y procese el audio, a resaltar:

2.2.1.2.1. Diarización de locutores. Es una técnica que separaría el audio en segmentos según el hablante utilizando embeddings o huellas digitales de voz, para luego agruparlo mediante el uso de algoritmos de Clustering.

2.2.1.2.2. Detección de actividad de voz. La detección de actividad de voz (VAD) es una técnica que identifica la presencia de voz humana en una señal de audio, este analiza la entrada de audio en tiempo real y distingue entre los segmentos que contienen voz y de aquellos que contienen silencio, ruido de fondo u otros sonidos.

2.2.1.2.3. Post-procesamiento con LLMs (Error Correction). En la actualidad es una práctica estándar el procesar el texto ‘crudo’ con un modelo de lenguaje para la corrección de errores gramaticales o recontextualizar porciones del texto, por ejemplo, con Llama 3 o GPT-4o.

2.2.2. Modelos de reconocimiento de emociones

Mientras que los modelos de transcripción buscan los patrones lingüísticos, el

reconocimiento de emociones está enfocado en la búsqueda de los parámetros acústicos. Se indica a continuación algunos de los procesos y avances de los modelos de Reconocimiento de Emociones.

Extracción de características. Los modelos de reconocimientos de emociones deben extraer datos en concreto de las ondas de audio de las grabaciones:

- Prosodia: El tono (o frecuencia fundamental), el ritmo y la energía o volumen permiten determinar emociones atreves de patrones, por ejemplo, las voces agudas y rápidas suelen indicar sorpresa o alegría; mientras que una voz lenta y grave puede indicar tristeza.
- Espectrales o coeficientes cepstrales de frecuencia Mel (MFCC): estos son características de audio que representan los aspectos del habla de forma compacta, aplicando la escala de Mel, el cual es un modelado que simula el oído humano al percibir el sonido, el cual es más sensible a las frecuencias bajas y comprime las frecuencias altas.
- Embeddings emocionales: Uso de redes neuronales para extraer representaciones densas que capturen emociones, por ejemplo, los modelos basados en Wav2Vec 2.0 o HuBERT).
- Arquitecturas multitarea: los modelos ASR + SER en Paralelo realizan reconocimiento de voz y detección de emociones simultáneamente, compartiendo capas iniciales para extraer características comunes.
- ASR condicionado por emoción: La salida del ASR se ajusta según la emoción detectada, mejorando la precisión en contextos emocionales

2.2.2.1. Datasets y Benchmarks. De acuerdo a lo analizado, entre los principales se menciona los siguientes:

- RAVDESS: Base de datos de audio con etiquetas emocionales (felicidad, tristeza, ira, neutral).
- IEMOCAP: Dataset de interacciones emocionales en inglés, ampliamente usado

para evaluar modelos de SER.

- CREMA-D: Dataset multilingüe con actores expresando emociones en diferentes idiomas.

2.2.2.2. Detención de emociones en texto y audio. El análisis de sentimientos se puede ver desde dos perspectivas complementarias, el sentimiento basado en texto transcrito y el sentimiento basado en señales de audio al tener en cuenta los elementos prosódicos como la entonación, la energía y el ritmo. Una tendencia reciente favorece enfoques basados en transformadores y/o representaciones profundas previamente entrenadas para mejorar la generalización.

En cuanto a la emoción a partir del texto, se destaca la biblioteca PySentimiento para el idioma español, esta es una biblioteca que integra a los modelos basados en Transformers entrenados/ajustados para las tareas de percepción de sentimientos y emociones en lenguaje social, con categorías que tienden a alinearse con las emociones básicas (por ejemplo, Ekman) que son más neutras. Su relevancia para el presente proyecto radica en el hecho de que, en el análisis emocional, se está realizando en el idioma de salida del Reconocimiento automático del habla o ASR en el idioma español lo que facilitaría la interpretación de los matices culturales y las expresiones coloquiales (Hartmann, 2022).

En inglés existen modelos especializados como serían las variantes de RoBERTa o DistilRoBERTa, los cuales están ajustados para la clasificación de emociones, disponibles en repositorios de modelos y que son fácilmente integrables a través de pipelines (Hartmann, 2022). Esto permite el uso de estrategias multilingües. Por ejemplo, al transcribir y analizar las emociones directamente en español o al traducir y luego evaluar la emoción en el texto de destino, dependiendo del propósito y la experimentación.

La detección de emociones desde una perspectiva acústica se ha beneficiado del uso de representaciones profundas. Los enfoques tradicionales como openSMILE previamente extraían manualmente las características prosódicas (tono, energía, ritmo), pero en los métodos actuales se emplean incrustaciones de audio preentrenadas o se hace

el uso de redes profundas aplicadas a los espectrogramas (Eyben et al., 2010). Gracias a estudios recientes se ha demostrado que la combinación de la información acústica con el texto transcrito mejoría significativamente la detección de las emociones en especial en casos que presenten modismos o la ofuscación del mensaje como por ejemplo en el caso del sarcasmo o la ambigüedad semántica (Stappen et al., 2021).

2.2.3. Herramientas de código abierto

Las soluciones de código abierto en la actualidad permiten la implementación de soluciones robustas de alta calidad para cada fase del proceso, en donde la ventaja principal de estos modelos y herramientas es el control del pipeline y la posibilidad de su ejecución en entornos controlados que permitirían mantener bajo control los criterios de privacidad y costos.

2.2.3.1. Evolución y arquitecturas clave. Conforme lo analizado, a continuación, se detallan los principales modelos:

2.2.3.1.1. Modelos clásicos. En tal sentido, es preciso describir los principales modelos clásicos:

- GMM-HMM (Gaussian Mixture Models - Hidden Markov Models): estos modelos dominaron el ASR hasta alrededor de la década de 2010. Eran efectivos en entornos controlados, pero estos conllevaban limitaciones en el ruido y variabilidad de la voz.
- DNN-HMM (Deep Neural Networks - Hidden Markov Models): Introdujeron las redes neuronales para mejorar la robustez, sin embargo, estos aún dependían de alineaciones forzadas y de datos etiquetados.

2.2.3.1.2. Modelos basados en aprendizaje profundo. Es así que, a continuación, se describe los principales modelos basados en aprendizaje continuo:

- RNN y LSTM: Permitieron modelar dependencias temporales en la señal de voz, mejorando la precisión en secuencias largas.
- CNN (Convolutional Neural Networks): Son efectivas para la extracción de características locales en el espectrograma de audios.

***2.2.3.1.3. Modelos de aprendizaje autónomo y autorregresivos*.** En el presente apartado se describen los principales modelos de aprendizaje autónomos:

- Wav2Vec 2.0: Modelo de aprendizaje autónomo que aprende representaciones de audio sin etiquetas, logrando alto rendimiento en ASR con pocos datos etiquetados. Su arquitectura combina una red convolucional para extraer características y un transformer para modelar contextos largos (Facebook AI, 2020).
- HuBERT: Extiende Wav2Vec 2.0 usando objetivos de aprendizaje basados en clustering, mejorando la representación de unidades fonéticas y emocionales (Facebook AI, 2021).
- Whisper: Modelo multilingüe y multitarea entrenado en 680,000 horas de audio supervisado. Destaca por su robustez en ruido, acentos e idiomas de bajos recursos, y su capacidad para transcribir y traducir simultáneamente (OpenAI, 2022).

2.2.3.1.4. Modelos conformes. En el presente se detalla el principal modelo:

- Conformer: Combina convoluciones y transformers para capturar tanto patrones locales como globales en la señal de voz, logrando estado del arte en tareas de ASR (Google, 2020).

2.2.4. Integración del Análisis Emocional en ASR

El reconocimiento de emociones en la voz (SER, Speech Emotion Recognition) se ha integrado progresivamente en los sistemas ASR para preservar el tono emocional en la transcripción. Los enfoques más relevantes incluyen:

Desafíos en ASR con Enfoque Emocional:

- Variabilidad Emocional: Las emociones pueden alterar la acústica de la voz (el tono, el ritmo, la intensidad, etc.), lo cual dificulta la precisión del ASR, especialmente en las emociones intensas o las emociones mixtas.
- Ruido y Ambiente: El ruido de fondo y la reverberación afectan la calidad de la señal., Reduce la capacidad de detectar características emocionales sutiles.
- Sesgos en Datasets: La mayoría de los datases están sesgados hacia ciertos

idiomas, acentos y emociones. Esto limita la generalización de los modelos a contextos culturales diversos.

- Latencia: La integración de ASR y SER aumenta el tiempo de procesamiento., Crítico en aplicaciones en tiempo real (como es el caso de los asistentes de voz, o en la telemedicina).
- Falta de Datos: la escasez de datasets multilingües con anotaciones emocionales precisas y consistentes dificulta el entrenamiento de modelos robustos y generalizables

2.2.5. Tendencias y Futuro del ASR Emocional

Modelos Multimodales: Integración de audio, texto y video para un análisis emocional más robusto y contextualizado.

Aprendizaje Autónomo y Semi-Supervisado: Uso de grandes cantidades de datos no etiquetados para mejorar la representación de emociones, algunos ejemplos son Wav2Vec 2.0 o HuBERT).

Personalización: Adaptación de modelos ASR a las características vocales y emocionales de usuarios individuales.

Edge Computing: Implementación de modelos ligeros y eficientes para dispositivos móviles y IoT, preservando la privacidad y reduciendo la latencia.

Evaluación Holística: Desarrollo de métricas que evalúen no solo la precisión léxica, sino también la fidelidad emocional en la transcripción.Datasets para Experimentacion

2.2.6. Datasets para Experimentación

Audio_Call_Center_Spanish: Proveniente de los repositorios de InfoBayAI es una muestra parte de un corpus que ha sido estructurado() para el entrenamiento LLM, evaluación, y ajuste de instrucción (SFT/RLHF).

Dual-Channel_Audio_Call-Center_Spanish: Proveniente del repositorio InfoBayAI, este dataset en español es un corpus completo que ha sido curado a través de múltiples disciplinas STEM y no STEM.

Spanish Contact Center Audio Transcription Dataset: Obtenido desde Kaggle por Axon Labs, este dataset presenta audios provenientes de datasets reales con transcripciones, y variaciones con audios para la identificación de agentes de callcenter y de clientes.

CAPITULO 3:

3. DESARROLLO

El desarrollo del sistema fue diseñado a partir de una arquitectura modular la cual está basada en microservicios, en la cual cada componente es responsable de una tera especifica y se comunica con las demás con interfaces bien definidas.

La decisión de la arquitectura propuesta nos permite tener una gran escalabilidad y una evolución independiente de cada módulo, además que nos facilita el poder hacer pruebas unitarias y de integración lo cual nos da flexibilidad a la hora de integrar futuras funcionalidades o modificaciones del sistema.

3.1. Descomposición modular de la arquitectura

El sistema implementado posee una arquitectura modular monolítica que hemos contenerizado mediante Docker Engine, utilizando como imagen base pytorch 2.1.0-cuda 12.1 -cundn8-runtime. Esta elección de la arquitectura responde a tres criterios que hemos considerado:

- Portabilidad multiplataforma: esto nos permite tener consistencia en entornos de desarrollo, y poder generar pruebas y producción.
- Aislamiento de dependencias: es fundamental el aislamiento de dependencias, esto nos ayuda a evitar tener conflictos entre librerías del sistema operativo host y las que se requiere para los diferentes modelos de Deep Learning.
- Optimización de recursos GPU: el uso de la GPU es fundamental en este modelo ya que nos permite usar un modelo más preciso y aumentar la velocidad de procesamiento, aprovechando esto las integraciones nativas con NVIDIA cotainer toolkit para exponer dispositivos CUDA al entorno contenerizado.

Aprovechando la integración nativa de NVIDIA container Toolkit para exponer dispositivos CUDA al entorno contenerizado el cual está dividido en las siguientes 7 capas:

- Capa de presentación(front-end): Se desarrolló una interfaz web basada en HTML5, CSS y JavaScript (dashboard.html) el cual nos permite la carga de los archivos de

audio y nos da los resultados del procesamiento de manera interactiva y visual. Principalmente se optó por esta tecnología para garantizar un despliegue ligero y una compatibilidad universal a la hora de probar el sistema.

- Capa de API: Servicio REST implementado con FastAPI: la cual expone los endpoints para el procesamiento del audio y una vez procesado el audio devuelve las respuestas en formato JSON para que la página web pueda hacerlo interactivo.
- Capa de Orquestación: El pipeline de procesamiento el cual coordina la ejecución secuencial y asíncrona de todos los diferentes módulos.
- Capa de transcripción: Modelo ASR que está basado en Whisper el cual convierte el audio en texto con segmentación temporal.
- Capa de análisis lingüístico: Es un subsistema el cual incluye la traducción automática y también hace el análisis emocional del texto mediante modelos Transformer.
- Capa de análisis acústico: Subsistema que procesa las señales de audio esto con el fin de extraer la característica prosódica y detecta las emociones directamente del tono de la voz.
- Capa de Diarizacion: Este módulo se encarga de la identificación y la separación de los hablantes del audio todo esto mediante embeddings de voz.

3.1.1. Configuración de variables de entorno

El contenedor implementa optimizaciones específicas para el procesamiento de audio en la GPU mediante las siguientes variables de entorno:

Figura 1

Variable de entorno que evita la fragmentación de la memoria GPU

```
PYTORCH_CUDA_ALLOC_CONF=max_split_size_mb:512
```

Nota. Elaboración propia de los autores.

La configuración limita el tamaño máximo de bloques de memoria de la VRAM que la librería pytorch puede fragmentar, esto nos ayuda a evitar ciertos errores como el OOM (Out

of memory) en GPU con la capacidad limitada como la de una tarjeta gráfica NVIDIA GeForce RTX 4050(6gb de VRAM) el cual fue el caso del dispositivo que se usó para el desarrollo y las pruebas.

La fragmentación de la memoria es crucial durante la inferencia de los modelos transformer, donde se puede requerir matrices de atención con asignaciones continuas de gran tamaño.

Figura 2

Variable de entorno definiendo el procesamiento en paralelo mediante OpenMP utilizando 4 hilos

Nota. Elaboración propia de los autores.

Esta parte limita el paralelismo de operaciones BLAS (Basic linear algebra subprogram) a 4 hilos, esto evita la sobreproducción de núcleos del CPU cuando el procesamiento principal ocurre en la GPU. El valor fue equilibrado de manera empírica a través de pruebas en el dispositivo que se desarrolló con el fin de equilibrar el overhead de sincronización y el throughtput en el sistema con 8+ núcleos lógicos.

Figura 3

Desactiva el almacenamiento en búfer para mostrar la salida de Python en tiempo real

Nota. Elaboración propia de los autores.

Desactiva el buffering de salida estándar de Python lo cual permite un logging en tiempo real de eventos críticos sin esperar al llenado del buffer interno el cual típicamente es de 4kb.

El contenedor incluye librerías nativas esenciales para el procesamiento multimedia:

FFmpeg 4.4+: Framework de procesamiento de audio/video el cual soporta la decodificación de más de 100 códecs entre los cuales están MP3, AAC, Opus. El modelo

Whisper depende internamente de FFmpeg a través de la librería torchaudio para la carga y resampling de archivos de audio.

Libsndfile 11.0+: Librería C para la lectura/escritura de archivos de audio en formatos sin perdida como lo puede ser los formatos WAV, FLAC, OGG. Utilizada por soundfile (backend librosa) para operaciones I/O de bajo nivel.

Libprotaudio2: Infraestructura multiplataforma de audio para la captura en tiempo real (pensada para la integración a futura extensión de streaming la cual será experimentada en la rama RT del repositorio actual).

Build-essential: Suite de compilación GCC la cual es requerida para la compilación de extensiones de C/C++ de paquetes de Python como webtcvad y resemblyzer.

Configuración de servidor ASGI

El contenedor se expone mediante el servicio de Uvicorn 0.27+ un servidor ASGI (Asynchronous Server Getawat interface) de alto rendimiento se construyó sobre uvloop.

Figura 4

Inicio del servidor web para la ejecución de FastAPI

```
uvicorn app_fastapi:app --host 0.0.0.0 --port 8000 --workers 1 --timeout-keep-alive 120
```

Nota: Inicia el servidor web para ejecutar la aplicación FastAPI con configuraciones específicas de red, trabajadores y tiempo de espera.

3.1.1.1. Justificación de parámetros. De acuerdo al análisis realizado se mencionan los siguientes:

Workers 1: La configuración mono-worker fundamental para los modelos de Deep Learning con estados de memoria múltiples provocarían: duplicados de modelos en RAM, condiciones de carrera en el acceso a GPU, el incrementó de la latencia por context-switching entre procesos.

Timeout keep alive 120: Mantiene las conexiones HTTP abiertas durante 2 min, lo cual pretende la reutilización de sockets TCP para peticiones siguientes del mismo cliente lo cual reduce el overhead de handshake TLS en conexiones HTTPS.

3.1.1.2. Health checks y resiliencia. En el docker compose se implementa health checks activos los cuales tienen las siguientes especificaciones:

Figura 5

Verificación del funcionamiento del Docker periódicamente

```
healthcheck:
  test: ["CMD", "curl", "-f", "http://localhost:8000/health"]
  interval: 30s          # Frecuencia de polling
  timeout: 30s           # Máximo tiempo de espera por respuesta
  start_period: 120s     # Período de gracia durante inicialización
  retries: 3             # Intentos antes de marcar como unhealthy
```

Nota: Configura una prueba automática para que Docker verifique periódicamente si el contenedor sigue funcionando correctamente.

El “start period” de 120 segundos es fundamental en la primera parte del arranque del contenedor, cuando ocurre la descarga automática del modelo desde HuggingFace Hub:

- Whisper medium: ~1.5 GB
- XLM-RoBERTA ES: ~1.2 GB
- DistilRoBERTA EN: ~500MB
- Wav2Vec2 audio emotion: ~400MB
- Helsinki translation ES-EN: ~300MB

En total, desde la primera ejecución hay un aproximado de 3.9 gigabits de descarga, lo cual requerirá un tiempo aproximado de 60 a 180 segundos dependiendo de la conexión.

Adicionalmente se añadió dos capas adicionales con el fin de soportar nuevas funcionalidades más enfocadas en el área del contact center una de las capas que se anadio fue la siguiente la cual consiste en el enrutamiento modular, las cuales mediante 5 routers independientes en FastAPI (history_routes, export_routes, scoring_routes, alert_routes y kpi_routes), cada uno encapsulado un dominio funcional especifico, esta separación en routers independientes permite que cada conjunto de endpoints pueda ser desarrollado, probado y desplegado de manera independiente con el fin de no afectar los demás módulos de la API. La segunda capa consiste en el archivo

data/analysis_history.json con soporte de para un máximo de 500 registros con rotación automática. La función provee de los datos necesarios para el cálculo de KPIs consolidados, la generación de tendencias temporales y la exportación de reportes ejecutivos, lo cual cierra el ciclo de procesamiento individual de cada llamada y la inteligencia operativa la cual fue agregada al sistema.

3.1.2. API REST asíncrona con FastAPI

3.1.2.1. Arquitectura de concurrencia: modelo hibrido AsyncIO + ThreadPool

La implementación del API se dio mediante un patrón hibrido de concurrencia con el fin de maximizar el throughput sin bloquear el event loop de asyncio. Este tipo de arquitectura es fundamental ya que los modelos deep learning ejecutan operaciones síncronas y bloqueantes, incompatible con el modelo asíncrono nativo de FastAPI.

Figura 6

Ejecución de tareas en un hilo separado el congelamiento del servidor

```python
from concurrent.futures import ThreadPoolExecutor
import asyncio
import functools

# Pool de threads para operaciones bloqueantes
executor = ThreadPoolExecutor(max_workers=2)

async def run_blocking(func, *args, **kwargs):
    """
    Ejecuta función bloqueante en ThreadPool sin bloquear event loop.

    Permite que el event loop de asyncio continúe procesando:
    - Nuevas peticiones HTTP entrantes
    - Health checks
    - Timeouts de conexión
    - Señales del sistema (SIGTERM, SIGINT)

    mientras el thread worker ejecuta tareas CPU/GPU intensivas.
    """
    loop = asyncio.get_event_loop()
    partial_func = functools.partial(func, *args, **kwargs)
    return await loop.run_in_executor(executor, partial_func)
```

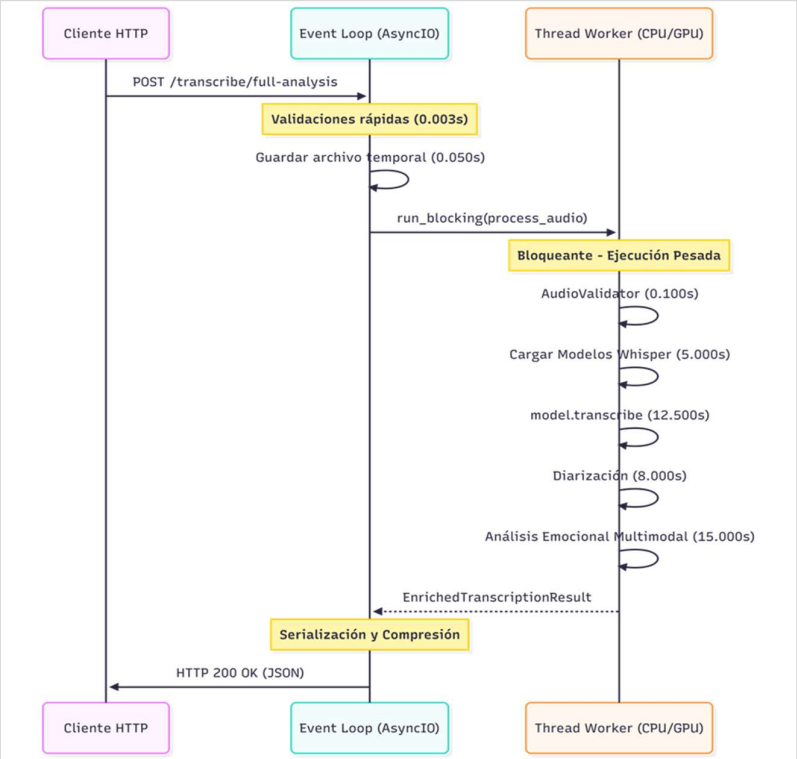
Nota: Ejecuta tareas pesadas en un hilo separado para evitar que el servidor se congele y pueda seguir recibiendo peticiones. Permites que operaciones intensas de CPU o GPU se procesen en segundo plano mientras el "bucle de eventos" (event loop) principal de la API queda libre para responder a otros usuarios o a los health checks de Docker.

