Universidad de Palermo

Tesis de Maestría

MODELO DE REFERENCIA PARA LA APLICACIÓN DE INTELIGENCIA ARTIFICIAL EN LA DETECCIÓN DE ESTAFAS TELEFÓNICAS Y DIGITALES

Lic. César David Boronat

FACULTAD DE INGENIERÍA

Maestría en Tecnología de la Información

Tutor de Tesis

Mg. Juan Manuel Miguez

Ciudad Autónoma de Buenos Aires, Argentina 2026

PÁGINA DE APROBACIÓN DEL COMITÉ

_____

DIRECTOR DE TESIS

_____

JURADO INTERNO 1

_____

JURADO INTERNO 2

_____

JURADO INTERNO 3

Defensa de la Tesis

Ciudad Autónoma de Buenos Aires, a los 17 días del mes de marzo de 2026.

AGRADECIMIENTOS

Quiero expresar mi agradecimiento a la institución y a la dirección de la carrera por abrirme las puertas a contenidos y espacios de aprendizaje que no solo ampliaron mis conocimientos técnicos, sino que también impulsaron mi desarrollo profesional y personal.

A mi tutor, por su objetividad, por cada pregunta que me desafió a pensar más allá de lo expresaba mi idea y por ayudarme a mirar cada problema desde nuevas perspectivas. Su guía fue fundamental para que este trabajo creciera en profundidad y claridad.

A mi jefa, por alentarme a asumir nuevos desafíos y confiar en mi crecimiento.

A mis amigos, por la comprensión cuando llegaba tarde, o cuando directamente no podía llegar, porque estaba cursando o avanzando con la tesis.

A mi novia, por acompañarme en todo el recorrido, por su paciencia en las noches largas, por entender los momentos de cansancio y por sostenerme cuando el camino se hacía más exigente.

Finalmente, quiero agradecer a las personas que, por la temática y el contenido sensible de esta investigación, no pueden ser nombradas individualmente, pero que con generosidad prestaron sus voces para la construcción del MVP.

RESUMEN

En este proyecto se pretende examinar las causas, medios y efectos producidos

por estafas a través de canales telefónicos y digitales como también los mecanismos que permitan su detección y prevención. Se presentarán las principales tecnologías involucradas en la generación de estos fraudes. Se identificarán los sectores de la sociedad más vulnerables frente a estas situaciones. Se establecerá la relevancia de la problemática en la actualidad y el rol de la inteligencia artificial en la identificación y mitigación de los actos delictivos a tratar. Se expondrán las herramientas y metodología disponibles y los resultados que pueden logran frente al uso de mecanismos tradicionales. Se propondrá un modelo de aplicación de inteligencia artificial en la detección de estafas telefónicas y digitales con el fin de generar un MVP (producto mínimo viable por su siglas en inglés) como entregable y herramienta de validación.

ABSTRACT

This project aims to examine the causes, means, and effects of scams through telephone and digital channels, as well as the mechanisms that enable their detection and prevention. The main technologies involved in the generation of these frauds will be presented. The most vulnerable sectors of society will be identified. The current relevance of the problem and the role of artificial intelligence in identifying and mitigating the criminal acts to be addressed will be established. The available tools and methodologies will be presented, along with the results they can achieve compared to traditional mechanisms. A model for applying artificial intelligence to detect telephone and digital scams will be proposed, with the goal of generating an MVP (minimum viable product) as a deliverable and validation tool.

TABLA DE CONTENIDOS

1. INTRODUCCIÓN.....15
1.1 Antecedentes y Planteamiento del problema.....15
1.2 Justificación del estudio.....20
1.3 Objetivos.....22
1.3.1 Objetivo principal:.....22
1.3.2 Objetivos secundarios:.....22
1.4 Estructura de la tesis.....23
1.5 Limitaciones del estudio.....23
1.6 Metodología de investigación.....25
2. MARCO TEÓRICO.....26
2.1 Ingeniería Social.....27
2.1.1 Definición.....27
2.1.2 Mecanismos de manipulación.....28
2.1.3 Manifestaciones principales hoy.....28
2.1.4 Síntesis.....29
2.2 Tipos de amenazas: Phishing, Vishing, DeepFake.....30
2.2.1 Phishing.....30
2.2.1.1 Definición.....30
2.2.1.2 Clasificación de los ataques de phishing.....31
2.2.1.3 Factores de vulnerabilidad.....32
2.2.1.4 Técnicas más comunes.....32
2.2.1.5 Motivaciones e impacto.....33
2.2.1.6 Estrategias de mitigación.....34
2.2.1.7 Síntesis.....35
2.2.2 Vishing.....37
2.2.2.1 Definición.....37
2.2.2.2 Evolución y características.....37
2.2.2.3 Ejemplos comunes de vishing.....38
2.2.2.4 Impacto económico y social.....40
2.2.2.5 Factores de vulnerabilidad.....40
2.2.2.6 Tendencias actuales.....41
2.2.2.7 Problemas y retos.....42
2.2.2.8 Síntesis.....42
2.2.3 DeepFake.....43
2.2.3.1 Definición.....43
2.2.3.2 El DeepFake y la posverdad.....44
2.2.3.4 Clasificación.....45
2.2.3.5 Problemas y riesgos asociados.....47
2.2.3.6 Síntesis.....48
2.2.4 Análisis de transición.....48
2.3 TTS y ASR.....50

2.3.1 Text-to-Speech (TTS)....50
2.3.1.1 Definición....51
2.3.1.2 Evolución histórica....51
2.3.1.3 Arquitecturas y modelos principales....53
2.3.1.4 Calidad TTS....55
2.3.1.5 Limitaciones y desafíos....56
2.3.1.6 Aplicaciones actuales....57
2.3.1.7 Riesgos y usos maliciosos....58
2.3.1.8 Síntesis....58
2.3.2 Automatic Speech Recognition (ASR)....58
2.3.2.1 Definición....59
2.3.2.2 Evolución histórica....60
2.3.2.3 Enfoques End-to-End....61
2.3.2.4 Desafíos....62
2.3.2.5 Aplicaciones actuales....63
2.3.2.6 Síntesis....64
2.4 Voice Cloning....65
2.4.1 Definición....65
2.4.2 Aplicaciones legítimas....66
2.4.3 Riesgos....67
2.4.5 Impacto social....68
2.4.6 Síntesis....68
2.5 Modelos de IA....69
2.5.1 Definición....69
2.5.2 El Aporte de las disciplinas....69
2.5.3 Breve Historia Contemporánea....71
2.5.4 Clasificación de modelos de IA....73
2.5.5 Aplicaciones relevantes....75
2.5.6 Desafíos....76
2.5.7 Síntesis....76
2.6 Machine Learning: Aprendizaje supervisado y no supervisado....77
2.6.1 Definición....77
2.6.2 Aprendizaje supervisado....80
2.6.2.1 Regresión Logística....81
2.6.2.2 Árboles de decisión....82
2.6.2.3 Redes neuronales artificiales (ANN)....83
2.6.3 Deep Learning como evolución del supervisado....85
2.6.3.1 Transformers....86
2.6.4 Aprendizaje no supervisado....87
2.6.4.1 Clustering....88
2.6.4.2 Reducción de dimensionalidad (PCA)....89
2.6.4.3 Autoencoders....89

2.6.4.4 GANs en no supervisado....90
2.6.5 Comparación de los enfoques....90
2.6.6 Síntesis....91
2.7 Aplicación móviles....92
2.7.1 Introducción....92
2.7.2 Evolución de las aplicaciones móviles de seguridad....93
2.7.3 Aplicaciones móviles potenciadas por Machine Learning....94
2.7.4 Síntesis....94
3. MODELO....95
3.1 Presentación del modelo....95
3.1.1 Propuesta....96
3.1.2 Limitaciones....98
3.1.3 Factores del diseño....100
3.2 Arquitectura del modelo....101
3.2.1 Visión del Diseño....102
3.2.1.1 Criterios de diseño....103
3.2.2 Comparativa de enfoques arquitectónicos....104
3.2.2.1 Cloud Computing....104
3.2.2.2 Edge Computing (On-Device)....105
3.2.2.3 Arquitectura Híbrida (Edge - Cloud)....106
3.2.2.4 Punto de vista económico : costos de los enfoques....108
3.2.2.5 Definición del enfoque....110
3.2.3 Flujo operacional del modelo....111
3.3 Selección tecnológica de los módulos....114
3.3.1 Criterios para el análisis....114
3.3.2 Tecnologías analizadas....115
3.3.2.1 Captura de audio y entorno de ejecución (VoIP / WebRTC)....116
3.3.2.2 Reconocimiento automático del habla (ASR)....117
3.3.2.3 Clasificador basado en inteligencia artificial....118
3.3.2.4 Síntesis de voz (TTS)....119
3.3.2.5 Notificaciones....121
3.3.2.6 Seguridad y privacidad....122
3.3.3 Conclusiones....123
3.3.4 Aclaraciones adicionales....125
3.4 Viabilidad del modelo....125
3.4.1 Viabilidad técnica....125
3.4.2 Viabilidad económica....129
3.4.3 Viabilidad regulatoria y de privacidad....131
3.4.4 Viabilidad operativa....132
3.4.5 Conclusión integral....133
4. MVP : CLASIFICADOR DE AUDIOS....134
4.1. Objetivo del MVP....134
4.2 Alcance....135

4.3 Enfoque del MVP....137
4.3.1 Ingeniería social telefónica y su impacto en el análisis de audio....138
4.3.2 Justificación del enfoque acústico y textual....139
4.4 Herramientas y entorno de ejecución....140
4.5 El dataset (Conjunto de datos)....141
4.6 Preprocesamiento de audio....143
4.7 Pipeline....145
4.8 Modelo implementado....147
4.9 Resultados experimentales....150
4.9.1 Métricas de evaluación....151
4.9.2 Resultados del clasificador de texto....152
4.10 Análisis de resultados preliminares....155
4.11 Validaciones....156
4.12 Síntesis....157
5. CONCLUSIONES....158
6. FUTURAS LÍNEAS DE INVESTIGACIÓN....160
7. BIBLIOGRAFÍA....161
7.1 Referencias Bibliográficas....161
7.2 Documentos Electrónicos....162
7.3 Marco Normativo Argentino....175
8. Anexo....176
8.1 Procedimiento Completo para la Reproducción del MVP ProxIA....176
8.1.1 Introducción....176
8.1.2 Requisitos Previos....177
8.1.3. Herramientas Utilizadas....178
8.1.4 Creación del Entorno de Trabajo....178
8.1.4.1 Acceso a Google Drive....178
8.1.4.2 Creación del Proyecto....179
8.1.5 Preparación del Dataset....181
8.1.5.1 Grabación de Audios....181
8.1.5.2 Carga de Audios....182
8.1.6 Configuración en Google Colab....182
8.1.6.1 Acceso a Colab....182
8.1.6.2 Montaje de Drive....183
8.1.7 Ejecución del Modelo Acústico....185
8.1.7.1 Conversión de Audio....186
8.1.7.2 Extracción de Características (MFCC)....187
8.1.7.3 División del Dataset....188
8.1.7.4 Arquitectura del Modelo CNN....189
8.1.7.5 Entrenamiento....190
8.1.7.6 Evaluación y Resultados....191
8.1.8 Ejecución del Modelo Textual....193