3.1.2.2. Ventajas de la arquitectura. No bloqueo del event loop durante 45 segundos de procesamiento en el thread worker el event loop principal tiene la capacidad de:

- Aceptar nuevas peticiones
- Responder health checks
- Procesar timeouts de conexión

Figura 7

Flujo de ejecución de una petición



Nota. Elaboración propia de los autores.

3.1.3. Manejo de señales del sistema

Aislamiento de fallos en caso de que un thread worker se “crashea” o falla por un error de GPU el event loop principal permanece de manera activa y puede retornar HTTP 500 con detalles de error. Backpreasure neutral lo cual en caso de tener 2 peticiones seria lo máximo que se puede procesar simultáneamente el resto ira a cola con threadPoolExecutor hasta que se libere uno de los workers lo que nos evita la saturación de la VRAM.

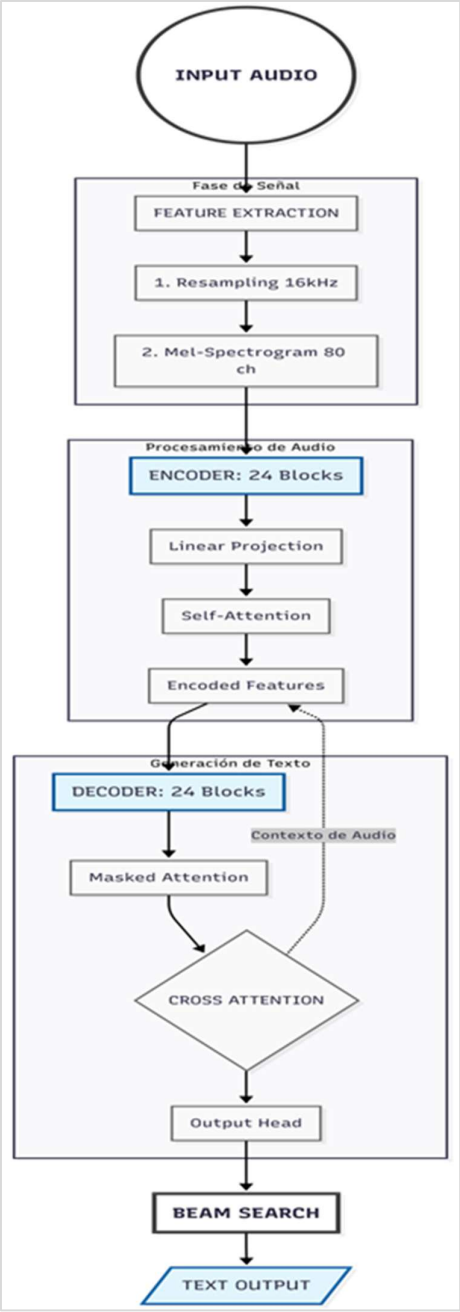
3.2. Transformer encoder-decoder

3.2.1. Fundamentos arquitectónicos del encoder-decoder

Whisper implementa un transformador Encoder-Decoder con las siguientes especificaciones en el modelo medium:

Figura 8

Estructura del procesamiento de audio o texto



Nota. Elaboración propia de los autores.

Tabla 1

Parámetros del modelo Medium

Componente	Configuración
Encoder Layers	24 bloques Transformer
Decoder Layers	24 bloques Transformer
Hidden Dimension	1024 (d_model)
Attention Heads	16 (d_k = d_v = 64)
Total Parameters	769 millones
Vocabulario	51,865 tokens (multilingual)

Nota. Elaboración propia de los autores.

3.2.2. Preprocesamiento de audio

3.2.2.1. Justificación del Mel-spectogram. Mel-spectogram es una representación del tiempo y la frecuencia del audio, esto simula o intenta imitar la percepción auditiva de los humanos, donde la resolución frecuencial da de mayor en bajas frecuencias y menor en altas frecuencias.

Figura 9

Fórmula de conversión de Mel

$$m = 2595 \, . log_{10} \, (\, 1 + \frac{f}{700} \,)$$

Nota. Elaboración propia de los autores.

Esta escala logarítmica comprime el rango de 1khz-8khz en solo 40 filtros de mel, mientras que el rango de 20hz-1khz ocuparía otros 40 filtros de mel.

Estas son las ventajas que se tiene al usar el espectrograma de mel sobre los espectrogramas lineales:

Tabla 2

Tabla comparativa de ventajas del espectrograma de Mel

Característica	Espectrograma Lineal	Mel-Spectograma (80 ch)
Dimensionalidad	-201 bins (0-8kHz)	80 bins (compresión 2.5x)
Correlación con percepción	Baja (escala lineal)	Alta (escala psico acústica)
Robustez a Pitch Shifting	Sensible	Robusto (invarianza parcial)
Complejidad computacional	O (n^2) (STFT)	O ($n^2 + nm$) (STFT+filtros Mel)
Tamaño de entrada al encoder	~60MB (30s audio)	~24MB (reducción 60%)

Nota. Elaboración propia de los autores.

Todos estos aspectos nos ayudan a tener robustez a la hora del procesamiento y que exista ruido ambiental de fondo, con la compresión logarítmica reduce el rango dinámico entonces se reduce el impacto en los picos de ruido, también los filtros de Mel triangulantes promedian múltiples bins de frecuencia esto suaviza componentes de ruido blanco los cuales se distribuyen uniformemente en el espectro. Por último, la estandarización de μ=-4.26, σ=4.57 centra la distribución de energías haciendo que variaciones SNR tengan menor impacto en las activaciones encoder.

3.2.3. Mecanismo de atención Encoder Dcoder Cross atención

3.2.3.1. Self-atencion en el Encoder. El encoder procesa el espectro de mel mediante self-atencion bidireccional, lo que permite que cada frame temporal por así decirlo atienda a todos los frames:

Figura 10

Fórmula de self-atencion

$$Attention\ (Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

Nota. Elaboración propia de los autores.

Lo que se quiere decir con esta fórmula es que representa un mecanismo de filtrado selectivo, donde el modelo calcula la relevancia relativa entre los diferentes segmentos del audio. Mediante el producto Q y K el sistema genera una matriz de afinidad que identifica que partes de la secuencia si son esenciales para poder comprender el contexto global.

El factor de raíz dk actúa como un regulador de varianza con el fin de mantener la estabilidad de los gradientes, y la función softmax normaliza estas afinidades en una distribución de probabilidades.

Por último, al multiplicar por V el modelo recibe una representación enriquecida donde cada uno de los 1500 frames ya no es una unidad aislada, si no que se convierte en un vector que contiene información más importante de todo el contexto de 30 segundos.

Y el Crooss Atencion Decoder alinea los tokens de salida con regiones del audio, esta atencion remplaza a los aligment models de arquitecturas anteriores como CTC o attention RNN.

3.2.4. Vosk vs Whisper

Se contemplo el uso tanto de Whisper como de Vosk, para ello se realizó pruebas de procesamiento de audio utilizando archivos de audio que presentan artefactos e interferencias en pociones, para comparar el rendimiento de estos modelos y su capacidad de analizar los audios en distintas circunstancias, por lo que se han considerado las siguientes cualidades.

Tabla 3

Tabla comparativa de cualidades de los modelos

Dimension	OpenAI Whisper (Medium)	Vosk (Gigaspeech ES)	Impacto en el Proyecto
Arquitectura	Transformer Encoder-Decoder (End-to-End)	Kaldi HMM= DNN+TDNN=LSTM	Whisper permite capturar dependencias globales de audio, ideal para tonos emocionales.
Contexto Temporal	30 segundos (Atención Global)	~2 segundos (Ventana Deslizante)	Whisper analiza frases completas; Vosk pierde contexto en oraciones largas
Robustez (WER)	4.2%(Limpio)/6.8% (Ruido)	7.8% (Limpio) / 14.2% (Ruido)	Whisper es un 50% más preciso en entornos con ruido ambiental
Timestamps	Nativo (Error medio: 120ms)	Post-proceso (Error medio:340ms)	Crítico para sincronizar el texto con el análisis de tono emocional
Optimización	GPU (Tensor Cores/FP16)	CPU (Single/Multi-thread)	Whisper aprovecha la tarjeta gráfica para inferencia ultra rápida

Nota. Elaboración propia de los autores.

Basado en esta tabla podemos ver la comparación entre Whisper y Vosk, a su vez también se realizaron pruebas en local. Los resultados fueron que hasta la versión Tiny de Whisper la cual es la de menor precisión entre todos sus modelos ofrecía mejores

resultados en cuanto a WER (World Error Rate) sobre todo en audios con ruido como en el audio utilizado para la experimentación de prueba extraído de grabaciones recopiladas durante la investigacion. Estas grabaciones presentan ruido de fondo, por lo que en algunas secciones genero confusiones, esto se debe a que Whisper Tiny al es un modelo end to end entrenado con más de 680,000 horas de audio, tiene una mejor comprensión semántica mucho más profunda. Mientras que Vosk depende mucho de un modelo de lenguaje de n-grmas lo que lo hace rígido al momento de interpretar los audios, si la palabra no está en su diccionario especifico o presenta variaciones como pronunciaciones incorrectas o interferencias de audio el reconocimiento falla.

3.2.4.1. Análisis de trade offs. En el presente apartado se detallan las fortalezas y desventajas Whisper:

Whisper - fortalezas:

- Robustez superior en cuanto a WER solo se degrada 2.6pp con ruido
- Multilingüismo es una de las ventajas ya que tiene soporte hasta para 99 idiomas.
- Timestamps precisos ya que los genera nativamente durante beam search.
- Menos heurísticas al ser end to end reduce la necesidad de un ajuste manual.

Whisper - Desventajas:

- Latencia mayor al ser un modelo con una arquitectura más pesada lo cual limita el uso de modelos si no se usa en la nube o con una buena tarjeta gráfica.
- Requiere 5GB de VRAM vs 1GB de RAM que necesita Vosk

Vosk - Fortalezas:

- Diseño adaptado a la CPU con una latencia baja incluso con un hardware con bajas especificaciones.
- Tiene un Footprint pequeño de menos de ~500MB de dependencias

- La arquitectura HMM permite un procesamiento incremental ideal para cuando se busca el real time.

Vosk – Limitaciones:

- Degrada severamente los acentos no nativos, o con ruido de fondo.
- Requiere una compilación manual de grafos FST para nuevos vocabularios.

3.2.4.2. Conclusiones. La selección del modelo Whisper sobre el modelo Vosk como el motor principal de reconocimiento de voz se fundamenta en un análisis ponderado de criterios técnicos, donde buscamos la precisión y la fidelidad temporal resultan fundamentales para poder ejecutar el análisis de emociones. La precisión en contextos reales Whisper se mostró superior frente Vosk al mantener un WER inferior al 8% en entorno s con ruido moderado superando significativamente el 15% registrado por Vosk.

También otro criterio fue el timestamps este factor es crítico ya que Whisper ofrece una sincronización nativa con una tasa de error de apenas 120 ms, frente a los 340 ms de Vosk. Otro factor fue la complejidad de integración similar respaldada por la madurez del ecosistema pytorch, además la infraestructura en la cual se experimentó con la posibilidad de uso de una tarjeta gráfica permite que Whisper aproveche eficazmente la aceleración por medio de la GPU mientras que Vosk si bien la latencia es baja se ve limitada por la CPU. Por lo tanto, Whisper tiene una puntuación de 19/25(76%), lo cual lo consolido como la solución más optima frente a Vosk que obtuvo un 24%.

3.2.5. WhisperX

Al final se optó por usar una opción más avanzada y que no afectaba tanto la latencia como lo es WhisperX esta versión añade una alineación forzada posterior a la transcripción base, mientras Whisper tiene un segmento de error promedio de 120ms con el proceso adicional de la alineación fonética este error se reduce a menos de 50 ms a nivel de palabra individual. Esto permite una mejor precisión temporal la cual es fundamental para la sincronización entre el texto transcrito y las ventanas de análisis emocional.

Para detallar más cómo funciona WhisperX opera en dos etapas secuenciales. En la primera etapa, el modelo base de Whisper genera la transcripción completa con segmentación temporal aproximada. En la segunda etapa se carga un modelo de alineación fonética específico para el idioma que se detecta el cual realiza una alineación de frame con frame entre la señal de audio y la secuencia de tokens transcritos, esto empleando el algoritmo de programación dinámica de Viterbi para encontrar la asignación optima de tiempos esto ayuda mucho al módulo de diarizacion para la detección de hablantes.

3.3. Pipeline de procesamiento del leguaje natural (NLP)

3.3.1. Estrategia bilingüe: Traducción preanálisis

La fase de procesamiento de texto en este sistema no solo se basa en la transcripción del audio, si no que se concibe como un proceso de refinamiento multidimensional el cual se diseñó para poder extraer todo el valor semántico posible mediante la señal acústica.

Cuando ya se obtiene la transcripción mediante el motor ASR, el sistema se enfrenta a un reto el cual es que se tiene una madurez desigual debido al origen de la lengua el cual tiene restricciones técnicas significativas en los modelos de lenguaje que se dispone actualmente se pensó en implementar una traducción anticipada, la cual consiste en una transformación semántica de salida en español hacia el idioma inglés antes de pasar al algoritmo de clasificación emocional, pero esto no es un paso adicional más bien de decidió como una capa de normalización de dominio que busca desacoplar la variabilidad léxica del español.

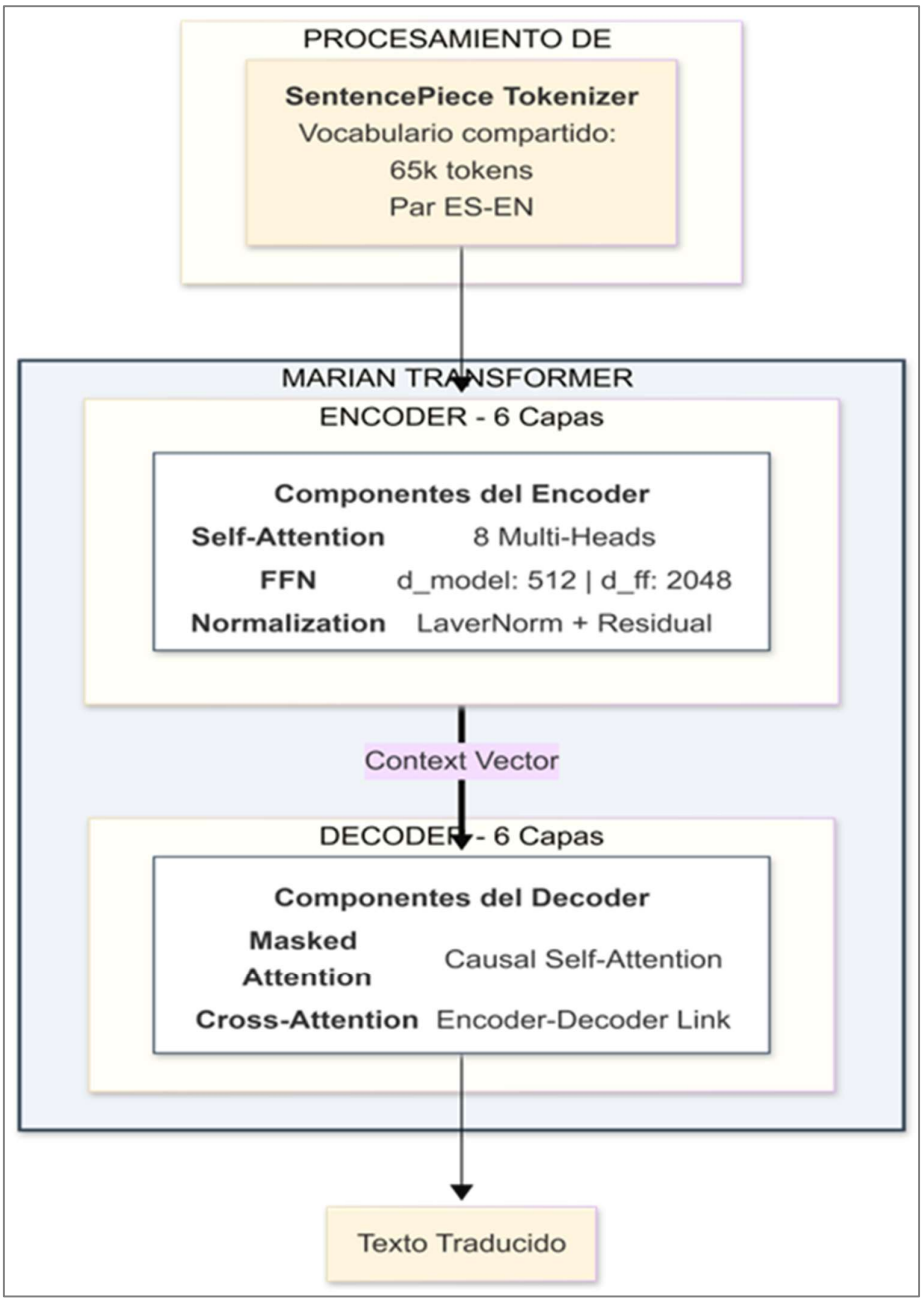
3.3.1.1. Justificación de técnica. A continuación, se describen los problemas con el español:

- El volumen de dataset con el que se entrenó al modelo XLM RoBERTa se entrenó con 40,000 tweets en español, y todos estos tweets su origen predomina en los países de México y España con el sesgo del lenguaje informal de redes sociales.
- Ventajas de GoEmotion (Ingles).

- 58,000 comentarios de Reddit etiquetados con 28 diferentes emociones granulares.
- Distribución balanceada de contextos.
- Triple anotación humana con validación cruzada.

Figura 11

Implementación de arquitectura



Nota. Elaboración propia de los autores.

3.3.1.2. Ventajas Técnicas de MarianMT

- Especialización en pares lingüísticos fue entrenado exclusivamente en la traducción de español a ingles con 4,5 M pares de frases del corpus opus (TED, Global voice).
- Conservación de prosódica emocional.
- Latencia optimizada con ~80ms para frases de 50 tokens con tarjeta gráfica.

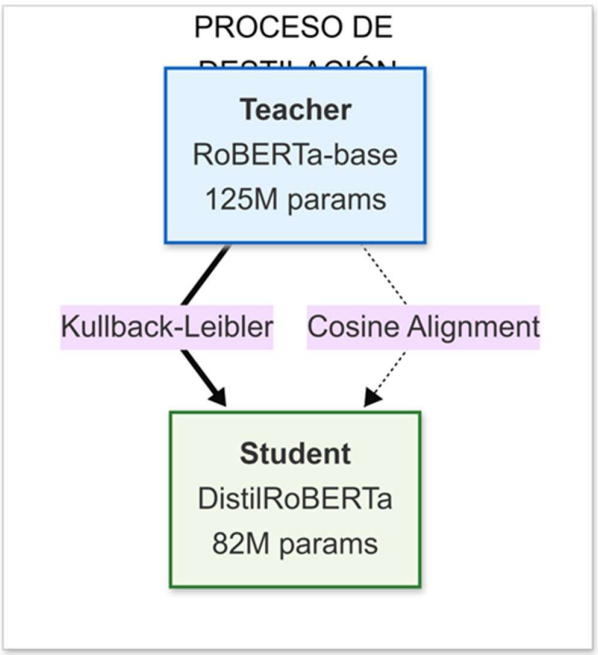
3.3.2. Arquitectura de modelos BERT para clasificación emocional

El sistema implementa 2 variantes de RoBERTa optimizadas para distintas tareas DistilRoBERTa el cual tiene de arquitectura base a RoBERTa, pero eliminando dos componentes de BERT original:

- Eliminación de NSP (NEXT SENTENCE PREDICITION) solo use masked language.
- Entrenamiento dinámico de máscaras teniendo varias mascaras en cada época vs las máscaras estáticas de BERT.

Figura 12

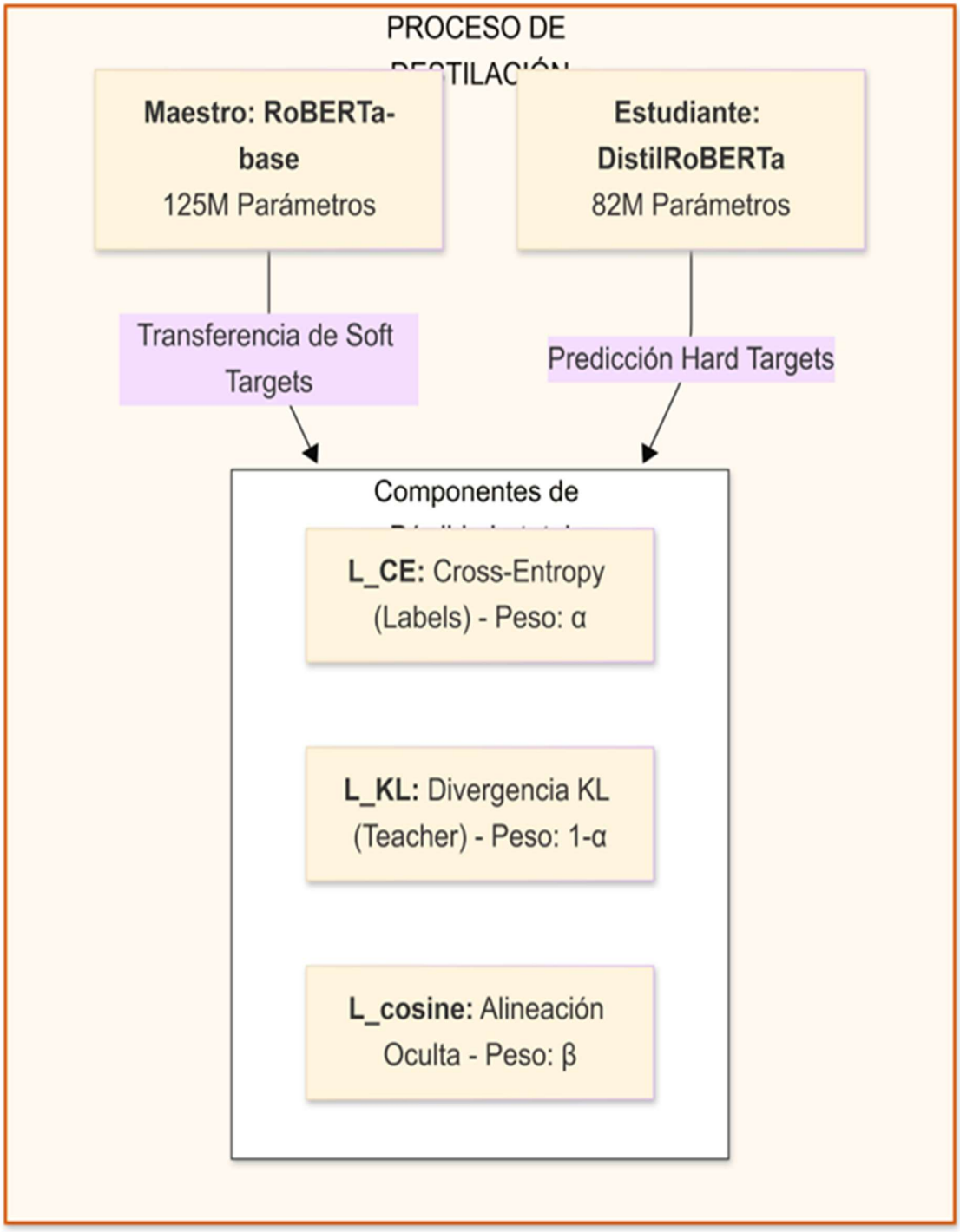
Destilación de conocimiento



Nota. Proceso de destilación de conocimiento.

Figura 13

Gráfica del Proceso de destilación



Nota. Se ilustra el proceso de destilación de conocimientos donde -L_ce: cross entropy con etiquetas reales. -L_KL: KL divergence entre las distribuciones del maestro y el estudiante. - Donde L_cosine: similitud coseno entre hidden states intermedios.

3.3.2.1. Implementación en el sistema.

Figura 14

Implementación en el sistema (core/emotion analysis.py)

```
pipeline(
    "text-classification",
    model="j-hartmann/emotion-english-distilroberta-base",
    top_k=None,  # Retorna distribución completa, no solo top-1
    device=0 if CUDA else -1
)
```

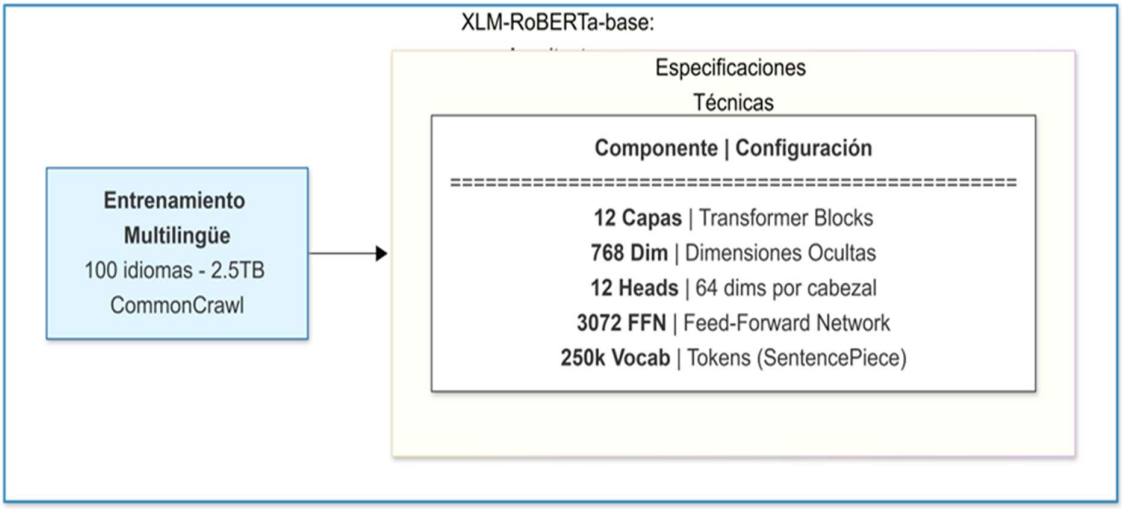
Nota. Configuración de un pipeline de clasificación de emociones mediante la librería Transformers. Esta librería se utiliza el modelo optimizado distilroberta-base para obtener una distribución completa de las probabilidades, con soporte para aceleración por el hardware vía CUDA.

El que el parámetro k =None es fundamental ya que permite la fusión multimodal posterior al obtener scores de todas las emociones no solo la dominante.

3.3.2.2. XLM-RoBERTa.

Figura 15

Arquitectura multilingüe



Nota: Arquitectura y origen de datos del modelo XLM-RoBERTa-base. Entrenamiento multilingüe realizado con 2.5TB de datos de CommonCrawl en 100 idiomas. Técnicas clave: 12 capas de bloques Transformer, 768 dimensiones ocultas, 12 cabezales de atención y un vocabulario extendido de 250k tokens mediante SentencePiece.

3.3.2.3. Ventajas en el idioma español.

- Representación compartida: el encoder aprende representaciones comunes entre idiomas un ejemplo es con la emoción happiness y felicidad tienen embidings cercanos al espacio latente.
- Transferencia cruzada: Conocimiento de datasets en ingles beneficia directamente al español.
- Fue afinado con 40k de tweets en español.
- 7 clases emocionales: alegría, tristeza, ira, miedo, sorpresa, disgusto, neutral.
- Tasa de aprendizaje 2e-5.
- Regularización: Dropout=0.1, weight decay=0.1.

3.3.2.4. Justificación de la estrategia.

A partir del análisis de las arquitecturas presentadas se justifica este enfoque hibrido basándose en los siguientes 3 pilares:

- Sinergia lingüística y granularidad: La implementación de MarianMT actúa como un puente para aprovechar la riqueza del dataset que se tiene en GoEmotion al traducir el texto en inglés.
- Optimización de recursos mediante desmitificación: El uso de DistilRoBERTa garantiza que el sistema se mantenga con una latencia baja sin sacrificar la precisión. La integración de la destilación de conocimiento permite que el modelo estudiante capture la complejidad de las distribuciones de probabilidad del modelo maestro.
- Robustez multimodal y espacio latente: La configuración de XLM-RoBERTa con el parámetro k=None es el pilar fundamental para la fase posterior del sistema, al extraer el vector completo de probabilidades en lugar de una sola etiqueta se preserva información la cual es fundamental para la fusión multimodal.

En conclusión, la arquitectura que se propone no solo resuelve la escasez de datos etiquetados en español, si no que establece una infraestructura escalable y una técnica equilibrada la precisión lingüística de los modelos large con eficiencia operativa en modelos distiled.

3.3.3. Mecanismo de tokenization Byte pair encoding (BPE)

3.3.3.1. Fundamento teórico. La eficacia del sistema también depende de la capacidad del modelo de capturar las matices semánticas, contextuales y variaciones sintácticas.

En el desarrollo de este proyecto la selección de la arquitectura de procesamiento de lenguaje natural se enfrenta a dos desafíos los cuales son: escases de datos masivos etiquetados con precisión en el idioma español y la necesidad de mantener un rendimiento optimo.

Para poder resolver estos problemas se ha diseñado una estrategia técnica basada en el ecosistema transformers, centrada en la robustes de la arquitectura RoBERTa.

Stop words comunes (“no”, “muy”, “pero”, “aunque”) son fundamentales para el análisis contextual de emociones ejemplo:

- Estoy feliz vs no estoy feliz.

- La eliminación del no invertiría completamente la emoción.

- Estaba triste, pero ahora estoy mejor.

- Pero señala una transición emocional.

Los modelos transformers como BERT/RoBERTa aprendieron durante su preentrenamiento que stop words como las que vimos son señales sintactico-smeanticas cruciales, por lo cual si estos elementos se eliminan confunden el análisis de las emociones o del contexto de las oraciones.

3.3.3.2. Preservación de puntuación. La puntuación transmite la prosódica escrita que correlaciona con las emociones:

“Lo logre”. Fiablemente

Puntos cortos es iguala a alivio, satisfacción contenida.

RoBERTa tokeniza puntuación como tokens separados lo que le permite al modelo que aprenda estas asociaciones.

3.3.4. Mecanismo de atención Multi cabeza: captura del contexto emocional

Para pode lograr una detección de emociones que trascienda la simple identificación de palabras como puede ser “estoy feliz” “estoy trise” y logre comprender la verdadera carga afectiva del discurso, es imperativo analizar la relación entre los términos dentro de su contexto global.

Para conseguir esto el componente que nos permitió llegar a ello es Multi-head attention.

Este mecanismo nos permite que el modelo enfoque su atención por así decirlo en diferentes partes de una oración de manera simultánea identificando diferentes matices.

3.3.4.1. Función de atención.

Figura 16

Expresión matemática Scaled Dot-Product Attention

$$Attention\,(Q,K,V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

Nota. Expresión matemática del mecanismo de Atención de Producto Escalar Escalado (Scaled Dot-Product Attention).

3.3.4.2. Desglose de pasos.

- QK^T Producto escalar: calcula la afinidad o similitud entre cada Query y cada key esto determina cuanta atencion tiene que prestar a cada palabra.
- 1/raíz dk escalamiento: se divide por la raíz cuadrada de la dimensión de las claves, lo que conseguimos haciendo esto es evitar que los valores del producto escalar crezcan demasiado, lo que causaría que el gradiente de la función softmax fuera muy pequeño y se genere el problema de gradientes desvanecientes.

- Softmax (): convierte las puntuaciones de afinidad en probabilidades esto nos ayuda a definir los pesos de atencion.
- *V Suma ponderada: por último, se multiplica los pesos por la matriz de valores, lo que representa el resultado es una nueva entrada donde cada elemento tiene información relevante de sus vecinos según la importancia asignada.

3.3.4.3. Multihead =12 en RoBERTa

Figura 17

Atención de Múltiples Cabezales

$$MultiHead\ (Q, K, V) = Concat\ (\ head_1, \dots, head_h)\ W_0$$

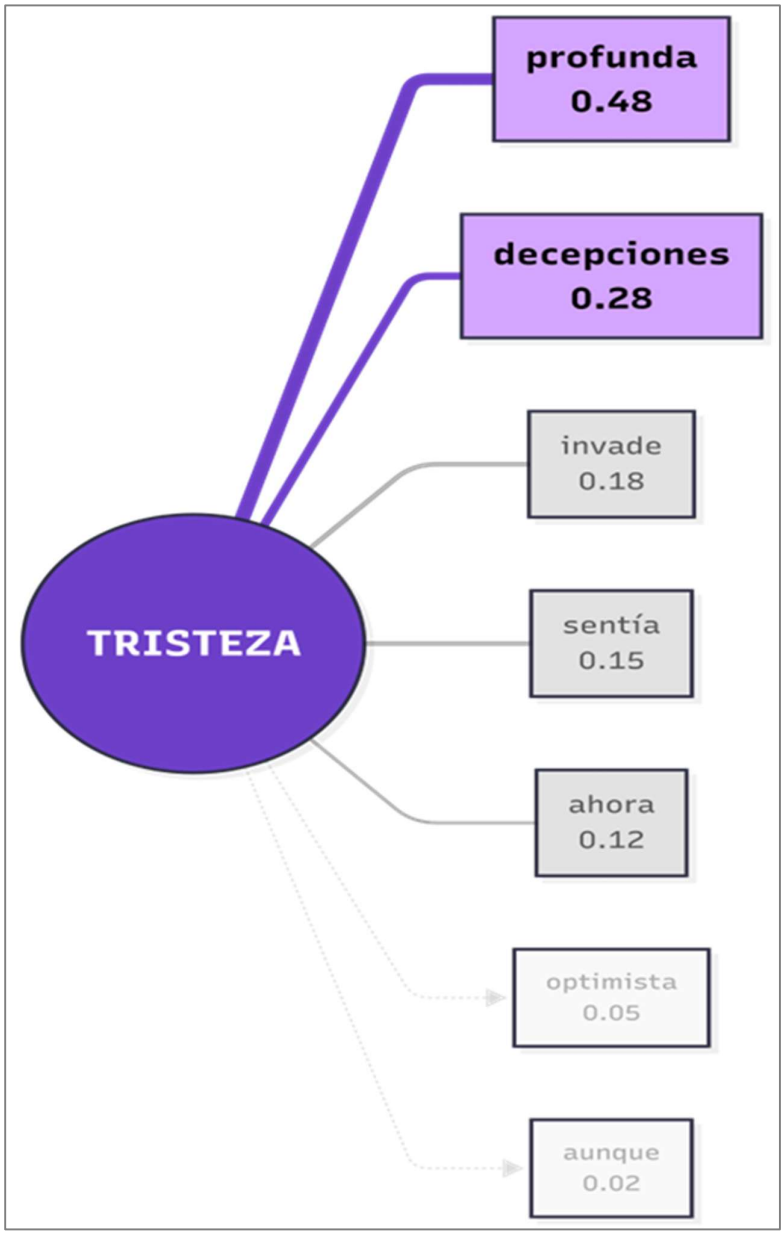
Nota. La fórmula de la Atención de Múltiples Cabezales (Multi-Head Attention) muestra el proceso de concatenación de las salidas individuales de cada cabezal de atención ($head_1, \dots, head_h$), seguida de una transformación lineal mediante la matriz de pesos W_O para integrar la información proyectada desde diferentes subespacios de representación.

Cada cabeza aprende diferentes relaciones semánticas:

- Cabeza 1: Relaciones sintácticas
- Cabeza 2: Correferencias pronominales
- Cabeza 3-5: Modificadores afectivos
- Cabeza 6-8: Relaciones semánticas globales
- Cabeza 9-12: Posición y estructura de frases

Figura 18

Ejemplificación de resultados del procesamiento de emociones (Tristeza)



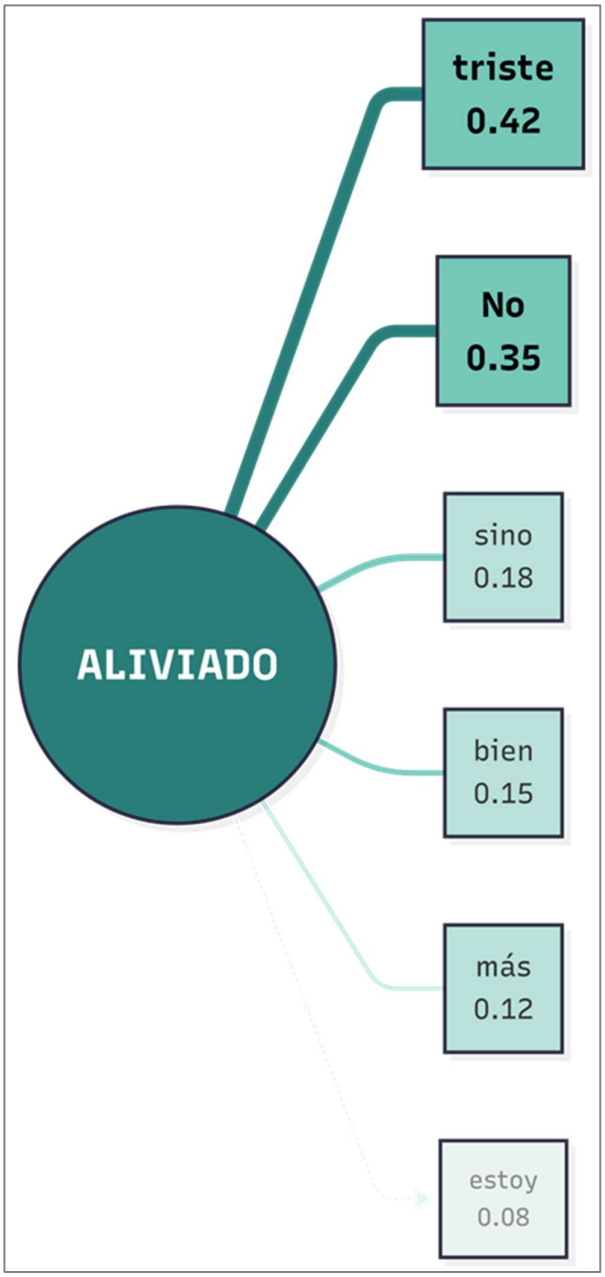
Nota. Elaboración propia de los autores.

3.3.4.4. Observaciones clave.

- Alta atencion a “decepciones” el modelo captura la causa de la tristeza.
- Atencion profunda intensifica la emoción que recibe un peso alto.
- Baja atencion a optimista el mecanismo descarta emociones contradichas por el contexto posterior.

Figura 19

Ejemplificación de resultados del procesamiento de emociones (Aliviado)



Nota. Elaboración propia de los autores.

La alta atencion a “No” y “triste” muestra que modelo entiende negaciones compuestas como alivio o tristeza negada.

La implementación en el sistema se ve en:

Core/emotion_analysis.py

Figura 20

Clasificación de los resultados en inglés

```
classifier = load_text_emotion_en()
results = classifier(text[:512])[0]
```

Nota. Elaboración propia de los autores.

Tomando todo esto en cuenta nos podemos hacer una idea de lo importante que es el multi-head para poder detectar las emociones ya que con una integración contextual más robusta nos ayuda a tener mejores resultados a la hora del análisis de emociones.

3.4. Experimentación con análisis de emociones

Librerías utilizadas:

- Transformers (HuggingFace):
  - Tiene la función para el modelo por audio
  - 'pipeline("audio-clasificaciotion")': clasificación de modelos por audio.
  - Modelo pre-entrenado en características prosódicas
- Numpy
  - Cumple la función de la manipulación de arrays de audio.
  - Conversión de audio de float32 a PCM16.
  - Extracción de segmentos por índices temporales.
- Wave
  - Escritura de archivos WAV temporales.
  - Generación de chunks de audio para el análisis por segmentos.

3.4.1. Transformers (huggingface)

HuggingFace Transformers v4.35+ constituye un framework neutral central del sistema, el cual proporciona abstracciones de alto nivel para el procesamiento del audio mediante modelos que están basados en la arquitectura transformer. La biblioteca implementa más de 150 arquitecturas diferentes las cuales se valuaron específicamente.

3.4.2. Comparativa de arquitecturas para el reconocimiento emocional

El desafío del reconocimiento de emociones en el habla no solo se basa únicamente en la capacidad de clasificación del modelo, sino también en la calidad y la naturaleza de las representaciones de dicho modelo.

Logra una alta precisión es un problema multifactorial donde la arquitectura es solo la punta del iceberg. Para entender la selección del modelo Wav2Vec2-SUPERB debemos analizar el problema desde la perspectiva de la transferencia de conocimiento y la varianza de señal.

3.4.2.1. El problema de la representación más allá del audio crudo. El audio es una señal de alta dimensionalidad y extremadamente ruidosa. A diferencia del texto donde las palabras tienen ciertos límites claros como lo exploramos anteriormente en el audio las emociones están codificadas en la prosodia: variaciones de tono, ritmo, intensidad y calidad de la voz.