8.1.8.1 Instalación y Configuración de Whisper..... 194
8.1.8.2 Transcripción Automática..... 194
8.1.8.3 Entrenamiento del Modelo Textual..... 195
8.1.9 Fusión Multimodal..... 195
8.1.10 Reproducción con Nuevos Audios..... 196
8.1.11 Ubicación Final de Resultados..... 197
8.1.12 Consideraciones de Reproducibilidad..... 198
8.1.13 Disponibilidad del Código Fuente..... 198
8.1.16 Código..... 199
Notebook 01..... 199
Notebook 02..... 214
8.1.15 Cierre..... 225

LISTAS DE TABLAS

Tabla 1. Estadísticas de ataques de Phishing y Vishing. Elaboración Propia mediante datos obtenidos por Zscaler, 2024 y Kaspersky, 2024..... 17
Tabla 2 Comparación Arquitectura TTS. Fuente: Elaboración propia con fines académicos..... 53
Tabla 3. Clasificación de modelos AI. Fuente: Elaboración propia con fines académicos..... 74
Tabla 4. Comparación entre aprendizaje supervisado y no supervisado. Fuente: Elaboración propia con fines académicos..... 91
Tabla 5. Comparación entre enfoques de arquitectura. Fuente: Elaboración propia con fines académicos..... 108
Tabla 6. Costos de referencia - Servicios en la nube. Fuente: Elaboración propia con fines académicos basados en datos obtenidos en noviembre de 2025..... 109
Tabla 7. Métricas del clasificador acústico. Elaboración propia con fines académicos basados en resultados obtenidos durante la ejecución..... 151
Tabla 8. Métricas del clasificador de texto. Elaboración propia con fines académicos basados en resultados obtenidos durante la ejecución..... 153

LISTAS DE FIGURAS

Fig. 1. Modelo de sistema vishing potenciado por AI. Fuente: Elaboración propia con herramienta Draw.io (ahora app.diagrams.net)..... 16
Fig. 2 Ciclo de vida de un ataque de phishing.Fuente: Elaboración propia con fines académicos..... 31
Fig. 3. Ejemplo ficticio de correo de phishing bancario. Fuente: Elaboración propia con fines académicos..... 33
Fig. 4 Ejemplo de ataque de Smishing. Fuente: Elaboración propia con fines académicos..... 36
Fig. 5 Esquema de ataque de vishing. Fuente: Elaboración propia con fines académicos..... 39
Fig. 6 Transformación rostro por deep fake . Fuente : Elaboración propia con asistencia de inteligencia artificial (ChatGPT, 2025), basada en Noriega (s.f.) y Zucconi (2018)..... 44
Fig.7 Deepfake de Tom Cruise. (Imagen: Chris Ume)..... 46
Fig. 8 Diagrama general de un sistema TTS Fuente: Elaboración propia con fines académicos..... 51
Fig 9. Línea de tiempo de la evolución TTS. Fuente: Elaboración propia con fines académicos..... 52
Fig. 10 Esquema básico de un sistema ASR. Fuente: Elaboración propia con fines académicos..... 59
Fig 11. Línea de tiempo de la evolución ASR. Fuente: Elaboración propia con fines académicos..... 61
Fig. 12 Esquema básico de Voice Cloning. Fuente: Elaboración propia con fines académicos..... 66
Fig. 13 Diagrama de disciplinas influyentes en la IA . Fuente: Elaboración propia con fines académicos..... 70
Fig 14 Línea de tiempo evolución de la IA en la historia contemporánea. Fuente: Elaboración propia con fines académicos..... 72
Fig 15 Esquema tradicional de programación. Fuente: Elaboración propia con fines académicos..... 78
Fig. 16 Esquema básico de ML (Aprendizaje Supervisado). Fuente: Elaboración propia con fines académicos..... 79
Fig. 17 Representación curva Sigmoide . Fuente: Elaboración propia con fines académicos mediante herramienta google colab..... 82

Fig. 18 Árbol de clasificación de Smishing simplificado. Fuente: Elaboración propia con fines académicos.....83
Fig. 19 Red neuronal artificial simplificada. Fuente: Elaboración propia con fines académicos.....84
Fig. 20 Clustering simplificado. Fuente: Elaboración propia con fines académicos....89
Fig. 21 Flujo simplificado detección de fraudes. Fuente: Elaboración propia con fines académicos.....93
Fig. 22 Modelo simplificado aplicación para detección de fraudes .Fuente: Elaboración propia con fines académicos.....97
Fig. 23 Flujo general de operación de ProxIA. Fuente: Elaboración propia con fines académicos.....98
Fig. 24 Modelo reestructurado de aplicación para detección de fraudes en tiempo real. Fuente: Elaboración propia con fines académicos.....99
Fig. 25 Flujo de operación del modelo. Fuente: Elaboración propia con fines académicos.....113
Fig. 26 Diagrama de Componentes preliminar ProxIA. Fuente: Elaboración propia con fines académicos.....114
Fig. 27 Diagrama para toma de decisión del modelo. Fuente: Elaboración propia con fines académicos con herramienta Draw.io.....128
Fig. 28 Representación MFCC utilizada como entrada del modelo. Fuente: Elaboración propia con fines académicos.....145
Fig. 29 Representación de pipeline MPV. Fuente: Elaboración propia con fines académicos.....147
Fig. 30 Presentación de modelos usados en pipeline MPV. Fuente: Elaboración propia con fines académicos.....151
Fig. 31 Matriz de confusión del clasificador acústico. Elaboración propia con fines académicos basados en resultados obtenidos durante la ejecución.....153
Fig. 32 Matriz de confusión del clasificador de texto. Elaboración propia con fines académicos basados en resultados obtenidos durante la ejecución.....155
Fig. 33 Validaciones del modelo acústico. Elaboración propia con fines académicos basados en resultados obtenidos durante la ejecución.....157
Fig. 34 Validaciones del modelo acústico. Elaboración propia con fines académicos basados en resultados obtenidos durante la ejecución.....157

LISTAS DE SIGLAS

AE: Autoencoder (Autoencoder).

AED: Attention-based Encoder-Decoder (Codificador-Decodificador Basado en Atención).

ANN: Artificial Neural Network (Red Neuronal Artificial).

APWG: Anti-Phishing Working Group (Grupo de Trabajo Anti-Phishing).

ASR: Automatic Speech Recognition (Reconocimiento Automático del Habla).

CTC: Connectionist Temporal Classification (Clasificación Temporal Conexionista).

Cloud Computing : Servicios de computación en la nube

Deepfake de voz: Voz generada sintéticamente para imitar a una persona real.

DL: Deep Learning (Aprendizaje Profundo).

DNS: Domain Name System (Sistema de Nombres de Dominio).

DT: Deepfake Technology (Tecnología Deepfake).

Edge Computing : Servicios de computación en el borde

GANs: Generative Adversarial Networks (Redes Generativas Antagónicas).

GPU :Graphics Processing Unit (Unidad de procesamiento de gráficos)

HMM : Hidden Markov Model (Modelo Oculto de Markov)

IA: Inteligencia Artificial.

IOT: Internet of things (Internet de las cosas)

IVR: Interactive Voice Response (Respuesta de Voz Interactiva).

MCD: Mel-Cepstral Distortion (Distorsión Mel-Cepstral).

MFA: Multi-Factor Authentication (Autenticación Multifactor).

MFCC: Mel Frequency Cepstral Coefficients (Coeficientes Cepstrales en las Frecuencias de Mel).

ML: Machine Learning (Aprendizaje Automático).

MITM: Man in the Middle (Ataque de Intermediario).

MOS: Mean Opinion Score (Puntaje Medio de Opinión, métrica subjetiva de calidad del habla).

MVP: Minimum Viable Product (Producto Mínimo Viable).

NLP: Natural Language Processing (Procesamiento del Lenguaje Natural).

OCR: Optical Character Recognition (Reconocimiento Óptico de Caracteres).

PCA: Principal Component Analysis (Análisis de Componentes Principales).

PCM : Pulse Code Modulation (Modulación por Código de Pulsos)

PII: Personally Identifiable Information (Información Personal Identificable).

Phisher: Persona que lleva a cabo un ataque de phishing.

Robocalls: Llamadas telefónicas automáticas con mensajes pregrabados.

RNN: Recurrent Neural Network (Red Neuronal Recurrente).

RNN-T: Recurrent Neural Network Transducer (Transductor de Red Neuronal Recurrente).

SDK : Software Development Kit (Kit de desarrollo de software)

SE: Social Engineering (Ingeniería Social).

SIP : Session Initial Protocol (Protocolo de inicio de Sesión)

SLR: Systematic Literature Review (Revisión Sistemática de Literatura).

SMiShing: SMS Phishing (Phishing por SMS).

Softphone : Software, aplicativo, para realizar y recibir llamadas de VoIP.

Spoofing: Técnica de ciberdelincuencia basada en la suplantación de identidad para engañar a usuarios o sistemas.

SRTP: Secure Real-time Transport Protocol (Protocolo de transporte seguro en tiempo real)

SVM: Support Vector Machine (Máquina de Vectores de Soporte).

TTS: Text to Speech (Conversión de Texto a Voz).

TF-IDF: Term Frequency–Inverse Document Frequency (Frecuencia de Término - Frecuencia Inversa de Documento)

URL: Uniform Resource Locator (Localizador Uniforme de Recursos).

Vishing: Phishing por voz (Voice Phishing).

Voice Cloning: Clonación de voz, técnica de IA para replicar voces humanas.

VoIP: Voice over Internet Protocol (Voz sobre Protocolo de Internet).

WebRTC: Web Real-Time Communication (Comunicaciones Web en tiempo real).

WER: Word Error Rate (Tasa de Error de Palabras).

W3C: World Wide Web Consortium (Consorcio mundial web)

Zero Day: Vulnerabilidad desconocida o ataque no documentado ni corregido previamente.

1. INTRODUCCIÓN

El propósito de este trabajo es proponer una solución en la mitigación de estafas digitales y telefónicas, estableciendo como punto de partida una introducción a la problemática, definiciones necesarias y finalmente la presentación de un modelo que permita hacer frente a estas amenazas.

1.1 Antecedentes y Planteamiento del problema

En la actualidad, la inteligencia artificial ya ha dejado de ser una tecnología emergente. La afirmación anterior se corresponde cuando sabemos que hoy una persona con acceso a internet, un teléfono móvil u otro dispositivo inteligente puede hacer uso de esta herramienta, incluso, de manera gratuita.