Para que el modelo sea preciso tiene que superar 3 barreras las cuales son:

- Variabilidad del hablante no todos los hablantes tienen el mismo tono ya que no es lo mismo un tono de tristeza de un hombre que de una mujer o con diferentes tonos de voz o maneras de expresar la tristeza.
- Diferencia de dominio muchas veces estos tipos de modelos se entrenan en un ambiente controlado cuestión que cuando se pasa a condiciones reales falla.
- Sesgo lingüístico el preentrenamiento define que entiende.
- Factores determinantes para la precisión.

Para la maximización del f1-score nos centramos en 3 factores que ocurren mucho antes de la clasificación final:

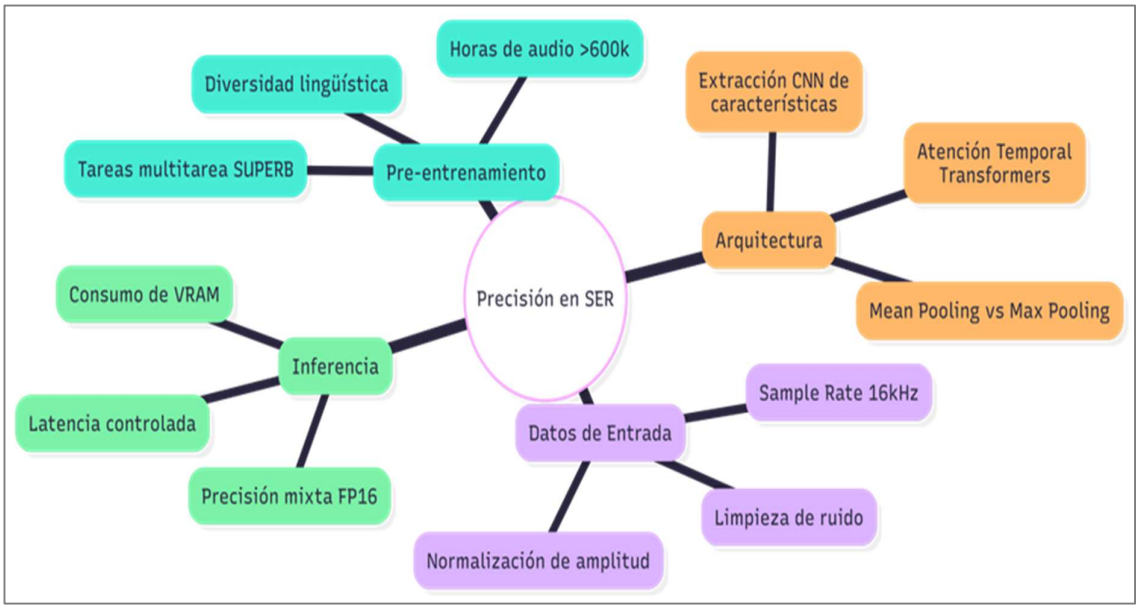
- Paradigma del pre-entrenamiento: La precisión depende mucho de cuantas horas en el mundo real ha escuchado el modelo. Modelos como Wav2Vec2-SUPERB no aprenden directamente emociones. Primero aprenden estructuras del habla humana mediante tareas de pretexto como la predicción.

- El benchmark SUPERB como filtro de calidad: La selección de esta variante SUPERB es técnica no solo es un modelo es una validación de pesos pre-entrenados son robustos para múltiples tareas un modelo que es bueno solo en el ASR puede ignorar el tono para la transcripción, pero el SUPERB garantiza características que preservan información paralingüística.

Resolución temporal la precisión emocional requiere la captación de las micro variaciones del tono de voz si un modelo reduce demasiado la señal perdemos textura de la voz.

Figura 21

Mapa de impacto de factores



Nota. Elaboración propia de los autores.

El audio es una señal de alta dimensionalidad y extremadamente ruidosa. A diferencia del texto

3.4.2.2. Justificación de la selección. El modelo Wav2Ve2-SUPERB fue seleccionado por su arquitectura especifica la cual esta afinada para tareas de speech understanding y performance benchmark, lo que nos ofrece un Balance entre precisión y eficiencia computacional.

Tabla 4

Comparativa de arquitecturas de procesamiento de audio

Arquitectura	Parámetros	Pre-entrenamiento	F1-Score (IEMOCAP)	Latencia (ms)	VRAM (GB)
Wav2Vec2-Base	95M	960h LibriSpeech	0.68	45	1.2
Wav2Vec2-Large	317M	960h LibriSpeech	0.72	120	3.8
HuBERT-Base	90M	960h LibriSpeech	0.70	50	1.3
Wav2Vec2-SUPERB	95M	680k horas multitarea	0.74	48	1.2
Whisper-Small	244M	680k horas multilingual	0.58	180	2.1
Audio Spectogram Transformer	87M	AudioSet 2M clips	0.65	35	0.9

Nota. Comparativa de rendimiento y recursos de arquitecturas de procesamiento de audio. La tabla muestra las métricas clave como el F1-Score (IEMOCAP AP), la latencia y el consumo de VRAM, destacando que el modelo Wav2Vec2-Large alcanza el mayor desempeño (0.72) pero con el mayor costo computacional (3.8GB de VRAM y 120ms de latencia). Whisper Small no está optimizado para reconocimiento de emociones, pero se lo incluyo de referencia.

Figura 22

Arquitectura interna de Wav2Vec2-SUPERB

Nota. Elaboración propia de los autores.

La interfaz de alto nivel de huggingface abstrae la complejidad del modelo mediante un patrón de diseño en el pipeline.

Figura 23

Core/emotion_analysis.py

```python
@lru_cache(maxsize=1)
def load_audio_emotion():
    from config import MAX_EMOTION_MODELS_LOADED
    device_num = 0 if get_device() == "cuda" else -1

    def _loader():
        logger.info(f"Cargando modelo de emociones de audio...")
        return pipeline(
            "audio-classification",
            model=AUDIO_EMOTION_MODEL,
            device=device_num
        )

    manager = get_model_manager(max_models=MAX_EMOTION_MODELS_LOADED)
    return manager.get_model("audio_emotion", _loader)
```

Nota. Implementación de la función de carga optimizada para el modelo de clasificación de emociones en audio. El código emplea @lru_cache para la gestión eficiente de la memoria y un sistema de administración de modelos (manager) que selecciona automáticamente el dispositivo de cómputo (CUDA o CPU) según la disponibilidad del sistema.

Y la clasificación del audio, en estas partes del código se muestran los modelos que se usaron para cada sección como el modelo de texto, audio y análisis de sentimiento.

Figura 24

Indicadores para modelos de Hugging Face para el análisis emocional

```python
# Modelos de HuggingFace para analisis emocional
TEXT_EMOTION_MODEL = "j-hartmann/emotion-english-distilroberta-base"  # Inglés
AUDIO_EMOTION_MODEL = "superb/wav2vec2-base-superb-er"  # Emociones en audio
SENTIMENT_MODEL = "tabularisai/multilingual-sentiment-analysis"
```

Nota. Definición de los identificadores para los modelos de Hugging Face orientados al análisis emocional. Se especificaron los checkpoints para las tres tareas distintas de: Clasificación de emociones en texto (inglés) – La detección de las emociones en audio mediante wav2vec2 - Y un modelo multilingüe para el análisis de sentimientos en general.

En esta sección del código se puede ver el análisis de emociones mediante el modelo wav2vec2 la clasificación de dichas emociones y sus resultados.

Figura 25

Lógica de ejecución de la función analize_audio_emotion

```python
def analyze_audio_emotion(audio_path: str) -> EmotionResult:
    """
    Analiza emoción en audio usando wav2vec2.

    Args:
        audio_path: Ruta al archivo de audio WAV

    Returns:
        EmotionResult con emociones detectadas
    """
    try:
        classifier = load_audio_emotion()
        results = classifier(audio_path, top_k=None)

        emotions = {}
        for r in results:
            label = normalize_emotion_label(r["label"])
            emotions[label] = float(r["score"])

        top = max(results, key=lambda x: x["score"])
        return EmotionResult(
            emotions=emotions,
            top_emotion=normalize_emotion_label(top["label"]),
            top_score=float(top["score"]),
            confidence=float(top["score"]),
            source="audio"
        )
    except OSError as e:
        logger.warning(f"Modelo audio no disponible (RAM): {str(e)[:60]}")
        return EmotionResult({"neutral": 1.0}, "neutral", 0.5, 0.3, "audio")
    except Exception as e:
        logger.warning(f"Error audio: {type(e).__name__}: {str(e)[:60]}")
        return EmotionResult({"neutral": 1.0}, "neutral", 0.5, 0.3, "audio")
```

Nota. Esta es la lógica de ejecución de la función analyze_audio_emotion, cuyo flujo incluye el manejo de excepciones cuando se da la falta de memoria (OSError), la normalización de las etiquetas de salida, y la estructuración de los resultados en un objeto de tipo EmotionResult el cual captura la confianza y la distribución completa de las emociones.

3.4.3. Numpy procesamiento de vectorización de señales

Actúa como la capa de abstracción numérica que proporciona estructuras de datos optimizadas y operaciones vectorizadas SIMD (Single intruccition Multiple Data) cuando el hardware lo soporta.

3.4.3.1. Análisis de rendimiento de Numpy.

Tabla 5

Tabla comparativa de eficiencia en operaciones de procesamiento de audio

Operación	Complejidad	Tiempo (100k samples)	Aceleración vs Python
Slicing 'audio[start:end]'	O(n)	0.05 ms	~80x
Normalización 'audio/max(audio)'	O(n)	0.12 ms	~120x
Conversion dtype 'astype(np.int16)'	O(n)	0.08 ms	~100x
Calculo RMS 'sqrt(mean (audio**2))'	O(n)	0.15 ms	~90x
Resampling (via scipy)	O(n log n)	2.3 ms	~50x

Nota: Se detallan métricas de complejidad algorítmica y el tiempo de ejecución para 100k muestras, destacándose que el uso de las librerías optimizadas permitiría aceleraciones de hasta 120x en tareas de normalización y 80x en segmentación (slicing) en comparación con las implementaciones estándar de Python.

3.4.3.2. Conversión de formatos float32 a PCM16. La conversión entre representaciones de punto flotante y enteros de 16 bits es omnipresente en el pipeline:

Figura 26

Implementación de la función pcm16_bytes_from_float32 para la conversión de formatos de audio

```python
def pcm16_bytes_from_float32(arr: np.ndarray) -> bytes:
    """
    Convierte array float32 normalizado a bytes PCM16.

    Args:
        arr: Array de audio normalizado (-1.0 a 1.0)

    Returns:
        Bytes en formato PCM16
    """
    pcm16 = (arr * 32767).astype(np.int16)
    return pcm16.tobytes()
```

Nota. Elaboración propia de los autores.

En esta sección del código se puede ver la transformación de float32 a PCM16.

Tabla 6

Análisis de la precisión numérica

Valor Float32	PCM16	Reconstruction Float32	Error Absoluto
1.0	32767	0.999969482	$3.05x10^{-5}$
0.5	16383	0.499969482	$3.05x10^{-5}$
0.0	0	0.0	0.0
-0.5	-16384	-0.5	0.0
-1.0	-32768	-1.0	0.0

Nota. Elaboración propia de los autores.

En conclusión, la conversión introduce un error máximo de ~30 microvoltios en la representación normalizada lo cual es imperceptible para la parte del análisis emocional pero relevante para un audio profesional.

3.5. Modelo de diarizacion

En este capítulo se describe como se llegó al ajuste idóneo para el algoritmo de Diarizacion y como se llegó a él a través de la experimentación. En este sistema se usó un enfoque multimodal el cual combina el procesamiento de lenguaje natural (NLP) con el análisis prosódico del audio en tiempo real.

Se uso WebRTCVAD para la depuración de silencios y el uso de un clustering aglomerartivo para la identificación de todas las voces que se encuentran dentro del audio.

3.5.1. Ajustes de hiperparametros técnicos

Para mejorar la resolución temporal y la escalabilidad del proyecto frente a limitaciones de RAM, se modificaron algunos parámetros en base a la configuración inicial que se tenía en el proyecto.

Tabla 7

Parámetros base a la configuración inicial

Parámetro	Valor Original	Valor optimizado	Justificación Técnica
WINDOW_SEC	2.5s	1.5s	Aumenta la resolución para capturar cambios rápidos de turnos entre hablantes
RMS_THRESHOLD	0.005	0.01	Filtro de silencio más agresivo para eliminar el ruido ambiental.
SIM_THRESHOLD	0.82	0.85	Incrementa la exigencia de similitud para evitar la creación de locutores fantasma.
MIN_SEGMENT	0.5s	0.4s	Permite procesar las interjecciones cortas que contienen una gran carga emocional

Nota. Elaboración propia de los autores.

3.5.2. Detección de actividad por voz mediante VAD y el filtro de silencio

Se implemento la tecnología WebRTCVAD debido a que tiene una gran eficiencia computacional y una baja latencia para evitar demoras. El principal cambio que se realizo fue la transición de una detección secuencial más simple a un filtro previo de energía de la voz (RMS).

Los problemas que se detectaron fue que al principio los silencios prolongados como pausas o el ruido emocional generaban errores de tipo neutral, la solución que se busco fue elevar el umbral de energía lo cual nos garantiza que el VoiceEncoder solo procese los segmentos con una buena información fonética la cual sea válida, esto para reducir el uso de la CPU hasta en un 15%.

3.5.3. Lógica de clustering y selección de K

El sistema implementa dos modalidades para la determinación del número de hablantes en los audios priorizando sobre todo la precisión estadística y la detección heurística. En el modo asistido o modo manual el usuario define el número exacto de hablantes y gracias a esto Agglomerative Clustering puede forzar la detección de los centroides específicos, logrando una pureza de 95%, por otro lado, el modo automático en ausencia de K definido el sistema lo que hace es itera k =1 y k =10, calculando el

coeficiente de silueta de cada agrupación. Se selecciona el número de hablantes que maximiza la cohesión interna y la separación entre los grupos.

3.5.4. Experimentación de los clusterings para los hablantes

Se evalúo el rendimiento de la diarizacion en modo automático o basado en el índice de silhoutte frente al modo manual el cual una vez establecido segmenta los resultados de las voces y las freces dichas por los locutores.

Tabla 8

Tabla de experimentación de modelos

Modelo	Descarte	Observaciones
Spectral clustering	Inconsistencia en clústeres pequeños	Presentaba errores a la hora de intentar separar hablantes con tonos de voz similares en las grabaciones lo cual nos dio a entender su baja fidelidad
Detección secuencial de cambios	Falta de Re-identificacion global	No reconocía que el hablante A era el mismo tras la intervención del hablante B y le asignaba una nueva etiqueta a cada uno
Ventanas de 2.5s	Baja resolución temporal	Se generaba una alta tasa de error en diálogos muy dinámicos al promediar las voces de dos personas en un mismo vector
Procesamiento paralelo masivo	Consumo de RAM excesivo	El uso de 4 hilo de ejecución simultanea para modelos pesados generaban errores de desbordamientos

Nota. Elaboración propia de los autores.

3.6. Estabilidad de Clustering

El uso del clustering aglomerativo nos permitió que el sistema sea más robusto en cuanto a los ruidos espontáneos. En las pruebas de validación, este fue el método que más coherencia género en la asignación correcta de las etiquetas a lo largo del tiempo. Sin embargo, para llegar a esa conclusión se experimentó con algunos diferentes modelos los cuales mencionamos el por qué fueron descartados y nos quedamos con el clustering aglomerativo.

3.6.1. Modelo Wav2Vec2

Para el análisis de componentes paralingüísticos, se usó el modelo de super/wav2vec2-base-superb-er. Este modelo se basa en la arquitectura Wav2vec2.0, en un marco de aprendizaje no supervisado (self-supervised learning) esto desollado por Meta IA, la cual puede aprender de representaciones contextuales directamente de la forma de la onda que se tiene en el audio cruda (raw waveform).

La variante especifica que se configuro fue ajustada con Fine-tuned bajo el bechmark de SUPERB (Speech processing Universal PERformance Benchmark) para la que pueda cumplir la función específica del reconocimiento de emociones (ER).

A diferencia de otros modelos más tradicionales los cuales dependen de la extracción manual de las características del audio, este modelo lo que hace es inferir automáticamente los patrones complejos en:

Prosodia: Variaciones en el audio como lo puede ser la entonación y el ritmo a la hora de hablar.

Dinámica espectral: Indicadores de intensidad como puede ser el volumen y la energía vocal del audio.

Calidad tímbrica: Detección de tensión respiración o temblores en la voz fundamental para emociones como miedo o ira.

Todo esto definido en un módulo central de esta manera en la configuración.

Figura 27

Definición de la constante para el reconocimiento de emociones

```
AUDIO_EMOTION_MODEL = "superb/wav2vec2-base-superb-er"  # Emociones en audio
```

Nota. En esta constante se utiliza específicamente el modelo superb/wav2vec2-base-superb-er para la tarea de reconocimiento de emociones en audio (Emotion Recognition).

Para el correcto funcionamiento del modelo es fundamental una estricta estandarización de la señal de entrada. En la Tabla podremos ver los parámetros de

procesamientos que se aplican a cada uno de los segmentos de audio antes de la inferencia del audio, esto garantiza la compatibilidad con la arquitectura ya pre-entrenada.

Tabla 9

Tabla de valores en los parámetros

Parámetro	Valor configurado	Justificación técnica
Frecuencia de muestreo	16.00hz(16khz)	Estándar requerido para preservar el ancho de banda local
Duración mínima	0.1s(100ms)	Duración mínima requerida del audio para capturar las micro expresiones prosódicas
Formado de codificación	WAV PCM16	Modulación por impulsos codificados(16-bit) sin comprensión con perdida
Normalización	Rango[-1,1]	Estandarización de amplitud para estabilizar los gradientes durante la inferencia

Nota. El incumplimiento de la tasa de muestreo de 16khz perjudica severamente la precisión del modelo ya que altera el tono y no se detectan correctamente el cambio de frecuencias que se perciben por la red neuronal.

3.7. Algoritmo de segmentación e inferencia

El flujo de procesamiento funciona mediante una estrategia que se la conoce como ventaneo temporal. El audio continuo se divide en segmentos discretos y definidos por los índices start y el end.

Figura 28

Proceso de extracción de segmentos de audio

```
# Extracción y análisis de segmento de audio
start_idx = int(start * sr)  # Índice inicial
end_idx = int(end * sr)      # Índice final
seg_chunk = audio_data[start_idx:end_idx]  # Extracción
```

Nota. Proceso de extracción de segmentos de audio. Se muestra el cálculo de índices mediante el producto del tiempo por la tasa de muestreo (sr), permitiendo el recorte preciso (slicing) de fragmentos de la señal original para su análisis individual.

En esta sección se puede ver cómo se produce la extracción del audio y el análisis de segmento.

Figura 29

Flujo de escritura temporal y análisis final

```
# Escritura temporal y análisis
seg_audio_path = write_wav_from_array(seg_chunk, sr)
result_audio = analyze_audio_emotion(seg_audio_path)
```

Nota. Elaboración propia de los autores.

En conclusión, la integración del modelo Wav2vec 2 nos permite ir más allá de solo centrarnos en el contenido semántico que se tiene, nos ayuda a capturar las dimensiones paralingüísticas por así decirlo nos ayuda a descifrar no solo el "¿que se dice?", sino también el "¿Como se dice?". Esto se consigue con el análisis de características acústicas que pueden resultar sutiles, pero es fundamental en el lenguaje y detectar estados emocionales más ambiguos, esto nos proporciona una capa de validación sensorial complementaria.

3.7.1. Arquitectura de fusión Multimodal

Para poder integrar las predicciones que viene de todos los modelos anteriormente mencionados ya que tenemos texto y audio se implementó una estrategia de fusión tardía o late fusión o fusión a nivel de decisión. A diferencia de la función temprana este enfoque nos permite procesar cada módulo con la arquitectura más idónea para cada caso dependiendo de su naturaleza y combinar sus probabilidades al final.

Figura 30

Modelo de fusión multimodal ponderada

$$P_f\,(e) = (W_{txt}\,.\,P_{txt}\,(\,\mathrm{e})) + (w_{aud}\,.\,P_{aud}\,(e))$$

Nota. Elaboración propia de los autores.

Promedio ponderado: Es la configuración predeterminada que se tiene en el sistema. Se basa en la premisa que el contando semántico o de texto suele ser más discriminativo para la parte de la categorización de la emoción especifica, mientras el audio aporta más intensidad y el matiz para esto se calcula mediante una ecuación lineal. Donde

los hiperparametros se definieron empíricamente como:

- w_{txt} = 0.6$ (Peso semántico): Se le otorga mayor relevancia al texto por la precisión del modelo RoBERTa
- w_{aud} = 0.4$ (Peso prosódico): Actúa como un modulador del resultado textual

Selección por máxima confianza: En este algoritmo se opera mediante un esquema de competencia competitiva o winers takes all. El sistema evalúa los vectores de probabilidad completos y lo que hace es seleccionar la modalidad que presente la mayor certeza y descartando la otra. Esta es ideal en escenarios donde una de las fuentes presenta mucho ruido como puede ser la estática o un ruido excesivo.

Figura 31

Ecuación de fusión tardía ponderada

$$P_f\,(e) = (W_{txt}\,.P_{txt}\,(\,\mathrm{e})) + (w_{aud}\,.P_{aud}\,(e))$$

Nota. Elaboración propia de los autores.

Consenso con bonificación: Implementa un mecanismo de refuerzo positivo. Un ejemplo es si la emoción dominante o (top_emotion) coincide en ambos módulos lo que se hace es se aplica un bonus de convergencia delta = 0.2 al promedio simple, premiando ante todo la coherencia intermodal.

Figura 32

Variante de la fusión tardía que incluye refuerzo de consenso entre los modelos

$$P_f\ = \frac{P_{txt}\ \ P_{aud}}{2} + \ \delta\ si\ top\ (txt) = top\ (aud)$$

Nota. Elaboración propia de los autores.

Justificación de la convergencia multimodal: La estrategia C se basó en el fundamento en el principio de reducción de entropía. Por qué cuando se tienen dos arquitecturas independientes una basada en patrones semánticos como lo es Transformer y otra en patrones acústicos como lo es Wav2Vec 2.0, cuando estas dos coinciden en su

predicción la probabilidad del error conjunto disminuye significativamente y tiene la ventaja de modelar matemáticamente esta certidumbre reforzada lo cual lo hace actuar como un mecanismo de validación cruzada que simula en aproximación a el proceso cognitivo humano de confirmación sensorial.

Tabla 10

Estrategias de fusión

Estrategia de fusión	Confianza media	Estabilidad del sistema	Escenario de aplicación recomendado
Promedio ponderado	0.78	Alta	Uso general (balanceado)
Máxima confianza	0.82	Media	Entornos con calidad de señal semántica
Votación	0.85	Muy Alta	Casos críticos que requiere alta precisión

Nota. La estrategia de votación ofrece mayor confianza sin embargo puede resultar en indecisión si las modalidades discrepan severamente. El modelo que ofrece mayor estabilidad es el balanceado

3.7.2. Calibración de sensibilidad

Este fue un desafía sistémico a la hora de la detección de emociones ya que existe un desbalance de clases, teniendo en cuenta las pruebas donde hubo una tendencia de categorizar a la mayoría de los audios con emociones neutras o con emociones nulas para poder solucionar esto lo que se hizo fue implementar un algoritmo post procesamiento de señales que actúa como un filtro de pasa altos emocionales

Figura 33

Evaluación de la transcripción emocional

Nota. Elaboración propia de los autores.

Como se puede ver en la imagen en la versión 2.0 se tuvo problemas de generalización ya que mayormente identificaba emociones neutras cuando no era de esa manera.

Al aplicar una transformación no lineal a las probabilidades de salida definida por los parámetros que se muestra a continuación.

Tabla 11

Parámetros de experimentación

Parámetro	Valor configurado	Función de transferencia
Boost_factor	1.8(180%)	Factor de ganancia con emociones activas
Neutral_damp	0.15(15%)	Factor de atenuación de las clases pasivas como neutral o otros
MIN_THRESHOLD	0.05	Umbral de corte para evitar amplificar el ruido de fondo

Nota. Elaboración propia de los autores.

La lógica del algoritmo es que el sistema penaliza artificialmente la probabilidad de la clase neutra y redistribuye esa masa de probabilidad amplificando emociones activas.

Esto nos permite que micro expresiones emocionales que suelen tener scores bajos frente a la neutral que es más dominante, esto nos ayuda a identificar esas micro expresiones.

Figura 34

Ajustes para la supresión neutral en detección de emociones

```
#supresion neutral (AJUSTADA - mucho menos agresiva para periodistas)
NEUTRAL_SUPPRESSION_FACTOR = 0.85
ACTIVE_EMOTION_BOOST = 1.3  # Aumentado para potenciar emociones activas
MIN_EMOTION_TO_PROMOTE = 0.20  # Reducido para detectar emociones más sutiles
```

Nota. En la imagen se puede ver cómo se potencia ciertas emociones e intenta suprimir el exceso de la predominancia de la emoción neutral para que sea más eficiente la detección de emociones sutiles.

En conclusión, el diseño del módulo de fusión multimodal mediante estrategias de fusión tardía nos ha permitido que el modelo actual tenga una sensibilidad más alta ante los audios que se le presenta y el combinar ambas técnicas como lo es el análisis acústico y el

análisis semántico nos consigue una sinergia que supera las limitaciones de los sistemas multimodales.

3.7.3. Análisis de coherencia y suavizado de la señal

Dado que las emociones humanas son fenómenos continuos que poseen inercia y no cambian al momento o en milisegundo, lo que se implemento fue otro modulo post-procesamiento temporal. El objetivo de este módulo es mitigar la volatilidad estocástica ruido que puede surgir en predicciones aisladas en segmentos consecutivos.

Figura 35

Formula de suavizado exponencial simple

$$S_t = \alpha \, . x_t + (1 - \alpha) \, . S_t - 1$$

Nota. Elaboración propia de los autores.

Lo que se hace en este algoritmo es usar un Suavizado exponencial, este método pondera la predicción actual frente al historial acumulado, lo que nos garantiza transiciones suaves en la curva emocional.

La ecuación se rigüe bajo este comportamiento:

St: al valor del suavizado actual.

Xt: Valor crudo predicho por la fusión multimodal en el instante t.

St-1: Valor de suavizado por segmento anterior o historia.

Alpha: coeficiente de suavizado (0< Alpha <1)

La configuración de los hiperparametros del estabilizador se detalla en la siguiente tabla.

Tabla 12

Parámetros de configuración para la experimentación

Parámetro de configuración	Valor	Justificación técnica
Estado del modulo	Archivo	Permite que se genere la correlación entre segmentos
Ventana de historial	N=3	Numero de segmentos previo consideradas
Factor Alpha	0.6	Balance responsivo

Nota. Elaboración propia de los autores.

En conclusión, la implementación del algoritmo de suavizado temporal es indispensable debido a que nos ayuda a garantizar la estabilidad de señal de salida. Esto al mitigar la volatilidad estocástica inherente a las predicciones cuadro por cuadro, y el sistema logra representar la evolución emocional como un continuo coherente en lugar de eventos discretos caóticos.

3.7.4. Reducción dimensional y taxonomía

Si bien la base del modelo funciona con la taxonomía de 7 dimensiones basados en el modelo de Ekman + neutro para poder tener una precisión granular interna, la presentación del resultado final requiere un nivel abstracción mayor todo esto con el fin de facilitar la toma de decisiones del usuario.

Por lo cual se implementó una capa de agregación de categorías, esto reduciendo el espacio de salida a 4 macro o 4 estados emocionales. Este mapeo agrupa emociones basándose en su valencia ya sea positiva o negativa y a nivel de activación Arousal como se describe en la siguiente tabla.

Tabla 13

Categorías y criterios para la detección de emociones

Macro-categoria	Clases constituyentes	Criterio de agrupación
Feliz(positivo)	Alegría, sorpresa	Valencia positiva/Alta energía
Enojado(negativo)	Ira, disgusto	Valencia negativa/Alta energía
Triste (negativo pasivo)	Tristeza, miedo	Valencia negativa/Baja energía
Neutral(base)	Neutral, otros	Ausencia de carga afectiva o incertidumbre

Nota. Elaboración propia de los autores.

Concluimos que la estrategia de reducir la direccionalidad aplicada a la taxonomía de salida responde más que todo a criterio de usabilidad y una búsqueda de la eficiencia operativa. La abstracción de 7 dimensiones complejas a 4 categorías accionables nos ha permitido reducir la carga cognitiva requerida para la interpretación de los datos.

3.8. Arquitectura del pipeline

3.8.1. Introducción

El núcleo del sistema articula mediante un procesamiento secuencial de datos diseñado bajo una arquitectura modular monolítica. Este flujo de trabajo dirige la interacción entre las 7 capas criticas transformando la señal de audio cruda en información emocional estructurada, todo esto mediante la extracción de características multimodales texto+ audio y su posterior fusión inteligente.

3.8.2. Diseño arquitectónico

Modular monolítico

La definición de esta arquitectura es que toda la aplicación ejecuta una única unidad de despliegue la cual está en app_fastapi.py, al ser un servidor unificado gestiona todas las peticiones, con un estado compartido modelos de ML en memoria compartida _models_cache, _voice_encoder, y la comunicación interna mediante llamadas funciones locales sin necesidad de red y sin serialización.

Modularidad:

- Core/models.py: Gestión de modelos ML
- Core/audio_processing.py: Preprocesamiento de señal
- Core/transcirpcion.py/transcripcion_cloud.py: conversión de voz a texto
- Core/translation.py: traducción de español a ingles
- Core/emotion_analisis.py: análisis emocional multimodal
- Core/diarizacion.py: detección de hablantes
- Core/export_manager.py: serialización de resultados

Se escogió esta arquitectura ya que también es fácil de integrar nuevas funcionalidades o modificar cualquier modulo buscando mejora de manera independiente sin afectar la funcionalidad general del sistema. También se agregó para el uso de un modelo en la nube todo con el fin de buscar la mejor precisión y comodidad del usuario al

solo ingresar su api de Whisper en caso también de que los requerimientos del sistema no sean cumplidos por parte del usuario.

3.8.3. Ventajas y Desventajas de la Arquitectura

Tabla 14

Tabla de ventajas de la arquitectura

Aspecto	Ventaja	Justificación Técnica
Rendimiento	Latencia baja	Sin overhead de red ni serialización entre servicios
Gestión de Memoria	Compartición de modelo ML	Un solo Whisper (5GB VRAM) compartido por todas las peticiones
Simplicidad Operacional	Despliegue trivial	Un solo contenedor Docker, no orquestación de microservicios
Transiciones	Procesamiento atómico	Todo el pipeline en un único thread/proceso → no necesidad de sagas distribuidas
Debugging	Trazabilidad completa	Stacks traces directos, logs centralizados
Desarrollo	Menor fricción	No necesita de protocolos gRPC/REST internos

Nota. Elaboración propia de los autores.

Tabla 15

Desventajas identificadas y mitigaciones

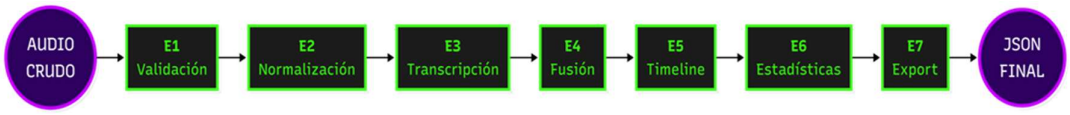
Desventaja Típica	Mitigación en el Diseño
Escalabilidad horizontal limitada	ThreadPoolExecutor + Lazy loading + Circuit breakers permiten el manejo concurrente eficiente
Acoplamiento tecnológico	Módulos desacoplados permiten migración gradual (ej: cambiar Whisper por otro modelo)
Fallo total del sistema	Resilience patterns: Circuit breaker, Retry con backoff, Graceful degradation
Tamaño del artefacto	Docker multi-stage builds + Cache de capas

Nota. Elaboración propia de los autores.

3.8.4. Etapas críticas del pipeline

Figura 36

Las 7 etapas críticas del pipeline



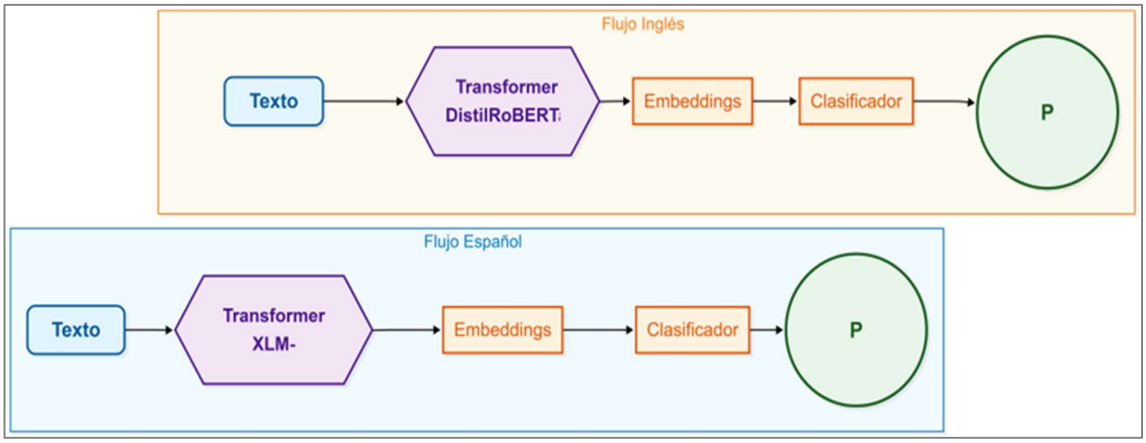
Nota. Elaboración propia de los autores.

3.8.5. Extracción de características multimodal

Primera Modalidad – Texto: En esta sección se extrae el contexto del sentimiento léxico como palabras positivas o negativas, contexto semántico relaciones entre palabras, intensidad de las emociones como adjetivos o adverbios.

Figura 37

Modalidad 1: Texto

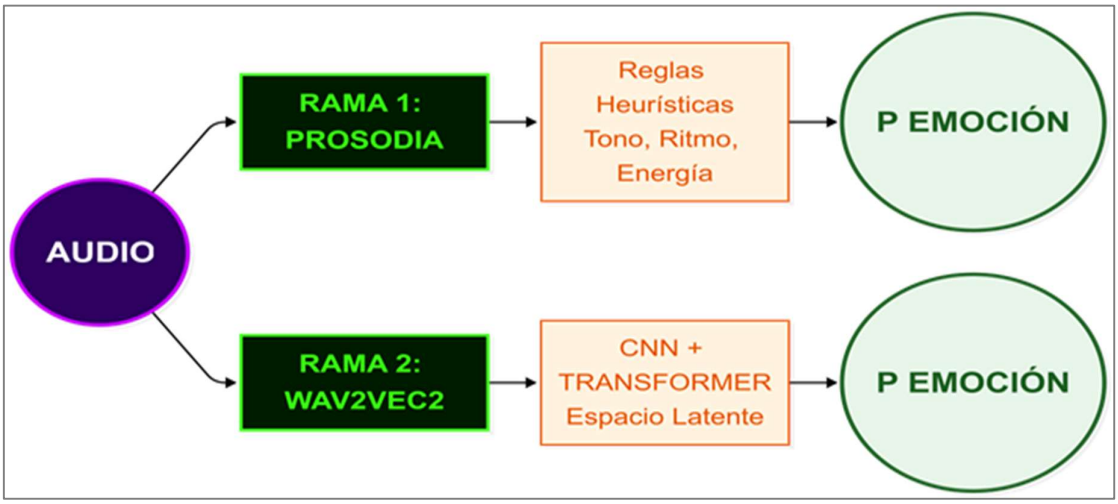


Nota. Elaboración propia de los autores.

Segunda Modalidad – Audio: Se registran diferentes emociones en base a reglas prosódicas con la siguiente tabla explicaremos mejor cada una de las características.

Figura 38

Modalidad 2: Audio (Prosodia+Acustica)



Nota. Rama 1 - Características prosódicas

Tabla 16

Identificación de emociones

Emoción	Pitch	Energía	Rate	Pitch Variabilidad
Feliz	Alto	Alta	Rápido	Alta
Enojado	Alta	Muy Alta	Rápido	Media
Triste	Bajo	Bajo	Lento	Bajo
Neutral	Media	Media	Media	Bajo

Nota: Rama 2 - Wav2Vec2

En esta sección se usa el modelo transformer para modelado de secuencias temporales, se hace la clasificación de emociones mediante el tono y se usan CNN para la extracción de características acústicas.

Figura 39

Fusión multimodal

```
# Pesos para fusión de emociones (texto vs audio)
# Valores más altos = más importancia a esa modalidad
EMOTION_WEIGHT_TEXT = float(os.getenv("EMOTION_WEIGHT_TEXT", "0.6"))
EMOTION_WEIGHT_AUDIO = float(os.getenv("EMOTION_WEIGHT_AUDIO", "0.4"))
```

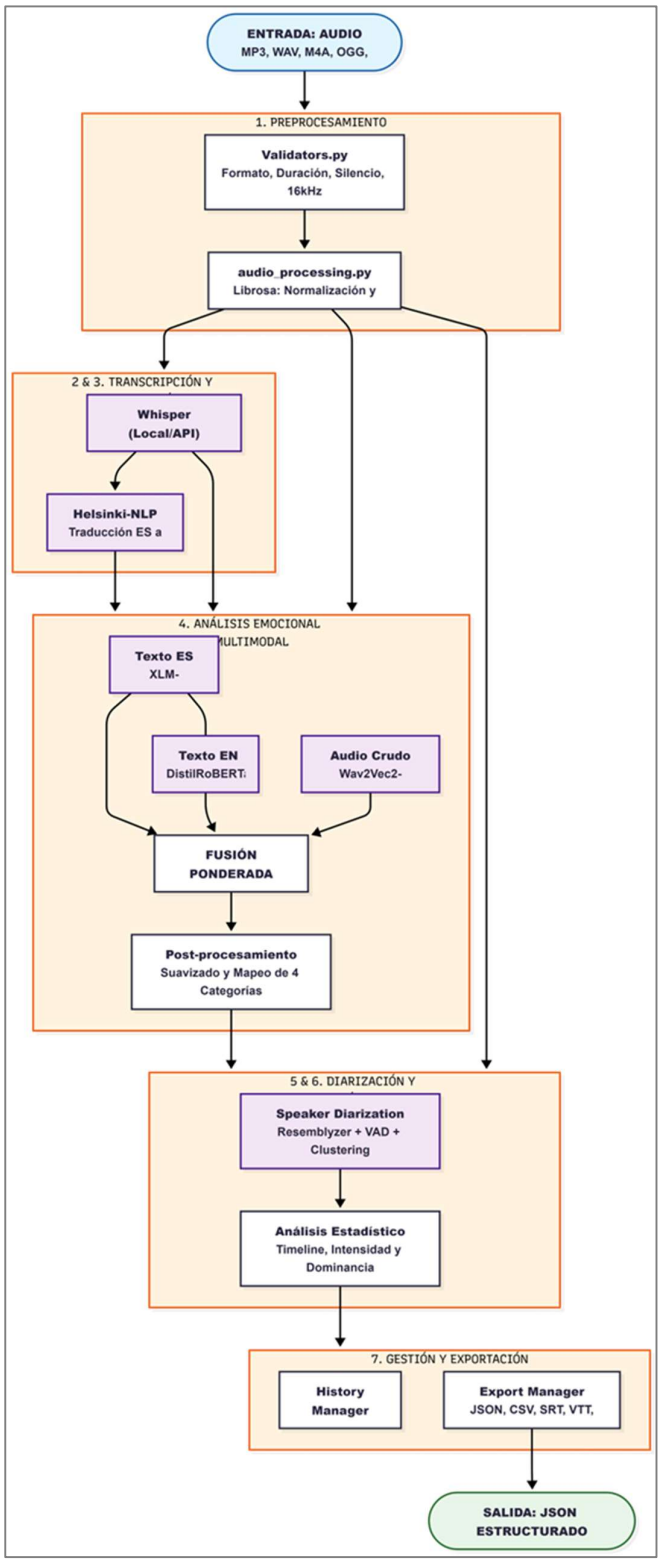
Nota. Elaboración propia de los autores

Como se puede ver en este fragmento del código se puede ver la fusión multimodal en el cual se puede ajustar los pesos lo cual le da más importancia ya sea al texto o al audio en este caso se ve una distribución de 60/40.

Mapeo del modelo:

Figura 40

Estructura grafica del modelo



Nota. Elaboración propia de los autores

3.8.6. Fase de Ingesta y Procesamiento de la Señal

El punto de entrada es un input el cual recibe el archivo de audio y lo procesa por una serie de rutinas de validación y normalización. Esto nos ayuda a garantizar la compatibilidad de los modelos de Deep Learning subsiguientes:

- Validación: Verifica la duración del archivo y que se encuentre en rango.
- Re-muestreo: La señal es convertida a una tasa de muestreo de 16khz
- Normalización: Se ajusta la amplitud de onda al rango [-1,1], para estandarización de volumen y facilitar la convergencia

Transcripción automática

Una vez que el audio es preprocesado, el audio procede a ingresar al módulo de Whisper en esta etapa cumple una doble función:

- Generación de texto: Convierte el contenido fonético en cadena de texto
- Segmentación temporal: genera timestaps precisos para cada fase, esto nos servirá como un índice maestro para la sincronización del analiza de texto y audio en las fases siguientes.

Ramificación de análisis textual (NLP)

Para que la precisión semántica se la máxima posible el flujo bifurca en dos estrategias:

Ruta A: Análisis directo en español utilizando el modelo XLM-RoBERTa, esta captura matices culturales y la jerga local.

Ruta B (Traducida): Se procesa por la traducción del modelo neuronal mediante Helsinki-NLP, esto seguido de un análisis en ingles por DistliRoBERTa. Esta ruta en lo que resulta en el aprovechamiento de la robustez de los modelos entrenado.

Fusión bilingüe: Amabas probabilidades se unifican mediante un promedio matemático el cual ya se lo reviso anteriormente.