En paralelo, y como se verá en su apartado dedicado, han evolucionado herramientas como conversores de texto a voz conocidos como TTS (Texto a diálogo por sus siglas en inglés) (Amazon Web Services, 2024). Estos aplicativos reproducen en forma de audio cualquier texto escrito con diferentes lenguajes, voces y acentos. De manera inversa el ASR (reconocimiento automático de voz por sus siglas en el inglés) permite procesar la voz humana transformándola a texto .

Los elementos mencionados pueden trabajar en conjunto siendo posible la comunicación entre un agente virtual y un ser humano incluso sin que este detecte la

diferencia respecto a hablar con otro par.

Un modelo de inteligencia artificial puede ser entrenado para generar diálogos coherentes y creíbles, un motor de TTS puede ser configurado con voces que emulen familiares cercanos, todo lo necesario para producir un engaño como el Vishing (tipo de estafa realizada a través de llamadas telefónicas donde atacantes, empleando técnicas como suplantación de identidad, intentan persuadir a la víctima para que ésta le proporcione información valiosa como datos personales y sensibles).

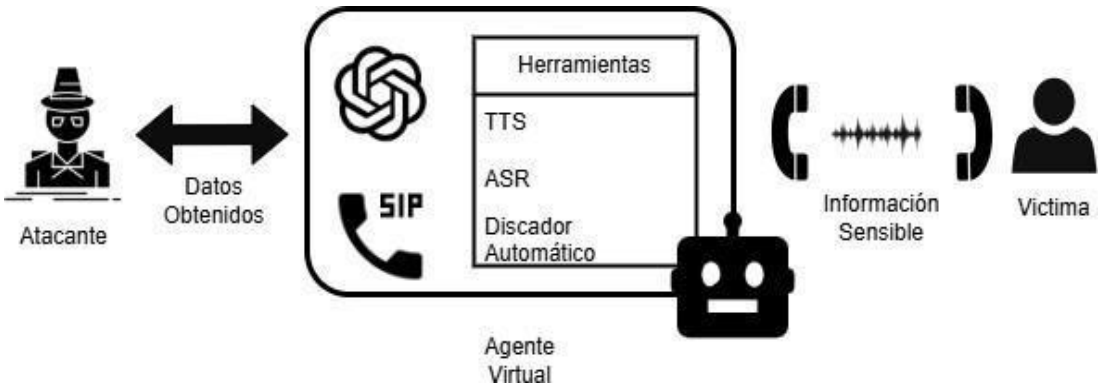


Fig. 1. Modelo de sistema vishing potenciado por AI. Fuente: Elaboración propia con herramienta Draw.io (ahora app.diagrams.net).

La figura 1 representa un modelo de discador telefónico automático cargado de registros con números e información vital para que un agente virtual o “Bot” realice su operación.

En este punto podríamos pensar que las estafas telefónicas son algo del pasado,

pero la realidad demuestra que continúan siendo vigentes, tal como lo muestra el siguiente cuadro:

Tipo de Estafa	2023	2024	Variación Interanual
Phishing	Aumento del 58,2% respecto a 2022, con casi 5 millones de ataques registrados.	Incremento del 26% en intentos bloqueados por Kaspersky, alcanzando 893 millones.	Aumento significativo impulsado por el uso de IA generativa y suplantación de marcas populares.
Vishing	Incremento del 260% en el cuarto trimestre respecto al mismo período de 2022.	Aumento del 442% en la segunda mitad del año, con ataques más sofisticados mediante IA.	Crecimiento exponencial debido a la adopción de tecnologías de voz sintética y deepfakes.

Tabla 1. Estadísticas de ataques de Phishing y Vishing. Elaboración Propia mediante datos obtenidos por Zscaler, 2024 y Kaspersky, 2024.

Los engaños se presentan de diferentes maneras evolucionando y adaptándose al tipo de víctima. Entre los casos más comunes, tenemos el ya mencionado phishing como un correo electrónico intentando suplantar la identidad de un tercero, (entidad gubernamental, banco, red social, amigos). Podemos encontrar alertas de estos

episodios cuando recibimos comunicados de instituciones o empresas informando sobre la amenaza en curso. En el año 2024 la Dirección General Impositiva de Uruguay debió indicar a sus contribuyentes sobre intentos de estafas en su nombre (Ámbito, 2024).

El phishing como el vishing conforman ataques de ingeniería social que buscan ganar la confianza de los usuarios a través de manipulación psicológica para así obtener acceso a datos personales de gran valor.

Las amenazas suelen dirigirse a grupos bien marcados de la sociedad. Por un lado se encuentran menores y jóvenes que han tenido, a temprana edad, acceso a dispositivos tecnológicos con baja supervisión adulta. En el otro extremo nos encontramos a las personas mayores, con gran alto grado de ingenuidad respecto a los peligros que pueden acechar a través de los sistemas de comunicación actuales (Spina, 2013).

Los medios tradicionales de seguridad, como los filtros antispam y otros métodos heurísticos, dependen de patrones previamente identificados para detectar amenazas. Estos sistemas clasifican el contenido basándose en reglas predefinidas o modelos probabilísticos construidos a partir de conocimiento adquirido. Consideremos, por ejemplo, un caso de correo electrónico, donde la interacción entre el atacante y la víctima ocurre de manera asincrónica, permitiendo el etiquetado del mensaje y la generación de alertas para el usuario antes de su posible exposición al fraude. En este contexto, estos sistemas pueden ser efectivos contra ataques conocidos, pero presentan vulnerabilidades ante tácticas novedosas.

Por el contrario, los modelos de inteligencia artificial aplicados a este propósito pueden adaptarse, aprender y mejorar su efectividad frente a riesgos no vistos anteriormente. Además, su capacidad de análisis en tiempo real permite su implementación en entornos de comunicación instantánea, aumentando significativamente la detección y prevención de amenazas emergentes.

La capacidad de procesar lenguaje natural (NLP sus siglas inglés) sumado la aplicación de aprendizaje automático permite no solo analizar la estructura y sintaxis de un mensaje sino además interpretar las intenciones subyacentes del atacante, pudiendo entonces accionar contramedidas de respuesta.

Khan (2025) señala que la introducción de inteligencia artificial en la detección de estafas ha logrado mejorar el proceso. En particular, el autor reporta que se ha logrado una reducción de los falsos positivos (afirmación incorrecta de la presencia de una condición, en este caso una falsa alarma) del orden del 85% comparado con los medios tradicionales mencionados. Así también se han desplegado sistemas con la capacidad de analizar más de 100.000 (cien mil) transacciones por segundo, haciendo que la detección de fraudes en tiempo real no sea solo posible sino también altamente eficiente .

Estos resultados permiten validar que la IA podría ser un “Game-Changing” (que cambia las reglas de juego por su significado en inglés) en la problemática establecida.

La empresa Google (Security Blog 2025), por su parte, ha lanzado en marzo de 2025 aplicaciones móviles destinadas a la detección de estafas en tiempo real aplicando inteligencia artificial. A través de modelos de avanzada reconocen patrones sospechosos en la lectura y escucha de las conversaciones en curso. Así entonces son capaces, mediante notificaciones, de advertirnos del posible engaño. De momento estas funciones solo se encuentran disponibles en los Estados Unidos de América, Canadá y Reino Unido.

Evitar la conversación con extraños o desconfiar de las apariencias e intenciones de estos, son algunas de las enseñanzas que las personas han adquirido en las primeras etapas de aprendizaje. Pero lo descrito anteriormente muestra la nueva cara de los mecanismos usados por los atacantes para vulnerar estas defensas. Y si los atacantes adaptan sus tácticas a los tiempos que corren y desbloquean nuevas maneras de delinquir, entonces un nuevo y verdadero problema se ha revelado. No se trata de cómo detener las estafas, sino más bien de cómo prevenirlas y combatirlas; cómo usar la tecnología disponible y ponerla al servicio de este fin.

1.2 Justificación del estudio

Nos encontramos en un punto de inflexión, la inteligencia artificial (IA) es catalogada como líder de la cuarta revolución industrial (Guerschberg, L., 2025). Esta herramienta ha automatizado tareas repetitivas y permitido la creación de sistemas autónomos capaces de aprender y tomar decisiones con mínima intervención humana. Su utilización presenta grandes oportunidades en diferentes áreas como la educación, la salud, la seguridad, la economía y hasta la forma de relacionarnos. Sin duda ha

transformado nuestras vidas, desde la forma en que nos relacionamos hasta en la manera de cómo hacer negocios. Pero también existe sobre ella un lado oscuro que ayuda a la proliferación de cibercrímenes.

Solo para adicionar ejemplos de lo expresado anteriormente tenemos que :

- En Perú ,en el año 2024, se ha contabilizado que los delitos informáticos han tenido un crecimiento del 40%.Esto es adjudicado a la aparición de nuevas metodologías impulsadas, justamente, por IA (Landio, M. ,2025).

- Según la FTC (Comisión Federal de Comercio de Estados Unidos) en el año 2024 si bien no se contabilizaron más estafas que en su año predecesor estas tuvieron un incremento en perjuicio del orden de un 25% sumando así más de 12,5 mil millones de dólares en pérdidas económicas solo en ese país (Federal Trade Commission., 2025).

Esto se replica en otros países por lo que el aumento de las estafas virtuales se ha convertido en un fenómeno internacional preocupante, afectando a individuos y economías a nivel global. En este escenario generar propuestas de solución a un problema de semejante envergadura tendrá visiblemente un impacto social importante ya que la eficiencia en la detección y prevención de estafas permitirá ahorros monetarios sustanciales tanto a individuos como empresas.

La idea de usar IA en tiempo real para detectar las amenazas, especialmente integrando tecnologías como ASR y TTS para suplantar a la víctima y que el atacante no interactúe directamente con el usuario real, es una propuesta creativa y novedosa.

No solo se está proponiendo con este proyecto una aplicación pasiva de detección, sino una respuesta activa, añadiendo así un elemento disruptivo.

1.3 Objetivos

Esta sección enlista los objetivos establecidos y a cubrir por el presente proyecto.

1.3.1 Objetivo principal:

Proponer modelo de detección de estafas telefónicas y digitales por medio de la inclusión de herramientas de inteligencia artificial con el objetivo final de hacer frente al delito reduciendo así su ejecución .

1.3.2 Objetivos secundarios:

- Describir los distintos tipos de estafas telefónicas y digitales mediante un estudio de la bibliografía existente para su identificación y clasificación.
- Describir Ingeniería Social, ASR, TTS, Voice Cloning AI y técnicas aplicadas a la detección de fraudes con el fin de determinar las herramientas de combate a las estafas telefónicas y digitales.
- Diseñar una arquitectura teórica de un sistema que, con la aplicación de inteligencia artificial, detecte estafas telefónicas y digitales.
- Generar el MVP para validar el modelo mediante experimentación y el análisis de los resultados obtenidos.

1.4 Estructura de la tesis

Es posible dividir el contenido de la tesis en cinco partes:

En la primera parte se encuentra la introducción al tema, la definición del problema, la justificación y los objetivos del trabajo .