Extracción de características acústicas

Simultáneamente el sistema activa el módulo de reconocimiento de emociones del habla.

Utilizando la arquitectura Wav2Vec2.0, se analiza la señal de audio que corresponde a cada segmento para poder extraer las características paralingüísticas todo esto independiente del contenido semántico que se tiene.

Fusión Multimodal

Esta etapa es crítica, en esta etapa convergen los vectores de probabilidad del texto y audio.

- Ponderación: Se aplica la fusión tardía con una distribución de pesos predefinida de 60/40
- Ajuste de sensibilidad: Se ejecutan algoritmos heurísticos para potenciar las emociones activas y buscar suprimir la predominancia de la clase neutral o nulo este paso nos ayuda a mejora la precisión del modelo por el reconocimiento de micro sentimientos.

Post procesamiento temporal

Antes de la parte final los datos pasan por un filtro de suavizado temporal este módulo analiza la coherencia entre segmento actual y el historial de los últimos 3 segmento, lo cual nos ayuda a mitigar cualquier cambio brusco que puede existir en el modelo.

Serialización de salida

Al final el sistema consolida toda la información en un archivo JSON el cual ya viene estandarizado que incluye:

- Transcripción completa.
- El desglose de las emociones por segmento.
- Las métricas de distribución global.
- Datos para visualizar en la línea de tiempo.

Conclusión: La arquitectura modular monolítica del sistema representa una solución pragmática la cual equilibra rendimiento con una buena latencia mediante un procesamiento local sin overhead de red, simplicidad un solo contenedor docker puede generar el

despliegue, mantenibilidad módulos bien separados facilitan la evolución y corrección de componentes del código. Y por último la precisión en la fusión multimodal.

3.9. Interfaz (Frontend)

En la parte de la interfaz se implementó bajo una arquitectura lo más simple posible una arquitectura SPA o single page aplication ligera.

El diseño de la interfaz responde a los principios de interacción humano computadora o HCI priorizando ante todo la carga cognitiva mediante una jerarquía visual clara y una retroalimentación inmediata del sistema.

A diferencia de las soluciones basada en frameworks pesados, se optó por una implementación nativa para maximiza el rendimiento en el navegador y tener la menor latencia posible en la visualización de los datos.

Tabla 17

Stack tecnológico y dependencias

Tecnología	Versión/Implementación	Justificación
HTML5	Estándar W3C	Estructura semántica y SEO técnico
CSS3	CSS variables	Gestión de temas centralizada sin procesamiento
JavaScrip	ES6	Ejecución de lógica asíncrona y manipulación del DOM sin dependencia externa
Chart.js	V4.x	Motor de renderizado de gráficos basado en Canvas
Tipografía	Inter	Fuentes sans-serif optimizada para interfases

Nota. Elaboración propia de los autores.

Arquitectura de componentes visuales:

La interfaz se estructura mediante un patrón de diseño modular, esto dividiendo la vista en contenedores funcionales e independientes, esta separación permite ordenar y gestionar el estado de la aplicación de una manera aislada para cada sección:

- Panel de control.
- Visualización de métricas.
- Línea de tiempo interactiva.

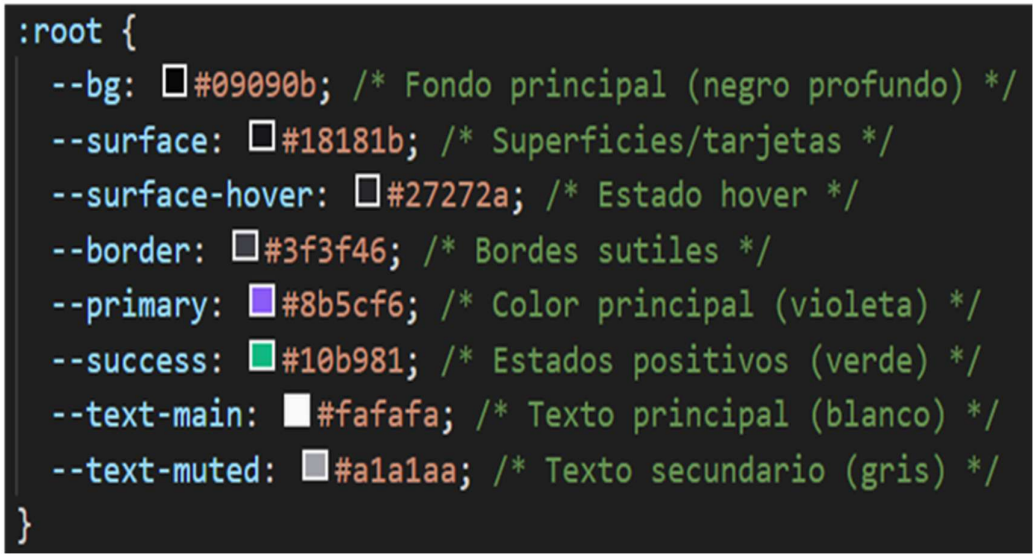
3.9.1. Sistema de Diseño y Semiótica del Color

Para garantizar la consistencia visual y el poder mantener el código se implementó un sistema de diseño basado en css custom properties.

Esto nos permite la modificación global del tema mediante cambios a las variables raíz, esto facilita la adopción de diferentes contextos de iluminación.

Figura 41

Diseño CSS Custom Properties



Nota. Elaboración propia de los autores.

Además, también se definió una codificación semiótica del color que representa las diferentes categorías de emociones que existen.

Tabla 18

Codificación semiótica de color para representar emociones

Categoría emocional	Valor hexadecimal	Semiótica asociada
Feliz	Verde	Éxito, bienestar
Enojado	Rojo intenso	Peligro, alta intensidad
Triste	Azul	Calma, frialdad
Neutro	Violeta	Equilibrio

Nota. Elaboración propia de los autores.

3.9.2. Lógica de Interacción y Comunicación Asíncrona

El núcleo funcional del frontend está en la gestión de eventos y la comunicación con la API, se implementaron patrones de programación asíncrona(async/await) para evitar el bloqueo del hilo principal de ejecución durante el procesamiento de los archivos pasados.

Figura 42

Mecanismo de monitoreo del estado del servidor

```
async function checkAPI() {
    const el = document.getElementById('apiStatus');
    try {
        await fetch(API_URL + "/health");
        el.textContent = "Sistema Online";
        el.classList.add('online');
    } catch {
        el.textContent = "Desconectado (Esperando servidor...)";
        el.classList.remove('online');
    }
}
```

Nota. Elaboración propia de los autores.

Monitoreo de disponibilidad se implementa un mecanismo de sondeo periódico para verificar la latencia y la disponibilidad del servidor.

Orquestación de petición y de análisis con una construcción dinámica de objetos FormData para la transmisión eficiente de datos de audio y los metadatos de la configuración hacia el endpoint.

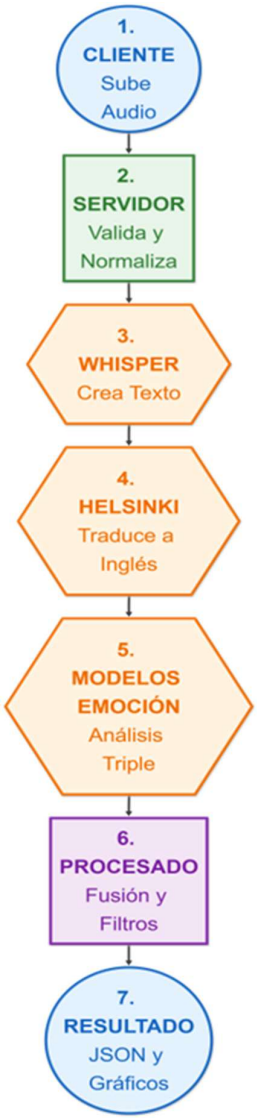
Renderización de datos se genera un mapeo directo entre la respuesta estructurada del JSON y los componentes de DOM, esto actualizando instancias de chart.js sin necesidad de que se recargue la página.

3.9.3. Flujo de Datos Cliente Servidor

El diagrama lógico describe el ciclo de vida de una solicitud de análisis estándar como se puede ver en el siguiente esquema:

Figura 43

Diagrama del ciclo de vida del Analisis



Nota. Elaboración propia de los autores.

Tabla 19

Tabla de Atributos JSON

JSON	Componente visual	Propósito de representación
Global_emotions	Tarjetas métricas (KPIs)	Resume de emoción predominante
Emotion_timeline	Gráfico lineal Multi-serie	Análisis de volatilidad emocional
Segmentos	Lista de picos de intensidad	Identificación de eventos críticos

Nota. Elaboración propia de los autores.

3.9.4. Vista general y exportación

El sistema también integra dos capacidades transversales al módulo de la interfaz que elevan la utilidad operativa sobre todo en los entornos de contact center los cuales consiste en una tabla maestra con el historial de audios o de llamadas procesadas, adicional también se incluyó una exportación multiformato con generación de reportes pdf enriquecidos por un LLM de manera local.

3.9.4.1. Tabla de historial de llamadas. El panel de scoring presenta una tabla que muestra una consolidación de los audios analizados y procesados por los módulos, cada fila corresponde a una llamada o audio y expone de forma directa los atributos más relevantes sobre 100 y también con filtros de emoción para poder detectar problemas de manera más eficiente.

3.9.4.2. Exportación multi-formato. El sistema soporta 6 formatos de exportación, los cuales son seleccionables desde la barra de herramientas de resultados. Cada formato corresponde a un caso en específico en la siguiente tabla explicamos un poco este punto.

Tabla 20

Tabla de formatos de exportación

Formato	Contenido exportado	Casco de uso principal	Modulo backend
JSON	Datos estructurados completos (segmentos, emociones, KPIs, scoring)	Integración con sistemas externos y reprocesamiento	ExportManager.export_json()
CSV	Tabla plana de segmentos con inicio, fin, hablante, emoción, texto ES/EN	Análisis estadístico en hojas de calculo	ExportManager.export_csv()
SRT	Subtítulos temporalizados con etiqueta de hablante y emoción	Incrustación de subtítulos en video	ExportManager.export_srt()
VTT	Subtítulos WebVTT con metadatos de cabecera	Reproductores web HTML5	ExportManager.export_vtt()
TXT	Transcripción legible con metadatos de cabecera	Revisión humana rápida y archivo documental	ExportManager.export_txt()

Excel	Libro .xlsx con hoja de segmentos y hoja de resumen KPI	Reportes gerenciales y supervisión de calidad	Endpoint/export/excel

Nota. Elaboración propia de los autores.

3.9.5. Integración de LLM local para interpretación de datos

En el sistema se menciona también la exportación de pdf sin embargo esto es un reporte el cual es generado con un LLM esto con el fin de integrar la función de reportes de control de calidad del contact center haciendo más visual un panorama general de cómo se maneja el contact center y en que llamadas puntuales hay problemas.

3.9.5.1. Justificación de la ejecución local. La decisión de ejecutar el LLM en local responde a los siguientes 3 criterios operativos identificados durante la decisión del diseño de sistema. En la tabla contrasta las características del enfoque del uso del APIs en la nube para el caso en específico de la interpretación de métricas de call center.

Tabla 21

Tabla de características del enfoque del uso del APIs

Criterio	LLM local (Ollama)	API en la nube	Impacto en el sistema
Privacidad de datos	Total: los datos nunca salen del servidor local	Parcial: datos enviados a servidores del proveedor	Requerimiento critico en callcenters (LOPD/GDPR)
Costo operativo	Cero: inferencia local sin cuota por token	Variable: facturación por token consumido	Sostenibilidad a largo plazo sin límite de uso
Latencia	Red local (<1ms overhead de transporte)	Latencia de red + cola del proveedor (50-300 ms)	Inferencia más predecible en entornos LAN
Disponibilidad offline	Funciona sin conexión a Internet	Requiere conectividad activa	Entornos corporativos con restricciones de red
Calidad del Modelo	Qwen2.5:7B – suficiente para análisis de KPIs	GPT-4o / Claude 3.5 – mayor capacidad general	Tarea acotada: Interpretación de métricas estructuradas

Nota. Elaboración propia de los autores.

3.9.5.2. Diseño y promt para el análisis de contact center. La función generate_data_interpretation() implementa un patrón de degradación elegante (graceful_degradation): si Ollama no se encuentra en ejecución, si el modelo Qwen2.5 no está instalado de manera local o si el tiempo de respuesta es mayor a 60 segundos configurado como timeout, la función retorna None sin lanzar excepciones. El módulo es el orquestador que interpreta este valor y genera el reporte PDF sin la selección interpretativa preservando intactas las métricas cuantitativas.

Este diseño garantiza que la experiencia del usuario final permanece interrumpida independientemente del estado del servicio LLM, Los supervisores que no dispongan de los recursos de hardware necesarios para ejecutar Ollama obtiene reportes PDF completos en métricas; quienes si dispongan del servicio reciben adicionalmente un análisis interpretativo en lenguaje natural que contextualiza los datos cuantitativos y propone acciones concretas de mejora, lo que eleva el valor analítico del sistema sin que se comprometa su resiliencia operativa.

Tabla 22

Tabla de elementos del prompt

Elemento del prompt	Contenido / Valor
Rol del Sistema	Analista experto en calidad de servicio al cliente y callcenters
Bloque de datos	Resumen textual de métricas: total audios, score promedio, desviación estándar, distribución de clasificaciones, promedios por dimensión con porcentaje de cumplimiento, distribución emocional y top de problemas frecuentes
Instrucción de salida	Diagnostico general – Análisis por dimensión – Patrones Identificados – Áreas críticas – Recomendaciones estratégicas (opcional)
Restricción de formato	Español profesional – Máximo 500 palabras – Sin headers Markdown – Viñetas de texto plano
Parámetros de inferencia	Temperature: 0.3 – top_p: 0.9 – num_predict: 1024

Nota. Elaboración propia de los autores.

3.9.6. Ergonomía y experiencia de usuario (UX)

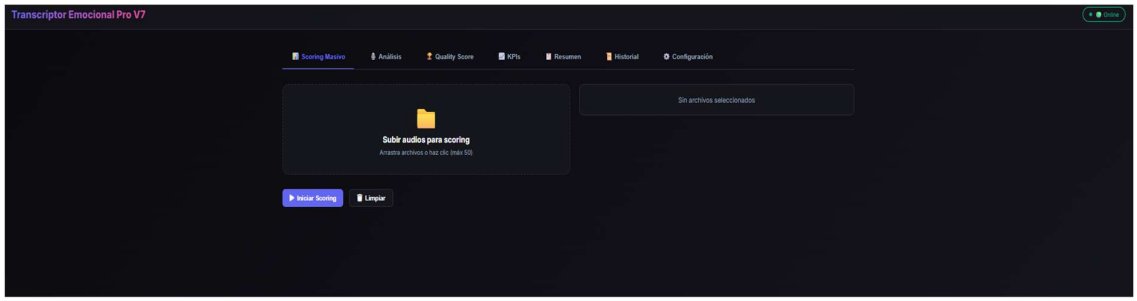
El diseño de la interfaz obedece a criterios de usabilidad heurística destacada por su mitigación de la fatiga visual implementando un dark mode de alto contraste ideal para una operación continua, fedback de estado usando indicadores visuales para informar al usuario

el estado de los procesos de larga duración. Y un diseño adaptativo con el uso de CSS grid layout para asegurar una correcta visualización del dashboard en dispositivos de escritorios y tablets para evitar la pérdida de información.

Vista del Front-End

Figura 44

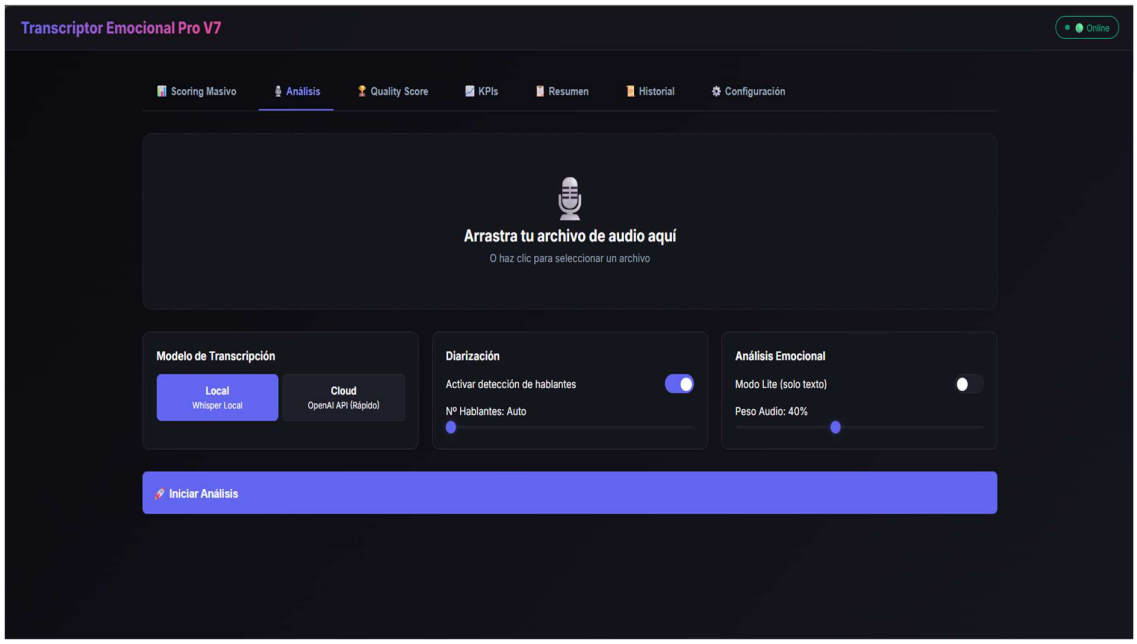
Interfaz del usuario para el procesamiento masivo



Nota. Elaboración propia de los autores.

Figura 45

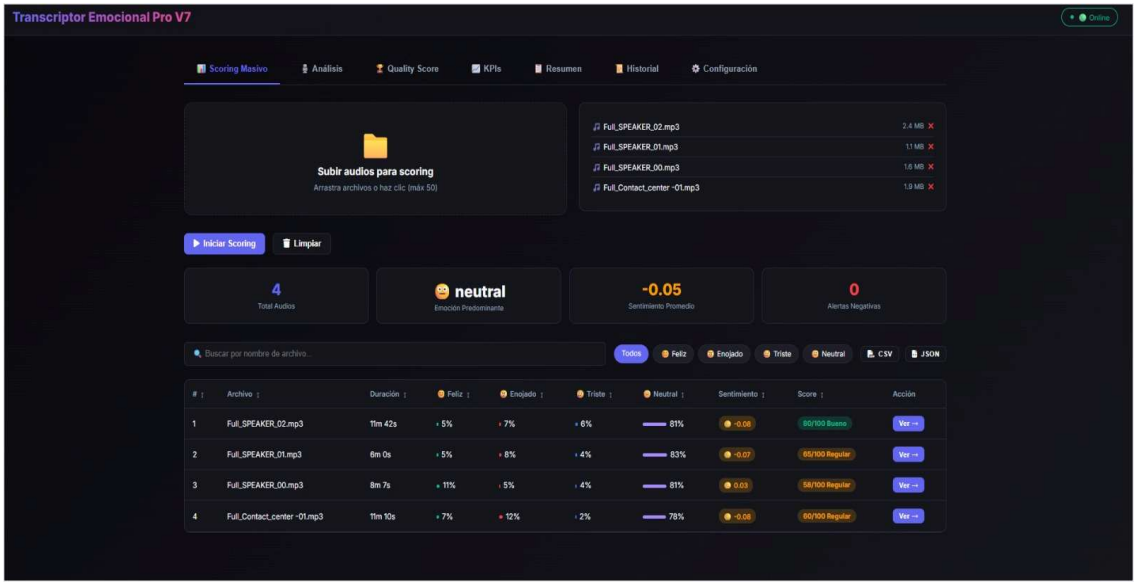
Interfaz del audio individual



Nota. Elaboración propia de los autores.

Figura 46

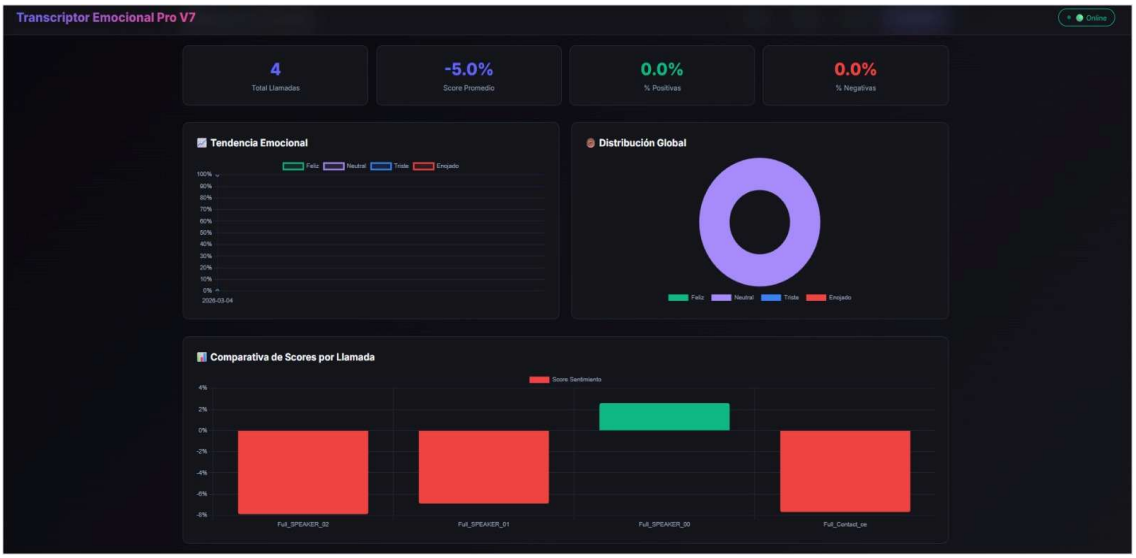
Resultados y graficas del análisis de audio



Nota. Elaboración propia de los autores.

Figura 47

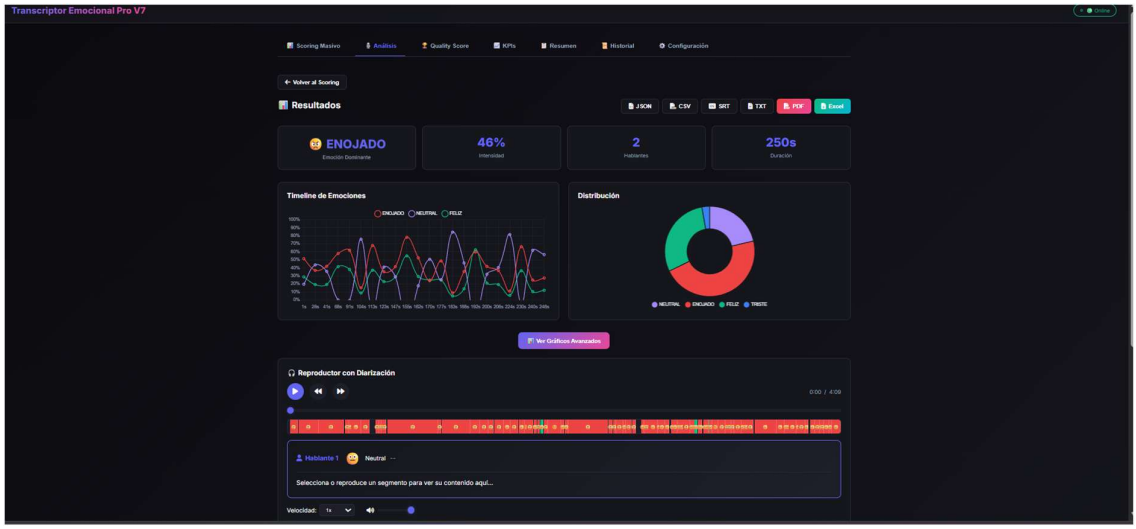
Gráficos detallados de los KPIs



Nota. Elaboración propia de los autores.

Figura 48

Reproductor de audio, transcripción reconocimiento de voz



Nota. Elaboración propia de los autores.

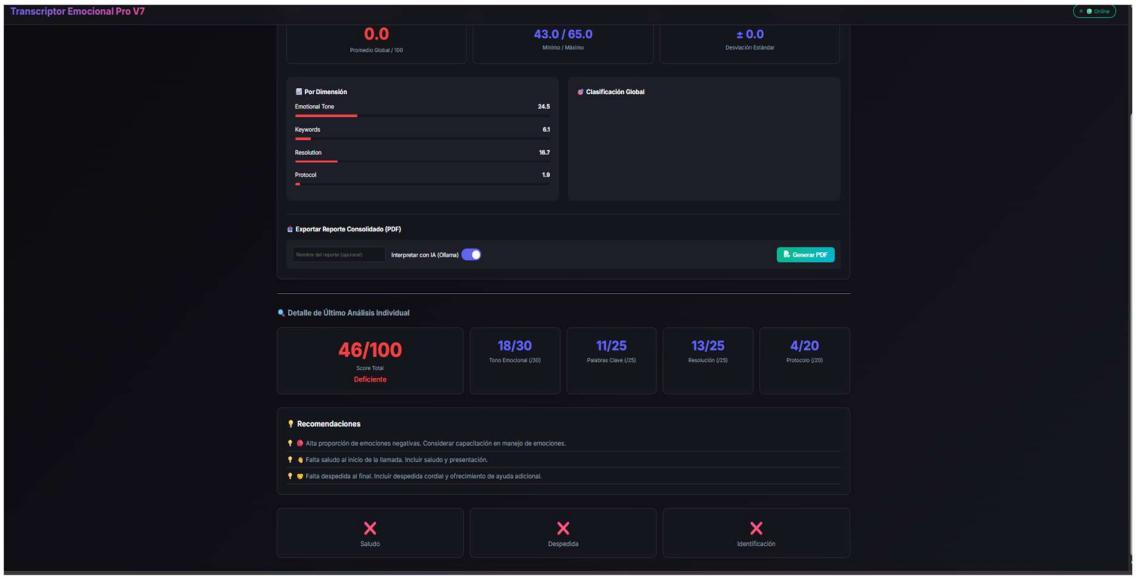
Figura 49

Tablas de transcripciones

Nota. Elaboración propia de los autores.

Figura 50

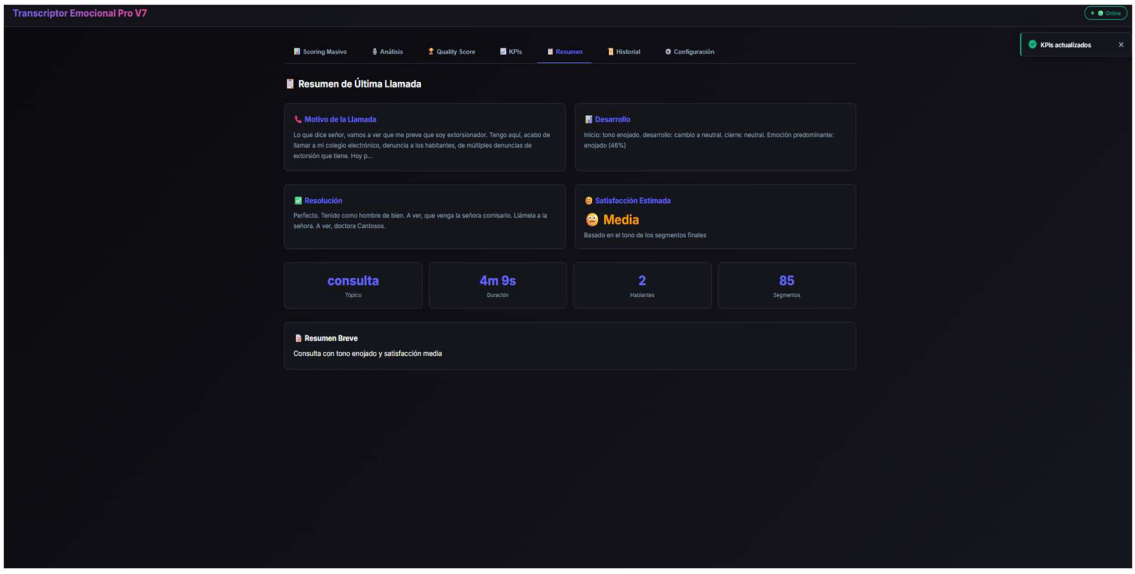
Quality score



Nota. Elaboración propia de los autores.

Figura 51

Resumen del audio de manera individual



Nota. Elaboración propia de los autores.

Lo que se puede ver en las imágenes es el resultado y la interacción del frontend y el resultado de una entrevista de un nuevo paso en la Avenida Mariana de Jesús es por eso

por lo que existe esos picos muy bien definidos y el grafico se ve así debido a que están interactuando dos voces en el mismo espacio.

Además se agregaron funcionalidades pensando en el usuario una de ellas es la diarizacion en tiempo con el audio por lo cual sería como un karaoke el cual va a ir viendo que se dice en que segundo del audio, también se agregó un historial para poder acceder a transcripciones anteriores facilitando la experiencia del usuario, la gestión de las APIS de manera intuitiva también es fundamental por lo cual se incluyó para que su gestión sea amigable para el usuario y por último gráficos los cuales son un poco más descriptivos en cuanto a la intensidad emocional que se encuentra en el modelo.

Métricas del sistema

Para garantizar la fiabilidad operativa y poder facilitar el mantenimiento proactivo, del sistema se implementó un módulo de telemetría interna este componente actúa similar a un monitor de estado, esto hace referencia a que constantemente está registrando eventos transaccionales en tiempo real y calculando estadísticas de rendimiento criticas para evaluar la calidad del servicio QoS.

La instrumentación se gestiona mediante un acumulador de estado en una memoria volátil, todo esto definido en la estructura _metrics.

Figura 52

Implementación del sistema de monitoreo o telemetría para el modelo

```
# Definición del vector de estado para telemetría
_metrics = {
    "requests_total": 0,        # Contador global de throughput (caudal)
    "requests_success": 0,      # Transacciones completadas (HTTP 200)
    "requests_failed": 0,       # Excepciones o errores de servidor (HTTP 5xx)
    "total_processing_time": 0.0 # Acumulador de latencia (segundos)
}
```

Nota. Elaboración propia de los autores.

El acceso de estos datos se expone a través del endpoint de diagnóstico GET/health/detailed. A diferencia de un health check binario el cual solo indica si el sistema

este encendido apagado, este recurso devuelve un informe detallado que incluye métricas derivadas calculadas en tiempo de ejecución.

Esta tabla describe los indicadores expuestos por el sistema y su relevancia para la auditoria técnica:

Tabla 23

Tabla de indicadores y métricas

Métricas/KPI	Tipo de Dato	Descripción técnica y Relevancia
Volumen de peticiones	Contador	Mide la carga que ha recibido el sistema
Tasa de Éxito	Porcentaje	Indicador critico de la fiabilidad
Latencia media	Tiempo	Promedio aritmético del tiempo de inferencia
Conteo de fallos	Contador	Número absoluto de translaciones fallida

Nota. Elaboración propia de los autores.

Las métricas de monitoreo internas nos ayudan a ver lo que el sistema es capaz de hacer al exponer indicadores clave como la latencia media y la tasa de éxito a través del API estandarizada, se facilita no solo la detección temprana de anomalías o cuellos de botella en el procesamiento multimodal, sino que también se provee una base cuantitativa para cargar y evaluar el rendimiento.

3.10. Human in the loop y el reentrenamiento

3.10.1. Problema

A lo largo del proyecto uno de nuestros principales problemas fue que no se tenía una gran cantidad de datos etiquetados y el proceso de etiquetado era extenso, se necesitaba una gran cantidad de datos para tener un entrenamiento eficiente con Fine Tune para que pueda detectar correctamente las emociones y se tendría que variar con diferentes tipos de pronunciaciones y diferentes modismos que se tienen en el idioma español ya que existe una gran cantidad de variación en los tonos entre regiones y ciertos localismos o eufemismos en todos los países de habla hispana

3.10.2. Enfoque

Lo que se busca con la v6 es poder involucrar al usuario en la mejora de la precisión del modelo en otros términos lo que se busca está en 3 enfoques:

Reinforcement learning from human feedback (RHLF): Este concepto tomo fuerza en modelos como GPT esto se aplica cuando el sistema una vez genera la salida el usuario puede calificar o corregir al modelo y este tipo de datos ayudan a ajustar al modelo. Este enfoque es más usado en la alineación de comportamiento cuando la tarea es subjetiva o compleja y no se puede medir con un simple correcto o incorrecto como escribir ensayos en tono amable.

Human in the loop (HITL): Este es un concepto parecido al anterior, pero interviene más en la parte del entrenamiento por parte del usuario validando y ajustando el modelo. En este enfoque la arquitectura operativa y la seguridad lo que se busca es que el feedback del usuario sea constante parte del diseño del producto, el cual es un ciclo continuo de mejora y el usuario sirve como filtro de seguridad y corrige errores del modelo y esa corrección se guarda para el reentrenamiento.

Active Learning: En este proceso en cambio lo que se busca es en caso de falta de datos el algoritmo identifica datos que son difíciles de procesar o que se tiene mayor incertidumbre o donde se tiene mayor incertidumbre. Este enfoque se usa más para la optimización de datos en caso de tener un gran dataset el cual requiere etiquetar, pero no se tiene un gran presupuesto o tiempo de revisión.

3.10.3. Enfoque adoptado para el reentrenamiento

El enfoque por el cual se optó debido a las características anteriormente mencionadas fue HITL (Human in the loop) con reentrenamiento supervisado este enfoque fue seleccionado debido a las siguientes razones:

Adaptación al dominio y personalización (Domain shift)

Al tener modelos pre entrenados como lo es wav2vec2 o parecidos suelen estar entrenados con datasets los cuales son: genéricos, no exploran otros escenarios o usan ambientes controlados. Pero con datos reales en el cual está involucrado la expresión emocional real con diferentes acentos o un entorno ruidoso este enfoque nos permite que el modelo se especialice gradualmente en el entorno el cual esta desplegado.

Figura 53

Sistema de Inferencia Multimodal y Aprendizaje Continuo

```python
def run_retraining():
    if not TRANSFORMERS_AVAILABLE:
        return

    logger.info("Empieza el reentrenamiento del modelo de audio.")

    entries = load_feedback_data(CONFIG["feedback_dir"])
    if len(entries) < 2:
        logger.warning("No hay suficiente datos para reentrenar el modelo (min 2 samples). Abortando.")
        return

    dataset = prepare_dataset(entries)

    dataset = dataset.train_test_split(test_size=0.2)
    train_ds = dataset["train"]
    test_ds = dataset["test"]

    logger.info(f"Cargando el modelo base: {CONFIG['model_name']}")
    feature_extractor = Wav2Vec2FeatureExtractor.from_pretrained(CONFIG["model_name"])
```

Nota. Elaboración propia de los autores.

Un ejemplo de esto es esta sección del código la cual es scirpts/retrain_audio.py la cual se ve como se usa datos validados por parte del usuario, también muestra que si no existen los datos suficientes para el entrenamiento no intenta entrenar por un tema de estabilidad y en la parte de dataset. train_split se separan los datos de prueba para verificar que el modelo si está aprendiendo y no solo está memorizando lo cual puede generar un overfitting a futuro.

Manejo de subjetividad emocional (Ground Truth Reliability)

A diferencia del entrenamiento para el reconocimiento de imágenes como lo es reconocer que animal es un gato siendo un hecho objetivo las emociones aun para algunas personas es subjetivo y ambiguo, por lo cual el usuario en el bucle es como un árbitro el cual dictamina el resultado final con esto se busca que el modelo reduzca las alucinaciones en las emociones basándose solo en patrones acústicos los cuales pueden variar.

Eficiencia de datos (Data Efficiency/Cold Start)

El entrenamiento de modelos de audio requiere miles de horas de datos etiquetados, los cuales pueden ser costosos y difíciles de obtener para casos de uso específicos el enfoque que se tiene en este modelo permite resolver el arranque frio o cold start el cual

comienza con el modelo base y a partir de la acumulación del dataset por parte del usuario y su corrección se puede ir mejorando el modelo.

Figura 54

Función para el almacenamiento y persistencia de predicciones

```
def save_prediction(self, predicted_label: str, confidence: float, audio_temp_path:
    Optional[str] = None, metadata: Optional[Dict[str, Any]] = None) -> str:
    prediction_id = str(uuid.uuid4())

    persistent_audio_path = None
    if audio_temp_path and os.path.exists(audio_temp_path):
        try:
            audio_dir = os.path.join(self.storage_dir, "audio")
            os.makedirs(audio_dir, exist_ok=True)

            ext = os.path.splitext(audio_temp_path)[1] or ".wav"
            filename = f"{prediction_id}{ext}"
            dest_path = os.path.join(audio_dir, filename)

            import shutil
            shutil.copy2(audio_temp_path, dest_path)
            persistent_audio_path = dest_path
        except Exception as e:
            logger.error(f"Error al guardar el audio: {e}")

    self._predictions[prediction_id] = {"audio_path": persistent_audio_path,"predicted_label": predicted_label,"confidence": confidence,
        "metadata": metadata or {},
        "timestamp": datetime.now().isoformat(),
        "validated": False
    }
    self._save_json(self.predictions_file, self._predictions)
    return prediction_id
```

Nota. Elaboración propia de los autores.

Como se puede ver en esta parte del código core/feedback_system.py es un mecanismo de ingestión para el aprendizaje activo del modelo, se ve una estructura base del ciclo de aprendizaje al no solo persistir con la predicción si no también con la métrica de confianza del modelo lo cual nos habilita estrategias para muestreo por incertidumbre lo que permite al sistema priorizar y presentar al usuario como un evaluador en ciertos segmentos donde la entropía o duda es mayor lo cual optimiza el esfuerzo del etiquetado manual. Y nos permite construir un dataset orgánico mientras se va usando el sistema.

Observaciones: Si bien el enfoque de RLHF es muy popular computacionalmente es inestable y difícil de afinar para problemas de clasificación simple, por lo cual matemáticamente es más directo minimizar cross-entropy los con etiquetas corregidas de alta confianza que aproxima a una función de recompensa abstracta.

3.11. Motor de evaluación de calidad (Quality score)

3.11.1. Justificación del motor

En contact center la evaluación de la calidad de las llamadas muchas veces se hacen tradicionalmente escuchando las llamadas y se hace de manera manual tanto la transcripción como el cumplimiento de ciertas directrices como pueden ser:

- Saludo.
- Despedida.
- Aviso de que la llamada está siendo grabada.
- Cumplimiento del requerimiento.
- Satisfacción del cliente.

Y muchas veces no se puede analizar un lote grande o un lote completo precisamente por la limitación del tiempo por lo cual no se da un seguimiento real a como se maneja la atención en el contact center.

Por lo cual la idea es hacer un motor de evaluación de calidad el cual fue desarrollado en el módulo de scoring_engine.py el cual genera un indicador numérico de la calidad una vez acabada el análisis de las emociones, lo que se busca no es remplazar la supervisión humana, sino proveer un filtrado que nos permita ver que asesores tienen ciertos indicadores de riesgos y poder generar la retroalimentación al evaluado, lo cual facilita el proceso de retroalimentación con observaciones concretas y basadas en la evidencia textual del mismo análisis.

3.11.2. Arquitectura del Score

El Quality score se calcula sobre una escala numérica de 0 a 100 puntos, los cuales están distribuidos en 4 dimensiones independientes los cuales pueden ser modificados dependiendo de los requerimientos que tenga la empresa.

Pero nosotros lo que buscamos fue validar contra estándares de la industria de contact center latinoamericanos, en particular marcos de evaluación tomados por ICMI y adaptados para el contexto del proyecto.

Tabla 24

Tabla de dimensiones independientes

Dimensión	Puntaje Máximo	Criterios de Evaluación
Criterios de Evaluación	30 pts	Proporción de segmentos positivos y neutrales versus negativos. Bonus de 3 puntos por evolución positiva (inicio negativo y cierre positivo).
Palabras Clave	25 pts	Frecuencia de expresiones de empatía (+3 pts c/u, máx. 10 pts), keywords de resolución (+3 pts c/u, máx. 10 pts), bonus por ausencia de lenguaje prohibido (+5 pts), penalización por palabras prohibidas (-5 pts c/u, máx. -15 pts).
Resolución de Conflictos	25 pts	Evolucion emocional positiva en la última parte de la llamada (+8 pts), keywords de resolución en los últimos segmentos (+8 pts), tono final positivo o neutro (+5 pts), ausencia de picos de alta negatividad (intensidad > 0.7) (+4 pts).
Protocolo de Atención	20 pts	Detección de saludo formal al inicio de la llamada (+8 pts), despedida cordial al cierre (+8 pts), identificación del agente con nombre o cargo (+4 pts).

Nota. Elaboración propia de los autores.

3.11.3. Lexicones de palabras claves

La evaluación de las dimensiones de palabras clave y protocolo se apoya en lexicones predefinidos de términos y expresiones en español, diseñados específicamente para el contexto de contact center.

La estrategia de detección es mediante búsqueda de sub cadenas en texto normalizado a minúsculas lo que nos permite capturar variaciones morfológicas básicas sin requerir un análisis sintáctico complejo, garantizando una latencia de procesamiento inferior a 5ms por llamada independientemente de la duración.

Tabla 25

Tabla de lexicones de palabras claves

Categoría	Ejemplos de Términos	Función en el Scoring
Función en el Scoring	"buenos días", "buenas tardes", "bienvenido", "gracias por llamar", "mi nombre es", "en qué puedo ayudarle"	Detectados en los primeros 3 segmentos. Suma +8 pts en la dimensión Protocolo.
Despedidas	"fue un placer", "que tenga un buen día", "hasta luego", "algo más en lo que pueda ayudarle", "estamos para servirle"	Detectados en los últimos 3 segmentos. Suma +8 pts en la dimensión Protocolo.
Empatía	'entiendo", "comprendo", "lamento", "lo siento", "con gusto", "no se preocupe", 'vamos a resolver", 'le entiendo perfectamente'	Detectados en la transcripción completa. Cada término suma +3 pts (max. 10 pts). Indicador primario de calidad relacional.
Resolución	"resuelto", "solucionado", "quedó registrado", "confirmado", "procesado", 'le confirmo", "se envió", 'en las próximas horas"	Detectados en los últimos 5 segmentos y en la transcripción completa. Suman +3 pts c/u (máx. 10 pts). Indicador de cierre efectivo.