La segunda parte conforma el marco teórico donde se definirán los conceptos generales y específicos necesarios para una correcta comprensión.

En la tercera parte se presentará el modelo de solución, su arquitectura y el alcance del prototipo funcional.

En la cuarta parte se dará cierre a la tesis entregando conclusiones y resultados obtenidos a través de la investigación y las pruebas.

Finalmente en la quinta parte mencionarán posibles líneas de investigación futuras.

1.5 Limitaciones del estudio

Este proyecto tiene como objetivo principal proponer un modelo como respuesta tecnológica a la detección temprana de estafas telefónicas y digitales

mediante el uso de inteligencia artificial. En este sentido, el alcance de la tesis se centra en el diseño conceptual de la solución, su arquitectura y su potencial funcional. Todo esto sin abordar en profundidad las incumbencias o implicaciones legales, regulatorias y éticas asociadas al uso de este tipo de tecnologías.

En el contexto argentino la privacidad de las comunicaciones se encuentra protegida por la Constitución Nacional y por diversas normativas específicas, como la Ley de Inteligencia Nacional N° 25.520, el Código Procesal Penal de la Nación y la Ley Argentina Digital N° 27.078.

Desde esta perspectiva el análisis de comunicaciones telefónicas y digitales en tiempo casi real podría interpretarse como potencialmente intrusivo. No obstante, el modelo planteado no contempla la interceptación ilegal de comunicaciones ni la grabación sin consentimiento. La solución se concibe como un mecanismo de asistencia al usuario final, activado de forma voluntaria y bajo su control, operando en escenarios de consentimiento explícito por parte del usuario.

Si bien el impacto de la legislación y los marcos normativos sobre el uso de tecnologías emergentes, como la inteligencia artificial, resultan relevantes en la práctica ,el análisis jurídico detallado, así como los procedimientos legales asociados a la intervención de comunicaciones no forman parte de los objetivos del presente trabajo.

Asimismo, profundizar en la implementación normativa específica de cada país implicaría abordar marcos legales significativamente diferentes entre jurisdicciones.

Un análisis de esa naturaleza resultaría extenso y complejo desviando el foco del trabajo, cuyo objetivo es la propuesta de una aplicación técnica generalizable e independiente de su adaptación normativa particular.

En consecuencia, no se analizan los posibles desafíos legales relacionados con la implementación de sistemas de IA para la prevención de fraudes y que puedan afectar su aplicación en distintos contextos geográficos.

Como contenido de la presentación de la solución se incluye un análisis preliminar de los costos asociados al desarrollo e implementación de la plataforma con el objetivo de reflejar la estructura mínima necesaria para su funcionamiento.

Por otra parte, como futura línea de investigación, se abrirá debate sobre los riesgos asociados a la adopción de una solución basada en inteligencia artificial, tales como la posible dependencia tecnológica, el margen de error de los modelos y el nivel de confianza que los usuarios depositen en la plataforma.

1.6 Metodología de investigación

El enfoque utilizado para el desarrollo de este trabajo se encuentra conformado por la revisión documental, análisis de fuentes relevantes sobre estafas digitales y telefónicas así como el rol de la inteligencia artificial.

Para brindar garantía sobre la rigurosidad del proceso se han seleccionado artículos científicos publicados en portales académicos, informes gubernamentales, estudios de organismos internacionales. Se priorizaron fuentes provenientes de bases

de datos como IEEE Xplore y Google Scholar así como publicaciones académicas y oficiales de entidades como la FTC o el Ministerio de Justicia de Argentina.

El criterio de selección usado fue basado en el siguiente orden de prioridad:

1. Relevancia con la temática
2. Medio de procedencia de la fuente
3. Antigüedad de la publicación.

Para el análisis se realizó una lectura crítica de la fuente seleccionada, identificando durante el proceso los patrones comunes de las estafas digitales, los sectores más vulnerables y tecnologías implicadas en la generación del delito así como en su mitigación. La posibilidad de comparar diferentes estudios permitió generar conclusiones sobre la relevancia del tema y sentar las bases para la aplicación de una posible solución.

2. MARCO TEÓRICO

Esta sección documenta el estado del arte, conceptos y definiciones relacionadas con el eje del proyecto. Es así como se formulará un listado de temas a conocer necesarios para una mejor comprensión.

2.1 Ingeniería Social

2.1.1 Definición

En el ámbito de la ciberseguridad podemos decir que la ingeniería social es considerada una de las formas más efectivas de ataque dado que vulnera uno de los puntos más sensibles de un sistema : el humano. Mitnick y Simon (2002) la definen como “el arte del engaño” y esto aplica, dado que según López Grande y Guadrón (2015) “se refiere a la aplicación de técnicas que los atacantes utilizan para engañar a un usuario autorizado de sistemas informáticos de una compañía, con el fin de obtener información sensible o provocar acciones que generan vulnerabilidades explotables”.

Fuera de cualquier asociación que inicialmente, y de manera errónea, podamos hacer con las redes sociales, este concepto no surge en el campo de la informática. Originalmente, el término fue utilizado en el siglo XIX por el empresario holandés J. C. van Marken, quien lo aplicó a la gestión de conflictos emocionales en entornos industriales. Con el tiempo, el publicista Edward L. Bernays lo resignificó como un mecanismo de manipulación de masas, argumentando que las sociedades, al carecer de principios morales firmes, debían ser “guiadas” (López Grande & Guadrón, 2015). En el ámbito de la seguridad informática fue finalmente Kevin Mitnick quien popularizó el concepto a finales del siglo XX.

Spina (2013) señala la vulnerabilidad inherente de determinados grupos sociales respecto a estos ataques de manipulación digital, destacando que niños, adolescentes, adultos mayores y personas con discapacidades se encuentran más expuestos a maniobras de manipulación digital debido a la falta de experiencia o a limitaciones

físicas y psicológicas. Esta constatación ha llevado a que la ingeniería social se considere tanto un problema tecnológico como un desafío social, pues afecta con particular fuerza a los sectores que menos recursos poseen para protegerse.

2.1.2 Mecanismos de manipulación

El éxito de la ingeniería social está altamente relacionado con la explotación de sesgos cognitivos y principios de persuasión. Cialdini (2009) describe seis principios que suelen ser aprovechados por los atacantes: reciprocidad, compromiso y coherencia, prueba social, autoridad, simpatía y escasez. Estos principios se manifiestan en múltiples escenarios, como la simulación de urgencia ("su cuenta será bloqueada si no actúa de inmediato") o la apelación a la autoridad ("habla el área de soporte técnico de su banco").

La literatura especializada identifica técnicas recurrentes como el pretexting (creación de un escenario falso para obtener datos), el baiting (se deja como carnada algún elemento que juegue con la curiosidad de la víctima), o el quid pro quo (se promete un beneficio a cambio de información) (Abu Hweidi & Eleyan 2022). En entornos digitales y móviles, estas prácticas han evolucionado hacia el phishing, smishing (fraude vía SMS) y vishing (fraude telefónico), que se caracterizan por su capacidad de llegar a un alto volumen de potenciales víctimas en poco tiempo (López Grande & Guadrón, 2015).

2.1.3 Manifestaciones principales hoy

Hoy en día es posible ver aplicada la ingeniería social en diversos tipos de ataques y como manifestaciones más relevantes podemos nombrar:

- Phishing: correos electrónicos o mensajes diseñados para obtener información personal y financiera.
- Vishing: llamadas telefónicas donde se suplanta la identidad de entidades legítimas, como bancos o instituciones estatales.
- Deepfakes: utilización de modelos de IA para crear imitaciones hiperrealistas de voces o rostros, que refuerzan la credibilidad de los ataques.

Estas amenazas serán definidas y analizadas en mayor detalle en secciones posteriores, pero resulta importante entender que las mismas se construyen con los mismos principios de manipulación interpersonal.

2.1.4 Síntesis

La ingeniería social es un ataque silencioso y persistente, su éxito depende de la explotación de vulnerabilidades humanas y no particularmente de fallas técnicas. Para su comprensión se deben integrar perspectivas sociales, psicológicas y tecnológicas, pues vincula tanto a la seguridad de sistemas como a la protección de comunidades vulnerables. El conocimiento de esta resulta indispensable para cualquier modelo de prevención de estafas telefónicas y digitales.

2.2 Tipos de amenazas: Phishing, Vishing, DeepFake

2.2.1 Phishing

2.2.1.1 Definición

Benavides et al. (2020) señalan que el phishing es una de las amenazas más extensas y persistentes en el ámbito de la ciberseguridad actual, y lo definen como "la combinación de ingeniería social y exploits técnicos, diseñados para convencer a una víctima de proporcionar información personal, generalmente con fines monetarios" . En tal sentido, se trata entonces, de una técnica cuyo objetivo es engañar a los usuarios para que revelen información sensible, normalmente credenciales de acceso, datos bancarios u otro dato que pueda ser utilizada en fraudes o delitos informáticos.

Podemos encontrar los primeros registros de este tipo de ataque en los finales de los años noventa, pero su evolución ha sido constante y se ha adaptado a los cambios en el mundo digital. En su forma más natural el phishing se lleva a cabo mediante el envío de correos electrónicos falsos que simulan provenir de instituciones legítimas (bancos, proveedores de servicios, entidades gubernamentales), en los que se solicita a la víctima hacer clic en un enlace y proporcionar sus datos en un sitio web clonado. Esta técnica continúa siendo altamente eficaz debido a la capacidad de los atacantes para diseñar pantallas que resultan no poseer diferencias de las originales.

Yeboah-Boateng y Amanor (2014) señalan que, con la proliferación de teléfonos inteligentes y la ubicuidad de la conectividad móvil, el phishing ha encontrado un terreno fértil en dispositivos portátiles, donde las limitaciones de pantalla y la confianza del usuario en interfaces simplificadas aumentan la vulnerabilidad.

2.2.1.2 Clasificación de los ataques de phishing

Los ataques de phishing pueden clasificarse de acuerdo con distintos criterios. Una primera taxonomía los diferencia según el servicio atacado: servicios financieros, pasarelas de pago en línea, redes sociales, plataformas de compraventa o subastas, servicios de almacenamiento en la nube, o incluso falsas ofertas de empleo (Benavides et al., 2020). Una segunda clasificación se basa en el modus operandi, que incluye variantes como el phishing engañoso (correos masivos que simulan provenir de una entidad conocida), el malware phishing (uso de software malicioso), el pharming (redireccionamiento mediante manipulación de DNS), el phishing por motores de búsqueda (páginas fraudulentas posicionadas en buscadores) y los ataques man-in-the-middle (hombre en el medio por sus traducción del inglés), en los cuales el atacante intercepta las comunicaciones entre el usuario y el servicio legítimo.



Fig. 2 Ciclo de vida de un ataque de phishing.Fuente: Elaboración propia con fines académicos.