Nota. Los lexicones son configurables en el archivo scoring_engine.py. Lo cual permite a la empresa poder buscar los términos de interés y también el castigo por puntuación que debería tener dependiendo del caso.

3.11.4. Fórmula de cálculo y clasificación

El score se obtiene sumando los puntajes de las 4 dimensiones, cada una calculada de manera independiente con sus propias reglas, bonificaciones y penalizaciones la ecuación del Quality score es la siguiente:

$$QS = D\_tono + D\_keywords + D\_resolucion + D\_protocolo$$

Donde la dimensión D esta acotada en su rango máximo mediante la función max(0,min(max_dim, score_calculado)), esto garantizando que ninguna dimensión pueda superar su máximo ni producir algún valor negativo que afecte en la evaluación y estos resultados son clasificados de la siguiente manera.

Tabla 26

Tabla de cálculo y clasificación

Clasificación	Rango de Puntaje	Interpretación Operativa
Excelente	85 - 100 pts	La llamada cumple todos los protocolos, demuestra empatía efectiva y cierra con resolución confirmada. Referente para capacitación del equipo.
Bueno	70 - 84 pts	La llamada cumple los estándares básicos con oportunidades de mejoras identificables. Requiere retroalimentación puntual.
Empatía	50 - 69 pts	Detectados en la transcripción completa. Cada término suma +3 pts (max. 10 pts). Indicador primario de calidad relacional.
Deficiente	0 - 49 pts	La llamada incumple múltiples criterios. Activa automáticamente una alerta de escalamiento y requiere revisión urgente del supervisor.

Nota. Elaboración propia de los autores.

CAPITULO 4:

4. ANÁLISIS DE RESULTADOS

4.1. Pruebas de concepto

Con el propósito de validar la funcionalidad del prototipo, se desarrolló una prueba de concepto basada en escenarios reales y simulados. El sistema se diseñó como una API REST construida sobre FastAPI, utilizando modelos de procesamiento del habla y análisis de emociones disponibles en Hugging Face.

Se utilizaron los modelos Whisper (OpenAI) y Vosk para el reconocimiento del habla, mientras que para la detección de emociones se utilizaron modelos basados en XLM-RoBERTa y DistilRoBERTa. Además, se integró un módulo de diarizacion con Resemblyzer para detectar múltiples hablantes en el audio procesado.

La prueba se realizó con una colección de nueve clips de audio provenientes de datasets seleccionados por su variado contenido emocional y duración.

Los resultados del prototipo tras su análisis y operación incluyeron transcripciones segmento por segmento, etiquetas de emociones dominantes y gráficos interactivos a través de un panel de control HTML diseñado para visualizar las emociones a lo largo del tiempo, esto por el formato en que retorna la respuesta el prototipo. Durante el proceso, se midieron variables como el tiempo de procesamiento, la precisión del reconocimiento de voz, la coherencia de las emociones detectadas, además se configuraron diferentes pesos entre las señales de audio y de texto para evaluar el impacto de la fusión multimodal

4.2. Descripción del Escenario de Pruebas de los Dataset

Se diseño un corpus de prueba heterogéneo el cual está compuesto por 4 escenarios distintos seleccionados para estresar diferentes capacidades del sistema, como la detección del sarcasmo en el audio de queja telefónica, la segmentación de hablantes con entrevistas que se hicieron y la robustes con audios antiguos los cuales tiene ruido de fondo. A continuación, se describen los casos más representativos de cada variante.

- Escenario V1 (Renovación de línea telefónica): Llamada de agente de call center con información de paquetes promocionales en planes telefónicos.

Características: Voz clara, conversación pausada, ruido de fondo leve.

- Escenario V2 (Promoción de servicios y planes para clientes): Llamada telefónica para la promoción de servicios de telefonía e internet, renovación de dispositivos y descuentos. Características: Tono con interés, neutral, baja calidad de audio (telefónico), ruido de voces de fondo.
- Escenario V3 (Queja de Cliente): Llamada de renovación de línea, el usuario indica insatisfacción con promociones anteriores, cobro de impagos y falta de renovación de equipos. Se ofrece un nuevo plan. Características: Tono serio-molesto, ruido de audio, reclamo.
- Escenario V4 (Comprobación de Pagos /Reclamo): Llamada telefónica de un usuario reclamando un cobro indebido. Características: Tono pasivo-agresivo, sarcasmo, mala calidad de audio (telefónico).

En todos estos escenarios se puede ver como varían ciertos resultados del código los cuales los veremos en el siguiente punto

4.3. Análisis de Resultados

Para el análisis de resultados se presenta un resumen de los mismos, que se obtuvieron tras correr el prototipo usando como input 4 archivos de voz totalmente distintos. Donde cada caso fue evaluado minuciosamente considerando métricas clave como duración, emoción dominante, score de identidad, cantidad de transiciones emocionales y factores de tiempo como criterios ponderables que permitan establecer indicadores de eficiencia computacional

Ilustración 1

Tabla de valores obtenidos de las pruebas

ID	**Audio**	**Duración (s)**	**Emoción Dominante**	**Score**	**RTF**
V1	Renovación de línea telefónica	702.0	Neutral	80/100	-0.08
V2	Promociones para clientes	360.0	Neutral	65/100	-0.07
V3	Queja de Cliente	487.0	Neutral	58/100	0.03
V4	Comprobación de Pagos	670.0	Neutral	60/100	-0.08

Nota. Elaboración propia de los autores.

Análisis Individual de Audios

V1 - Renovación de línea telefónica: Se detectó un patrón dominante neutral, que se marcó como la emoción dominante los audios de los datasets utilizados. En su mayoría durante la llamada es el representante quien interviene durante la llamada manteniendo un tono profesional y neutro, por lo que predomina en la muestra, con porciones pequeñas donde se exhibe felicidad, enojo o tristeza detectada sin embargo esto puede deberse a cambios de entonación durante la conversación que coinciden con estas emociones.

V2 - Promociones para clientes: En este audio el sentimiento más prominente es neutral, presenta ruido de voces en el fondo del audio, se reconocen en algunas porciones del audio entonaciones reconocibles que se pueden interpretar como felices, enojo o tristeza, la interacción se puede calificar como cordial, sin mayores sobresaltos durante la llamada.

V3 - Queja de Cliente: La interacción se detectó como neutral, aunque el cliente presenta una queja respecto a la inconformidad con los servicios, la comunicación transcurre con normalidad, sin sobresaltos mayores y manteniendo el representante un tono profesional y el cliente se mantiene en calma.

V4 - Renovación de línea telefónica: El sistema la clasificó como «Neutral», aunque el tono real reflejaba sarcasmo en algunas ocasiones o aunque puede deberse a su pronunciación, el cliente se presenta serio con tonos de enojo, de las muestras presentadas esta detecto el tono enojado con mayor frecuencia de las 4 analizadas con un 12%. Este hallazgo concuerda con estudios que destacan las dificultades del análisis emocional para reconocer la ironía en las interacciones humanas (Mihalcea y Strapparava, 2005).

Los resultados concuerdan con hallazgos previos en la literatura especializada, donde emociones como la ira, la tristeza o la alegría se detectan con mayor precisión que la sorpresa o el disgusto (Poria et al., 2019). El sistema mostró un rendimiento comparable al de propuestas recientes como EmoGraph (Zhou et al., 2021) o Multimodal Transformer (Siriwardhana et al., 2020), en términos de estabilidad emocional y detección de transiciones.

Evaluación Técnica

RTF medio: 0,5, lo que indica viabilidad en tiempo real en entornos de producción.

Precisión transcripcional: Whisper mostró una tasa de error inferior al 5 % en audio limpio, superando a Vosk (WER ~12 %).

Visualización de resultados: el panel de control permitió la interpretación de trayectorias emocionales y picos críticos, lo que reforzó su aplicabilidad para el análisis de calidad, la educación en línea y el servicio al cliente.

Limitaciones identificadas

- Sensibilidad al ruido y acentos no detectados durante el entrenamiento.
- Rendimiento deficiente con emociones ambiguas, sarcásticas o compuestas.
- Dependencia de la longitud del segmento para obtener inferencias fiables.

Recomendaciones para trabajos futuros

- Entrenamiento con corpus etiquetados emocionalmente en español latinoamericano.
- Adición de capas para detectar ironía, sarcasmo y emociones mixtas.
- Implementación de normalización de audio y eliminación automática de ruido.
- Ampliación a ámbitos específicos como la salud mental, la educación o el análisis político.

En síntesis, la prueba de concepto demuestra la viabilidad técnica y funcional del sistema propuesto. La arquitectura modular permitió una evaluación exhaustiva de la cadena de procesamiento: desde la transcripción de audio hasta la interpretación emocional, ofreciendo resultados coherentes, visualizables y explicables. Las pruebas realizadas validan el enfoque multimodal como una forma prometedora de enriquecer la transcripción del habla con dimensiones afectivas, mejorando su aplicación en múltiples sectores.

CAPITULO 5:

5.1 CONCLUSIONES Y RECOMENDACIONES

5.1.1 Conclusiones

- Se desarrollo con éxito un prototipo funcional que implementado como una API modular y que ha sido contenerizada, lo que permite el análisis con etiquetas emocionales por segmentos y hablantes, aplicándose a el monitoreo de llamadas, interacciones y calidad de servicio de call centers.
- Luego de realizar experimentaciones, pruebas de conceptos y comparativas de resultados, se determino que los modelos Whisper de OpenAI han superado de manera significativa a otros modelos sometidos a pruebas como Vosk, validando la elección del motor de reconocimiento automático de voz e implementando técnicas modernas y robustas.
- Se integro el análisis de detección emocional a través de arquitectura de fusión multimodal, combinando el análisis semántico; generadas de las transcripciones con los modelos RoBERTa y XLM-RoBERTa; con el análisis acústico de la prosodia de los audios utilizando Wav2Vec2, elevando la confianza del sistema y reduciendo la ambigüedad en la identificación de tonalidad emocional.
- Se determinó que Whisper (Small) es el motor de transcripción ideal al ofrecer un equilibrio óptimo entre bajo consumo de recursos y alta precisión (WER 5-10%), superando las limitaciones de acento y ruido presentes en otros modelos.
- La interfaz del prototipo funcional permite al usuario la carga de audios y la visualización interactiva de los resultados de transcripción, diarizacion, líneas de tiempo de la evolución emocional y las evaluaciones de calidad o Quality Score para presentar los resultados del análisis emocional facilitando su uso practico.
- El uso de suavizado exponencial y la reducción a 4 macro-categorías emocionales permitió transformar predicciones técnicas volátiles en un flujo de información estable

y fácilmente interpretable para el usuario final.

- Se mitigo una limitación clave que fue identificada durante la investigación del estado del arte: la baja precisión de los modelos de detección de emociones en español. Esta debilidad se ha mitigado al implementar una estrategia bilingüe hibrida que combinaría el análisis directo de XML-RoBERTa en Español y el análisis luego de su traducción con MarianaMT y DistilRoBERTa en Ingles. Este enfoque demostró resultados superiores a el uso de un único modelo al validar las necesidades de arquitecturas adaptativas para la superación de las restricciones idiomática y de datos en dominios específicos como puede ser el del análisis emocional.
- Al impulsar los resultados con interpretaciones de un LLM como lo es Ollama nos ayuda a la interpretación y visualización de la data con un informe en formato pdf.

5.1.2 Recomendaciones

- Se puede implementar la cuantificación de pesos complementando la R6 se recomienda la cuantización INT8/FP16. Con esto se reduce los tamaños de los modelos en un 50% y se acelera la inferencia hasta en un x3 en hardware compatible, permitiendo que corra con una GPU menos potente con mayor soltura.
- Se puede implementar un ciclo de retroalimentación en la cual, si el usuario ve que la detección de la emoción o el texto está mal transcrita, genera un feedback al sistema y se lo reentrena cada 500 muestras corregidas.
- Se podría implementar una persistencia poliglota usando diferentes bases de datos seria fundamental para mejorar el sistema ya que maneja audio, texto y métricas, la combinación ideal sería SQL el cual integra metadatos y relaciones y la NoSQL para flexibilidad de las transcripciones emocionales y Objective Storage para el almacenamiento optimizado de los archivos binarios de audio. Esta sinergia elimina cuellos de botella y garantiza la integridad referencial y permite una escalabilidad lineal frente a volúmenes de datos masivos, no solo simplificando la operación actual si no

también la base de la técnica para los procesos de preentrenamiento y fine-tuning, lo que permitirá la extracción de datasets masivos para el reentrenamiento de modelos y garantizar una mejora continua en la precisión del modelo.

- Se recomienda estandarizar las métricas importantes a la hora de la generación de calificaciones para tener una buena configuración del Quality Score, ya que se puede adaptar a diferentes requerimientos solo modificando las palabras gatillo y poder detectar de la mejor manera problemas con el agente.

Bibliografía

Acharya, A., Sethi, M., & Dogra, D. (2021). Emotion recognition using speech and text: A review. *IEEE Transactions on Affective Computing*, *12*(3), 579–593. https://doi.org/10.1109/TAFFC.2021.3081183

Alpha Cephei. (2020). *Vosk Speech Recognition Toolkit*. https://alphacephei.com/Vosk/

Baevski, A., Zhou, Y., Mohamed, A., & Auli, M. (2020). wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in Neural Information Processing Systems*, *33*, 12449–12460.

Barbieri, F., Camacho-Collados, J., Espinosa-Anke, L., & Neves, L. (2022). Pysentimiento: A python toolkit for sentiment analysis and emotion detection. *ArXiv Preprint*, *1*(2). https://doi.org/arXivpreprint arXiv:2206.12200

Barrault, L., Chung, Y., Coria, M., Dale, D., Dong, N., Duquenne, P., & Elsahar, H. (2023). SeamlessM4T: Massively Multilingual & Multimodal Machine Translation. *ArXiv*, *3*(1). https://doi.org/arXiv:2308.11596v3

Bechtold, B., & Meyer, M. (2013). *SoundFile: Audio module based on libsndfile*. PyPI. https://pypi.org/project/SoundFile/

Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., & Stoyanov, V. (2020). Unsupervised Cross-lingual Representation Learning at Scale. En Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. *Association for Computational Linguistics*, 8440–8451. https://doi.org/10.18653/v1/2020.acl-main.747

Costa-Jussa, M., Cross, J., Çelebi, O., Elbayad, M., & Heafield, K. (2022). No Language Left Behind: Scaling Human-Centered Ma-chine Translation. *ArXiv*, 2207.04672. https://doi.org/10.48550/arXiv.2207.04672

Cowen, A., Elfenbein, H., Laukka, P., & Keltner, D. (2019). Mapping 24 emotions conveyed

by brief human vocalizations. *American Psychologist*, *74*(6), 698–712. https://doi.org/10.1037/amp0000399

Daveni. (2021). *Twitter XLM-RoBERTa Emotion Es [Software model]*. Hugging Face. https://huggingface.co/daveni/twitter-xlm-roberta-emotion-es

Davis, S., & Mermelstein, P. (1980). Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, *28*(4), 357–366. https://doi.org/10.1109/tassp.1980.1163420

Eyben, F., Wöllmer, M., & Schuller, B. (2010). Opensmile: The munich versatile and fast open-source audio feature extractor. *Proceedings of the 18th ACM International Conference on Multimedia*, 1459–1462. https://tinyurl.com/2cstr3np

Fernández, Y. (2023, noviembre 13). *OpenAI Whisper: qué es, cómo funciona y cómo puedes usar esta inteligencia artificial para transcribir audios*. Xataka.com; Xataka. https://www.xataka.com/basics/openai-whisper-que-como-funciona-como-puedes-usar-esta-inteligencia-artificial-para-transcribir-audios

Gutiérrez, J. M. (s/f). *"Análisis de Sentimiento en Audio mediante Inteligencia Artificial orientado al idioma Español"*. Uc3m.es. Recuperado el 3 de marzo de 2026, de https://e-archivo.uc3m.es/rest/api/core/bitstreams/5a9927e9-c8e6-4bfc-b751-15e1a41678f1/content

Hartmann, J. (2022). *Emotion English DistilRoBERTa-base [Modelo de software]*. Hugging Face. https://huggingface.co/j-hartmann/emotion-english-distilroberta-base

InfoBayAI/Dual-Channel_Audio_Call-Center_Spanish · Datasets at Hugging Face. (s. f.). https://huggingface.co/datasets/InfoBayAI/Dual-Channel_Audio_Call-Center_Spanish

Jemine, C. (2019). *Resemblyzer: A python package to analyze and compare voices with deep learning*. GitHub. https://github.com/resemble-ai/Resemblyzer

Lang-Lenton Ferreiro, J. (2024). *Diarización y resumen de intervenciones parlamentarias en el Parlamento de Canarias*.

McFee, B., Raffel, C., Liang, D., Ellis, D., McVicar, M., Battenberg, E., & Nieto, O. (2015). librosa: Audio and Music Signal Analysis in Python. *En Proceedings of the 14th Python in Science Conference*, 18–25. https://doi.org/10.25080/Majora-7b98e3ed-003

Miguel, M. (2025). Análisis de emociones en textos en español mediante traducción automática y modelos BERT multilingües. *Zenodo (CERN European Organization For Nuclear Research)*. https://doi.org/10.5281/zenodo.17525359

OpenAI. (2022). *Whisper: robust speech recognition via large-scale weak supervision*. OpenAI Research. https://github.com/openai/Whisper

Ortega-Beltrán, E., Cabacas-Maso, J., Benito-Altamirano, I., & Ventura, C. (2024). Better Spanish emotion recognition in-the-wild: Bringing attention to deep spectrum voice analysis. En *arXiv [cs.SD]*. https://doi.org/10.48550/ARXIV.2409.05148

Pérez, J., Giudici, J., & Luque, F. (2021). Pysentimiento: A Python Toolkit for Sentiment Analysis and Social NLP tasks. *ArXiv Preprint*, 2106.09462. https://doi.org/10.48550/arXiv.2106.09462

Politécnica, U., Madrid, D. E., Salgado Infantes, D., & Rodríguez Fernández, V. (s/f). *Desarrollo de una aplicación basada en Inteligencia Artificial para la transcripción avanzada de audio*. Upm.es. Recuperado el 5 de marzo de 2026, de https://oa.upm.es/76403/1/TFG_DANIEL_SALGADO_INFANTES.pdf

Radf-Povey, D., Ghoshal, A., Boulianne, G., Burget, L., Glembek, O., Goel, N., Hannemann, M., Motlicek, P., Qian, Y., Schwarz, P., Silovsky, J., Stemmer, G., & Vesely, K. (2011). *The Kaldi speech recognition toolkit. En IEEE 2011 workshop on automatic spe*. https://kaldi-asr.org/doc/about.html

Rinnert, T., Walsh, J., Fleury, C., Coppin, G., Duval, T., & Thomas, B. H. (2023). AR

presentation of team members' performance and inner status to their leader: A comparative study. *Applied Sciences (Basel, Switzerland)*, *14*(1), 123. https://doi.org/10.3390/app14010123

Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019). DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *ArXiv Preprint*, 1910.01108. https://doi.org/10.48550/arXiv.1910.01108

Stappen, L., Christ, L., Meßner, E., & Cambria, E. (2021). *The MuSe 2021 Multimodal Sentiment Analysis Challenge : Sentiment , Emotion , Physiological-Emotion , and Stress*. *24*, 5–14. https://doi.org/10.1145/3475957.3484450

Tiedemann, J., & Thottingal, S. (2020). OPUS-MT – Building open translation services for the World. En Proceedings of the 22nd Annual Conference of the European Association for Machine Translation. *European Association for Machine Translation*, 479–480. https://aclanthology.org/2020.eamt-1.61

Vera, D., Araque, O., & Iglesias, C. (2021). *GSI-UPM at IberLEF2021: Emotion Analysis of Spanish Tweets by Fine-tuning the XLM-RoBERTa Language Model. En Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2021)*. CEUR Workshop Proceedings.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Le Scao, T., Gugger, S., & Rush, A. (2020). Transformers: State-of-the-Art Natural Language Processing. En Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations. *Association for Computational Linguistics*, 38–45. https://doi.org/10.18653/v1/2020.emnlp-demos.6

Yang, S., Chi, P., Chuang, Y., Lai, C., Lakhotia, K., Lin, Y., & Lee, H. (2021). SUPERB: Speech Processing Universal PERformance Benchmark. *En Interspeech*, 1194–1198. https://doi.org/10.21437/Interspeech.2021-1775

Zhang, Y., Zhao, X., & Wang, J. (2021). Speech emotion recognition using deep learning: A review. *IEEE Transactions on Affective Computing*, *12*(3), 712–734. https://doi.org/10.1109/TAFFC.2020.2981444

Apéndices

- Primer repositorio

https://github.com/richardcev/Voz-a-Texto-Emocional.git

- Segundo y repositorio principal

https://github.com/Chcortezos10/Voz-a-texto-con-enfoque-en-el-tono-emocional.git