El ciclo de vida de un ataque de phishing (ver figura 2) puede ser descrito en fases: planificación y configuración, ataque, infiltración, recopilación de datos y extracción (Benavides et al., 2020). Durante la planificación, el atacante identifica a la organización o individuo objetivo; en el ataque, se envían correos electrónicos

falsificados; mientras que en la infiltración, el malware o la página fraudulenta capta los datos; posteriormente se recopila la información y por último se extrae y elimina la evidencia. Esta estructura organizada demuestra que el phishing es un proceso sistemático que puede repetirse y perfeccionarse en múltiples iteraciones.

2.2.1.3 Factores de vulnerabilidad

Diversos estudios coinciden en que el phishing se aprovecha de la falta de concienciación de los usuarios. Yeboah-Boateng y Amanor (2014) señalan que más del 60% de las víctimas de fraudes digitales reconocen haber hecho clic en enlaces enviados por mensajes de texto o correo electrónico sin verificar su procedencia, mientras que un 25% actúa con indiferencia bajo la creencia errónea de que "nadie querría atacarlos". Estas percepciones demuestran que la ignorancia, el exceso de confianza y la apatía frente a la ciberseguridad constituyen factores de riesgo clave.

A estos elementos psicológicos se suman las técnicas de persuasión ya mencionadas y descritas por Cialdini (2009), tales como la autoridad y la urgencia, que son empleadas con frecuencia en los correos de phishing. Por ejemplo, al simular un mensaje del área de soporte de un banco que exige actualizar datos "de inmediato" para evitar el bloqueo de la cuenta, se explota el sesgo de obediencia a la autoridad y la toma de decisiones bajo presión.

2.2.1.4 Técnicas más comunes

Benavides et al. (2020) evidencian que las dos técnicas más empleadas en la actualidad son el phishing por correo electrónico (email spoofing) y el phishing mediante sitios web falsificados (spoof websites). En el primer caso, el atacante utiliza campañas masivas de spam que aparentan provenir de entidades legítimas; en el segundo, construye páginas visualmente idénticas a las oficiales, con variaciones mínimas en la URL (como el uso de caracteres adicionales o sustituciones de símbolos), lo que confunde al usuario.

Otros métodos incluyen la utilización de enlaces disfrazados en los que se manipula la cadena de la dirección web, como en el ejemplo de la figura 2 en donde la víctima puede creer estar recibiendo un correo legítimo de su banco, pero en realidad se pretende conectarlo con un servidor malicioso.



Fig. 3. Ejemplo ficticio de correo de phishing bancario. Fuente: Elaboración propia con fines académicos.

2.2.1.5 Motivaciones e impacto

El phishing tiene como motor motivos económicos. Los atacantes buscan obtener beneficios financieros directos (transferencias, compras, desvío de fondos) o indirectos, mediante la venta de identidades robadas en mercados negros digitales. También existen motivaciones secundarias, como la notoriedad dentro de comunidades de ciberdelincuentes o el espionaje corporativo.

El impacto de estos ataques es significativo. Según el Anti-Phishing Working Group (citado en Benavides et al., 2020), en un solo año se reportaron decenas de millones de intentos de phishing, con pérdidas financieras que ascienden a miles de millones de dólares a nivel mundial. Incluso organizaciones de alta reputación en ciberseguridad han sido blanco de ataques exitosos, como el caso de RSA en 2011, lo que demuestra que no solo los usuarios finales son vulnerables, sino también empresas con infraestructura robusta.

2.2.1.6 Estrategias de mitigación

Benavides et al. (2020) señalan que “la detección temprana de ataques de phishing de tipo zero day solo puede lograrse mediante enfoques de aprendizaje automático, ya que las listas negras requieren la previa identificación del ataque”.

Los métodos de mitigación del phishing abarcan desde medidas tradicionales hasta técnicas avanzadas de inteligencia artificial. Dentro de lo clásico, se destacan las listas negras y blancas, aunque estas sean ineficaces frente a ataques “zero day” (vulnerabilidad,ataque nuevo o no registrado). Por ello, la tendencia actual se orienta

hacia algoritmos de Machine Learning y Deep Learning, capaces de detectar patrones anómalos en URLs, correos electrónicos y características de sitios web.

Yeboah-Boateng y Amanor (2014) concluyen que la educación en ciberseguridad, complementada con simulaciones de ataques y entrenamientos prácticos, tiene un impacto positivo en la reducción de la vulnerabilidad. Sin embargo, destacan que los programas de advertencia deben diseñarse de manera atractiva, pues la mayoría de los usuarios tienden a ignorar mensajes extensos o demasiado técnicos. Esto dictamina que la concienciación de los usuarios sigue siendo una estrategia crucial.

2.2.1.7 Síntesis

Como resumen el phishing constituye una amenaza global persistente que combina manipulación psicológica y recursos técnicos para engañar a los usuarios. Su éxito depende tanto de la sofisticación de los atacantes como de la falta de cultura de seguridad de los usuarios. La clasificación de sus variantes, el análisis de sus motivaciones y la evaluación de sus impactos permiten entender la magnitud del problema. La integración de técnicas de inteligencia artificial para su detección temprana y la capacitación de los usuarios son, al menos hasta hoy, las bases de la lucha contra estas estafas.

Además de lo definido cabe comentar que a partir de la técnica de phishing, han surgido otras variantes como el SMiShing (ataques a través de mensajes de texto) o el spear-phishing (altamente dirigido a un individuo u organización). Sin embargo, en

esta investigación el análisis se centrará en el phishing tradicional, el vishing y las técnicas de manipulación basadas en deep fakes, dado que hoy representan las amenazas más relevantes.

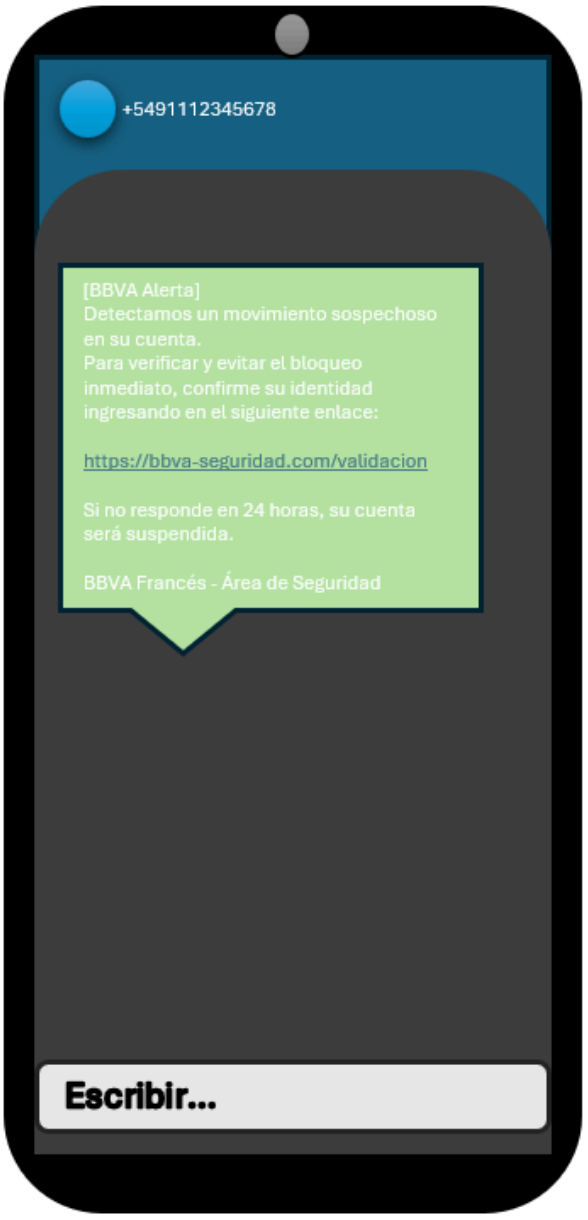


Fig. 4 Ejemplo de ataque de Smishing. Fuente: Elaboración propia con fines académicos.

2.2.2 Vishing

2.2.2.1 Definición

El vishing, unión de las palabras voice y phishing, es un tipo de estafa que hace uso de aplicación ingeniería social combinado con sistemas de comunicación telefónica para obtener información confidencial de las víctimas. Este ataque puede considerarse como una evolución del phishing tradicional, los ciberdelincuentes suplantan a instituciones legítimas, como bancos, aseguradoras, compañías de servicios públicos o incluso organismos gubernamentales realizando llamadas de voz.

Realizado el contacto, los atacantes persuaden a los usuarios para que otorguen datos sensibles, tales como contraseñas, números de tarjeta de crédito o códigos de verificación. En definición de Yeboah-Boateng y Amanor (2014), “el vishing es un ataque de ingeniería social en el que se recibe una llamada de voz que induce al receptor a proporcionar información personal con la intención de usarla de forma fraudulenta”.

2.2.2.2 Evolución y características

A diferencia del phishing por correo electrónico, el vishing hace abuso de la confianza históricamente depositada en la voz y en la telefonía como medio de comunicación. Esto es porque tradicionalmente, una llamada era percibida como un canal confiable y más difícil de falsificar que un mensaje electrónico; no obstante, el desarrollo de la telefonía por Internet (Voice over IP – VoIP) y las técnicas de caller ID spoofing (entendido como la falsificación o suplantación deliberada del número telefónico que se muestra en el dispositivo receptor) han permitido que sea posible

manipular la identidad visualizada en el dispositivo de la víctima. De este modo, las comunicaciones pueden parecer provenir de un número legítimo, lo que otorga al fraude un grado de verosimilitud difícil de cuestionar (Maseno, 2017).

El crecimiento de esta práctica se ha visto acelerada debido a dos factores principales. En primer lugar, la masificación de los servicios financieros móviles, ya que hoy el teléfono celular es uno de los canales principales para realizar transacciones y esto convierte a los usuarios en un blanco atractivo para los atacantes. En segundo lugar, la utilización de tecnologías automatizadas, como los robocalls, que permiten lanzar miles de llamadas de forma masiva con mensajes pregrabados, incrementando exponencialmente el alcance del ataque.

2.2.2.3 Ejemplos comunes de vishing

A continuación se listan algunos de los ejemplos de aplicación más comunes:

- Suplantación bancaria: el estafador se hace pasar por un representante de una entidad financiera que informa sobre un supuesto problema de seguridad en la cuenta y solicita a la víctima confirmar datos confidenciales.

- Soporte técnico falso: llamadas que advierten de supuestos virus o problemas en un dispositivo, ofreciendo asistencia a cambio de credenciales o acceso remoto.

- Fraudes con autoridades: delincuentes que simulan ser miembros de la policía, la fiscalía o la agencia tributaria, alegando irregularidades legales que deben solucionarse de inmediato para evitar sanciones.

- Premios o beneficios ficticios: notificación de supuestos premios de loterías o programas de fidelidad, para cuyo cobro la víctima debe aportar datos bancarios.

El vishing explota la autoridad, la urgencia y el miedo, principios psicológicos descritos previamente por Cialdini, que predisponen a las víctimas a obedecer sin cuestionar (Ver figura 4).

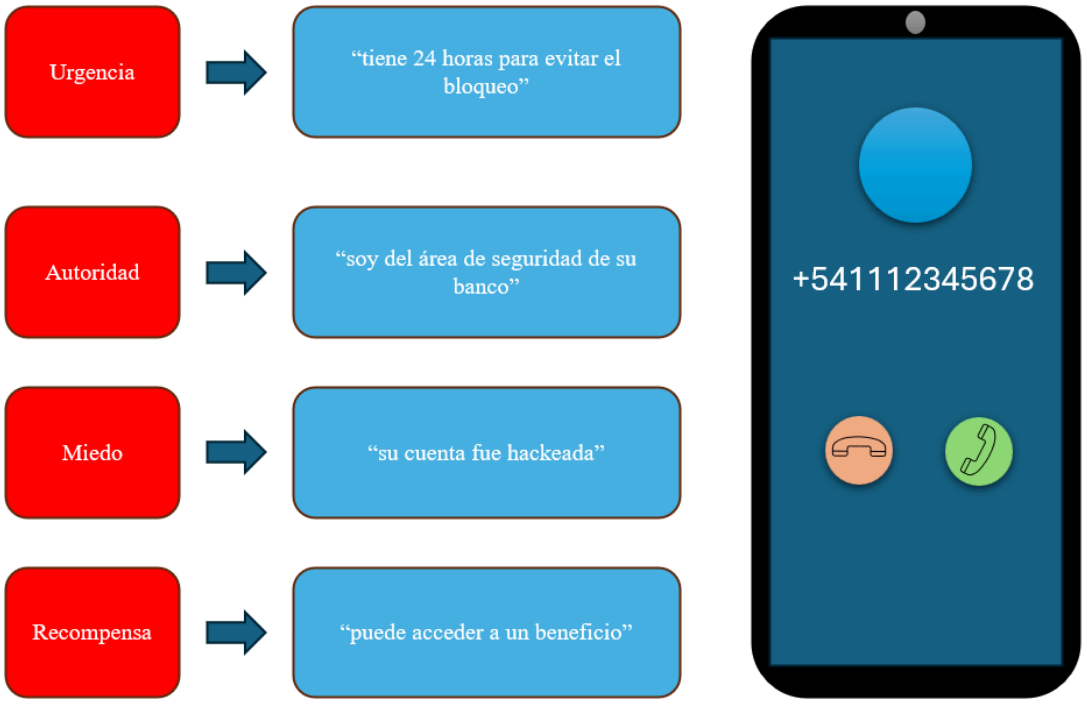


Fig. 5 Esquema de ataque de vishing. Fuente: Elaboración propia con fines académicos.

2.2.2.4 Impacto económico y social

El impacto del vishing es considerable. En su investigación sobre las pérdidas financieras derivadas del phishing y vishing, Bello et al. (2025) señalan que el costo económico de estas prácticas se mide en miles de millones de dólares anuales, afectando no solo a individuos sino también a instituciones financieras y compañías de telecomunicaciones. A ello se suma el impacto social: muchas víctimas, tras caer en un fraude de este tipo, no denuncian el hecho por vergüenza o por considerarlo un error personal, lo que dificulta cuantificar la magnitud real del problema (Maseno, 2017).

2.2.2.5 Factores de vulnerabilidad

Se evidencia que los grupos más vulnerables son los adultos mayores y los usuarios con bajo nivel de alfabetización digital, quienes tienden a confiar más en la voz como medio de validación de identidad (Yeboah-Boateng & Amanor, 2014). Esto genera señales de que el vishing suele tomar provecho de colectivos con menor capacidad de defensa frente a las amenazas tecnológicas.

A continuación se nombran algunos de los los factores que favorecen el éxito del vishing:

- Limitaciones tecnológicas: los sistemas de identificación de llamadas son fácilmente manipulables mediante spoofing.

- Complejidad de los dispositivos móviles: la sobrecarga de funciones y notificaciones reduce la capacidad crítica del usuario para detectar anomalías.

- Falta de educación en ciberseguridad: la mayoría de los usuarios desconoce cómo verificar la autenticidad de una llamada sospechosa.

- Carga emocional: el uso de miedo, presión de tiempo o incentivos económicos inmediatos genera un estado de vulnerabilidad psicológica.

Estos factores, crean un entorno propicio para que los atacantes logren ejecutar su treta.

2.2.2.6 Tendencias actuales

El vishing ha evolucionado más allá de las simples llamadas fraudulentas. Brijith (2023) destaca que los atacantes incorporan inteligencia artificial y técnicas de clonación de voz para imitar la entonación y estilo de habla de personas conocidas por la víctima, como familiares o representantes de instituciones. Este fenómeno, vinculado al auge de los deep fakes de voz, plantea un nuevo desafío para la detección, ya que incrementa el realismo de los ataques.

Otra cosa que se observa es una creciente tendencia en la elaboración de ataques, donde el vishing se combina con otras técnicas como el phishing o el smishing. Por ejemplo, un usuario puede recibir un correo electrónico fraudulento con información parcial, seguido de una llamada telefónica en la que el atacante refuerza la veracidad del engaño. Este engaño omnicanal multiplica permite aumentar su efectividad, pues la víctima percibe una coherencia entre los diferentes canales de comunicación.

2.2.2.7 Problemas y retos

El principal problema en la mitigación del vishing radica en la dificultad de identificar en tiempo real la legitimidad de una llamada. Mientras que en el phishing web pueden aplicarse filtros automáticos y listas negras de URLs, en el vishing el contacto humano introduce un nivel de imprevisibilidad mayor. Además, la globalización de las telecomunicaciones y el uso de VoIP hacen complejo rastrear la ubicación real del atacante, lo que dificulta la acción legal.

Por otra lado, al no denunciarse la mayoría de los fraudes, las estadísticas oficiales no reflejan la magnitud real del fenómeno, lo que a su vez limita el diseño de políticas públicas de prevención y la inversión en soluciones tecnológicas.

2.2.2.8 Síntesis

El vishing se muestra como una amenaza en rápido crecimiento dentro del mundo de los fraudes digitales. Su efectividad se sustenta en el uso de la voz como herramienta de confianza, la explotación de debilidades psicológicas y la adopción de

tecnologías avanzadas como VoIP y clonación de voz. Aunque su origen se vincula al phishing tradicional, es un ataque autónomo con impacto económico y social significativo. La dificultad de detección, la manipulación de identidades y la falta de denuncia hacen del vishing un fenómeno complejo que requiere un abordaje que provenga de diferentes perspectivas y disciplinas. Comprender cómo opera es importante para el desarrollo de modelos de detección temprana que se discutirán en capítulos posteriores.

2.2.3 DeepFake

2.2.3.1 Definición

El término deepfake proviene de la combinación de deep learning (aprendizaje profundo por su traducción del inglés) y fake (falso por su traducción del inglés), y hace referencia a contenidos sintéticos (imágenes, videos o audios) generados mediante algoritmos de aprendizaje profundo capaces de imitar con gran realismo la apariencia o la voz de una persona (Ver Figura 6). Estos contenidos son elaborados, principalmente, a través de Redes Generativas Antagónicas (GANs), en las cuales dos redes neuronales —un generador y un discriminador— compiten entre sí hasta producir resultados prácticamente indistinguibles de la realidad (Westerlund, 2019).

La sofisticación de esta técnica ha crecido a un ritmo acelerado desde su primera aparición pública en 2017 en la plataforma Reddit. Hoy, un usuario con una computadora estándar y acceso a bibliotecas de código abierto puede producir videos de gran realismo, incluso en tiempo real. Por ejemplo dentro de Tiktok (s. f.), una de

las más conocidas redes sociales en la actualidad, encontramos filtros disponibles que nos permiten generar imágenes o videos cambiando nuestro rostro por el de una celebridad. Esa imagen se ajusta casi a la perfección a nuestros movimientos y gestos

Este avance plantea serias preocupaciones sobre la autenticidad del contenido digital y su impacto en la confianza social.

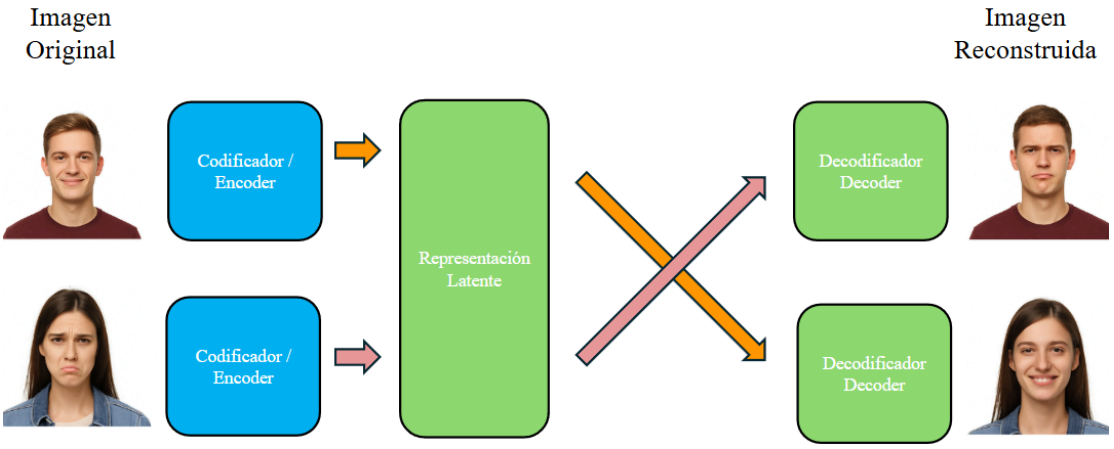


Fig. 6 Transformación rostro por deep fake . Fuente : Elaboración propia con asistencia de inteligencia artificial (Chat GPT, 2025), basada en Noriega (s.f.) y Zucconi (2018).

2.2.3.2 El DeepFake y la posverdad

El fenómeno de los deepfakes se inserta en lo que varios autores denominan la era de la post verdad, caracterizada por la prevalencia de narrativas emocionales por sobre los hechos objetivos. Bañuelos Capistrán (2020) señala que el deepfake constituye "una amenaza para los discursos del hacer creer en tiempos de la posverdad", pues desestabiliza la relación entre imagen y verdad, erosionando la credibilidad de los documentos audiovisuales como prueba de hechos.

En este ámbito los deepfakes no son solamente herramientas técnicas, sino también entes socioculturales que facilitan la diseminación de propaganda política, la desinformación mediática y la manipulación de masas. Al insertarse en un ecosistema digital saturado de fake news (noticias falsas por su traducción del inglés), estas entidades agravan el problema al proporcionar realismo audiovisual a narrativas falsas.

2.2.3.4 Clasificación

Los deepfakes se manifiestan en múltiples ámbitos, que van desde lo recreativo hasta lo delictivo:

- Política y desinformación: Según Awah Buo (2020), casos como el video manipulado de Nancy Pelosi, ralentizado para simular ebriedad, o el deepfake del ex presidente Barack Obama creado por Jordan Peele en 2018 para alertar sobre la amenaza de esta tecnología, ejemplifican su uso en campañas de manipulación y propaganda.

- Fraude corporativo y financiero: informes recientes evidencian el uso de voces clonadas para suplantar a ejecutivos de alto rango y ordenar transferencias millonarias. En 2020, un banco en Hong Kong transfirió 35 millones de dólares tras recibir órdenes falsas respaldadas por un deepfake de voz (FS-ISAC, 2024).

- Pornografía no consensuada: Bañuelos (2020), a partir del reporte de Deeptrace, señala que más del 90% de los deepfakes detectados en línea corresponden a material pornográfico en el que se insertan rostros de celebridades o personas privadas sin su consentimiento, lo que plantea graves problemas de acoso y violencia de género.

- Entretenimiento y cultura digital: existen aplicaciones legítimas en la industria del cine, la publicidad y el doblaje de voz, donde el deepfake se utiliza para recrear actores fallecidos, traducir expresiones faciales a otros idiomas o generar efectos especiales de bajo costo (Westerlund, 2019) (Ver figura 7).



Fig.7 Deepfake de Tom Cruise. (Imagen: Chris Ume)

Esta variedad de usos confirma que el deepfake es un arma de doble filo: puede permitir la proliferación de innovaciones positivas, pero también constituye una herramienta poderosa para el fraude y la manipulación.

2.2.3.5 Problemas y riesgos asociados

Los riesgos de los deepfakes abarcan dimensiones técnicas, sociales, políticas y legales. En el terreno de la ciberseguridad, se ha demostrado que muestras sintéticas de voz pueden emplearse para vulnerar sistemas de autenticación biométrica basados en reconocimiento del hablante, evidenciando la viabilidad práctica de ataques de suplantación sonora en entornos reales (Kassis & Hengartner, 2021).

Awah Buo (2020) advierte que los deepfakes "pueden alterar la presentación de evidencias en juicios, forzando a los tribunales a dedicar tiempo y recursos adicionales para verificar la autenticidad de un material audiovisual". Esto genera retrasos en los procesos judiciales y mina la confianza en las instituciones, siendo así ,en el plano jurídico, uno de los mayores desafíos radica en la manipulación de pruebas.

Desde el punto de vista sociopolítico, los deepfakes facilitan campañas de desinformación que socavan la democracia y pueden generar crisis diplomáticas. Ejemplo de ello fue la difusión de un deepfake en el que un funcionario chino publicó la imagen manipulada de un soldado australiano cometiendo atrocidades, lo que provocó un conflicto diplomático entre ambos países (Awah Buo, 2020). Aquí, en Argentina, durante las elecciones legislativas 2025 de la Ciudad Autónoma de Buenos Aires se encontró de manera viralizada un video falso donde se veía y escuchaba al máximo referente de un partido anunciando el retiro de su primera candidata (Infobae, 2025).

Finalmente, a nivel cultural y ético, los deepfakes contribuyen a la llamada apatía informativa o “reality apathy”, donde las personas comienzan a dudar de todo contenido videográfico o sonoro, llegando a desestimar incluso pruebas auténticas bajo el argumento de que “pueden ser falsas” (Westerlund, 2019). Este fenómeno representa quizás la consecuencia más disruptiva: la erosión generalizada de la confianza en la información digital.

2.2.3.6 Síntesis

El deepfake, como expresión de la inteligencia artificial aplicada a la manipulación audiovisual, es hoy una de las amenazas más complejas en el ámbito de la seguridad digital y social. Posee capacidad para alterar la percepción de la realidad afectando a individuos, empresas e instituciones públicas. Más allá de que existen aplicaciones bien intencionadas, los riesgos asociados a la desinformación, el fraude financiero, el acoso y la pérdida de la confianza pública lo convierten en un factor central a considerar en cualquier estrategia de detección temprana de estafas telefónicas y digitales. Este apartado ha mostrado que la problemática del deepfake trasciende lo técnico, situándose en la intersección de lo cultural, lo político y lo ético, y plantea un desafío urgente para el diseño de soluciones basadas en inteligencia artificial que permitan mitigar sus efectos.

2.2.4 Análisis de transición

El análisis realizado sobre los diferentes tipos de amenazas ,phishing, vishing y deep fakes, permite observar un punto en común fuera de las diferencias técnicas inherentes de cada una: todas ellas se basan en la combinación de ingeniería social y tecnologías digitales emergentes para manipular la confianza de los usuarios y generar escenarios fraudulentos.

En primer lugar, el phishing demuestra cómo la manipulación escrita y visual, apoyada en correos electrónicos, sitios web y mensajes falsificados, puede llevar a la víctima a entregar voluntariamente sus credenciales de acceso o información sensible. El éxito depende de cuán verosímil es la imitación de entornos familiares y del aprovechamiento de sesgos cognitivos como la urgencia o la obediencia a la autoridad.

En segundo término, el vishing traslada la misma lógica de engaño al terreno de la voz, un canal que solía brindar cierto nivel de confianza. El uso de llamadas telefónicas, robocalls y técnicas de "caller ID spoofing" muestran que la ingeniería social se adapta a los medios más inmediatos y comunes para los usuarios. Aquí, la presión emocional y el contacto directo se convierten en factores decisivos que aumentan la vulnerabilidad, especialmente entre adultos mayores y sectores con menor alfabetización digital.

Por último, los deepfakes representan la evolución más sofisticada de estas amenazas. Gracias a la inteligencia artificial, es posible generar audios, imágenes y videos hiperrealistas que no solo engañan a los usuarios, sino que además distorsiona la noción de lo real y no real en el contenido digital. Cuando el phishing y el vishing explotan canales de comunicación específicos, los deepfakes atacan la base cultural y

social sobre la cual se sustenta la credibilidad a través de la evidencia audiovisual, lo que incrementa su potencial disruptivo en ámbitos políticos, financieros y jurídicos.

En conjunto, estas amenazas constituyen una tipología híbrida de fraudes digitales, caracterizada por tres elementos comunes:

- La explotación del factor humano como punto de entrada.
- El uso de tecnologías en constante evolución (correos electrónicos, VoIP, algoritmos de IA).
- El impacto transversal en múltiples dimensiones: económica, social, política y cultural.

Comprender esta interrelación resulta fundamental para el desarrollo de modelos de detección temprana. En concordancia, el estudio de las tecnologías de soporte como el reconocimiento automático del habla (ASR), la conversión texto a voz (TTS) y la clonación de voz permitirá, en los apartados siguientes, examinar tanto las oportunidades como los riesgos que ofrecen estas herramientas en el combate contra las estafas telefónicas y digitales.

2.3 TTS y ASR

2.3.1 Text-to-Speech (TTS)

2.3.1.1 Definición

El Text-to-Speech (TTS) es una tecnología que transforma texto escrito en habla sintética mediante sistemas computacionales. A diferencia de los sistemas de respuesta vocal, que se limitan a reproducir frases pregrabadas o audios estáticos, un sintetizador TTS debe ser capaz de producir en tiempo real oraciones nuevas a partir de texto particular. Dutoit (1997) subraya que el TTS se basa en dos grandes componentes: el procesamiento de lenguaje natural (NLP sus siglas en inglés), encargado de pasar el texto a una representación fonética y prosódica, y el procesamiento digital de señales (DSP sus siglas en inglés), responsable de convertirla en una voz audible (Ver figura 8). Esta característica lo diferencia de "máquinas parlantes" de propósito limitado, permitiendo aplicaciones en telecomunicaciones, accesibilidad y educación.

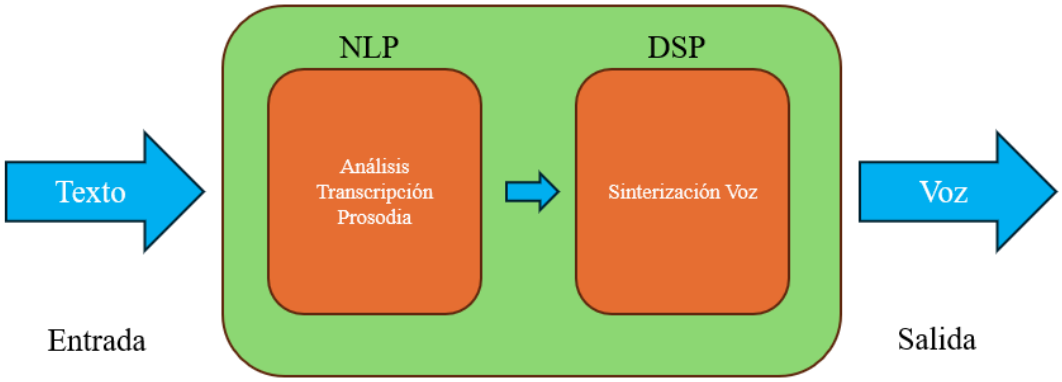


Fig. 8 Diagrama general de un sistema TTS Fuente: Elaboración propia con fines académicos.

2.3.1.2 Evolución histórica

La construcción de voz sintética ha pasado por varias etapas tecnológicas (Ver figura 9). Los primeros intentos se basaron en sistemas de reglas y modelos de formas, útiles en investigación fonética pero con un color robótico y artificial. Luego aparecieron los modelos de concatenación, que unían partes pregrabadas (diphones o triphones) para generar oraciones más fluidas. Esta técnica mejoró la naturalidad pero dependía de bases de datos de gran tamaño y resultaba rígida ante variaciones de prosodia y contexto.

Después se aplicaron enfoques paramétricos basados en modelos estadísticos, como los Hidden Markov Models (HMM), que ofrecieron mayor flexibilidad y menor dependencia de inventarios pregrabados. Sin embargo, la disrupción llegó con el aprendizaje profundo, que permitió entrenar redes neuronales capaces de mapear directamente secuencias de texto a representaciones acústicas. Ahmad y Rashid (2024) destacan esta transición como el paso de la síntesis concatenativa a sistemas end-to-end basados en deep learning, caracterizados por mayor realismo, expresividad y capacidad multilingüe.

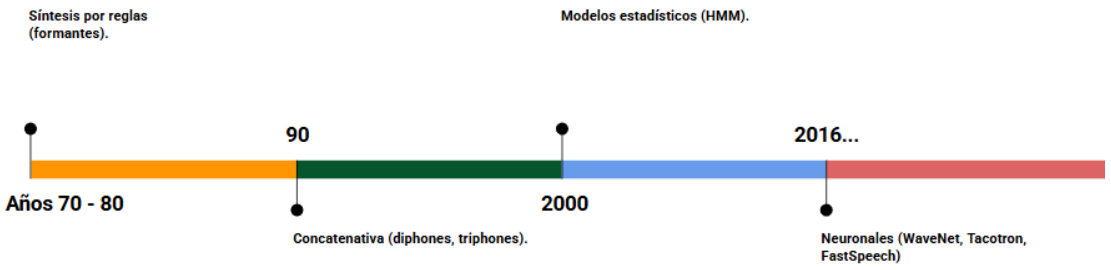


Fig 9. Línea de tiempo de la evolución TTS. Fuente: Elaboración propia con fines académicos.

Arquitectura	Ventajas	Limitaciones
Concatenativa	Simple implementación	Rigidez, poca variación prosódica. Depende de la base de grabaciones.
HMM (Paramétrica)	Flexibilidad, menor dependencia de grabaciones	Calidad limitada, voz no tan humana.
Neuronales (WaveNet,Tacotron, etc)	Naturalidad casi humana, multilingüe y expresiva	Requiere Big Data y alto poder de procesamiento

Tabla 2 Comparación Arquitectura TTS. Fuente: Elaboración propia con fines académicos.

2.3.1.3 Arquitecturas y modelos principales

Los sistemas TTS modernos abarcan diversas arquitecturas. WaveNet (DeepMind) es un modelo autorregresivo basado en convoluciones profundas, entendidas como mecanismos que permiten a la red neuronal analizar patrones locales

de la señal de audio, lo que representó un salto cualitativo en naturalidad, aunque con un elevado costo computacional.

Tacotron y Tacotron 2 (Google) son modelos end-to-end que convierten texto en espectrogramas, es decir, representaciones del sonido que describen cómo evolucionan las distintas frecuencias a lo largo del tiempo. Estos luego son transformados en audio por vocoders (codificador de voz) neuronales, logrando prosodia más expresiva.

Deep Voice (Baidu) propone una arquitectura modular que integra diferentes etapas (análisis de texto, duración, frecuencia fundamental y síntesis). FastSpeech, por su parte, es un modelo optimizado que reduce el tiempo de inferencia y ofrece mayor control sobre la prosodia.

Por último, los modelos híbridos y basados en transformers (cuyo funcionamiento se abordará en secciones posteriores) incorporan mecanismos de autoatención para mejorar el manejo de dependencias largas y la expresividad multilingüe.

Ahmad y Rashid (2024) presentan una taxonomía que agrupa estos sistemas en cuatro categorías: tradicionales basados en reglas, estadísticos, neuronales end-to-end e híbridos. Cada uno se evalúa en términos de calidad, eficiencia y requerimientos de datos, destacando que los modelos neuronales lideran en naturalidad pero demandan grandes volúmenes de datos y cómputo.

2.3.1.4 Calidad TTS

La calidad de los sistemas TTS se mide habitualmente mediante un conjunto de métricas subjetivas como objetivas. Entre los indicadores claves utilizados se encuentran:

- MOS (Mean Opinion Score): evaluación perceptiva donde los oyentes califican la naturalidad de la voz en una escala de 1 a 5.

- WER (Word Error Rate): porcentaje de errores de palabra en la salida de voz frente al texto original, usado para evaluar inteligibilidad.

- MCD (Mel-Cepstral Distortion): medida objetiva que compara características espectrales de la voz sintetizada con las de una voz real.

Dida et al. (2023) enfatizan que el uso combinado de estas métricas permiten evaluar de manera integral tanto la precisión como la naturalidad de los sistemas de TTS modernos. La creciente cantidad de datos generados por diversas fuentes, como grabaciones de voz, documentos de texto y redes sociales, proporciona una rica fuente de información que puede utilizarse para entrenar modelos. En este sentido,la integración de enfoques basados en Big Data contribuyen a mejorar de forma significativa los puntajes de naturalidad percibida (MOS), reduciendo al mismo tiempo las tasas de error de reconocimiento (WER) y las distorsiones acústicas (MCD).

2.3.1.5 Limitaciones y desafíos

A pesar de los avances, persisten retos importantes. Dutoit (1997) señalaba problemas estructurales como la dificultad de modelar adecuadamente la coarticulación y la prosodia en distintos contextos sintácticos. Décadas después, Ahmad y Rashid (2024) subrayan que los sistemas neuronales aún enfrentan desafíos como:

- Dependencia de datos masivos anotados: se requiere base de grabaciones amplias, costosas de preparar y con frecuencia limitadas a ciertos idiomas.

- Expresividad emocional y prosódica limitada: las voces suelen sonar neutras, con dificultad para transmitir matices y acentos propios de cada región.

- Costos computacionales elevados: el entrenamiento e inferencia de modelos como WaveNet o Tacotron requieren gran capacidad de procesamiento.

- Riesgo de sesgos: al entrenar con conjuntos de datos no balanceados, los modelos pueden reproducir sesgos lingüísticos, sociales o de género.

Dida et al. (2023) añaden que, aunque Big Data amplía la diversidad de datos y Chat GPT mejora la calidad del TTS, esto introduce nuevas dificultades: la gestión de volúmenes masivos de información, la necesidad de arquitecturas distribuidas y los riesgos de sobreajuste si no se controla la curación de datos.

2.3.1.6 Aplicaciones actuales

El TTS se ha consolidado en múltiples áreas:

- Accesibilidad: facilita el acceso a la información escrita a personas con discapacidad visual o con impedimentos del habla, y es ampliamente usado en lectores de pantalla.

- Asistentes virtuales y chatbots: sistemas como Siri, Alexa y Google Assistant dependen del TTS para interactuar fluidamente con los usuarios.

- Educación y aprendizaje de idiomas: utilizado en plataformas de enseñanza digital para reforzar la pronunciación y comprensión oral.

- Call centers y servicios automatizados: permite escalar interacciones de atención al cliente de manera más eficiente.

- Entretenimiento: aplicado en audiolibros, doblaje automático, videojuegos y herramientas de creación de contenido multimedia.

Dutoit (1997) anticipaba aplicaciones en telecomunicaciones, accesibilidad y educación, muchas de las cuales hoy constituyen usos consolidados.

2.3.1.7 Riesgos y usos maliciosos

En paralelo, el TTS plantea riesgos relevantes en seguridad digital. La clonación de voz (tema que será tratado en otro apartado) con fines fraudulentos se ha convertido en un problema emergente, ya que permite suplantar a directivos o familiares en llamadas telefónicas, facilitando estafas financieras (vishing). Así también, la combinación de TTS con modelos generativos ha impulsado la proliferación de deepfakes de audio, que pueden emplearse en campañas de desinformación o en fraudes a gran escala. Estos riesgos comprometen la confianza en la voz como medio de verificación de identidad y pone en peligro su uso como factor biométrico.

2.3.1.8 Síntesis

El Text-to-Speech ha pasado de ser un campo experimental con voces robóticas a convertirse en una tecnología de inteligencia artificial madura y de alto impacto social. Sus aplicaciones en accesibilidad, educación y comunicación son destacables pero su potencial uso malicioso se convierte en tema crítico para la ciberseguridad y la detección de fraudes. Comprender sus fundamentos técnicos y sus limitaciones es indispensable para analizar cómo la inteligencia artificial puede emplearse en la detección temprana de estafas telefónicas y digitales.

2.3.2 Automatic Speech Recognition (ASR)

2.3.2.1 Definición

El Reconocimiento Automático del Habla (ASR, por sus siglas en inglés) es la tecnología que permite convertir una señal de voz en texto mediante algoritmos computacionales. Anusuya y Katti (2009) lo definen como "el proceso de convertir una señal de habla en una secuencia de palabras, a través de un algoritmo implementado en un programa de computadora". Así también consideran al procesamiento del lenguaje como unas de las áreas más excitantes en la comunicación humano-máquina, constituyendo las bases tecnológicas y fundamentales de la interacción natural entre personas y sistemas digitales.

El objetivo central del ASR ha sido desde sus orígenes lograr que las máquinas puedan comprender y transcribir el habla humana en condiciones reales, lo que implica enfrentar variaciones de hablantes, acentos, estilos de habla y entornos acústicos. A diferencia de la síntesis de voz (TTS), que transforma texto en audio, el ASR se observa como el proceso inverso: toma una señal acústica y produce una representación lingüística (Ver figura 10).

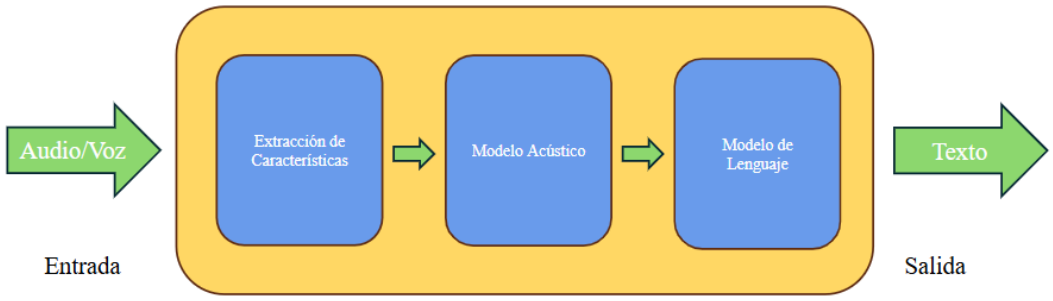


Fig. 10 Esquema básico de un sistema ASR. Fuente: Elaboración propia con fines académicos.

2.3.2.2 Evolución histórica

El desarrollo del ASR puede dividirse en varias etapas (Ver figura 11). En sus inicios, durante las décadas de 1950 y 1960, se exploraron los enfoques acústico-fonéticos, que intentaban identificar fonemas a partir de propiedades espectrales estables. Estas aproximaciones tenían su base en la idea de que cada sonido del habla podía describirse mediante características acústicas únicas, aunque en la práctica la coarticulación y la variabilidad inter-hablante dificultaron su aplicación masiva.

En los años 70 y 80 se consolidaron los enfoques de reconocimiento de patrones, que incluyeron técnicas como la Dynamic Time Warping (DTW) y los Hidden Markov Models (HMM). Estos métodos permitieron modelar estadísticamente la variabilidad temporal del habla y se convirtieron en el estándar de la industria durante varias décadas. Paralelamente, se desarrollaron técnicas de Vector Quantization (VQ) y primeros intentos de modelos conexionistas con redes neuronales, aunque estas últimas no alcanzaron madurez hasta mucho después.

En los años 90 y 2000, el ASR avanzó hacia sistemas de gran vocabulario y voz continua, integrando modelos acústicos basados en HMM con modelos de lenguaje estadísticos (bigramas, trigramas) y decodificadores de búsqueda eficientes. La financiación de proyectos como el programa DARPA en Estados Unidos permitió el desarrollo de sistemas como SPHINX (Carnegie Mellon) y BYBLOS (BBN), que marcaron hitos en transcripción de voz en contextos complejos.

Finalmente, a partir de 2010 y con mayor fuerza después de 2016, se produjo la transición hacia los modelos basados en aprendizaje profundo, que sustituyeron progresivamente a los enfoques híbridos HMM-DNN. En la actualidad, los sistemas end-to-end (E2E) integran todas las etapas del pipeline en un solo modelo neuronal, alcanzando tasas de error sustancialmente menores.

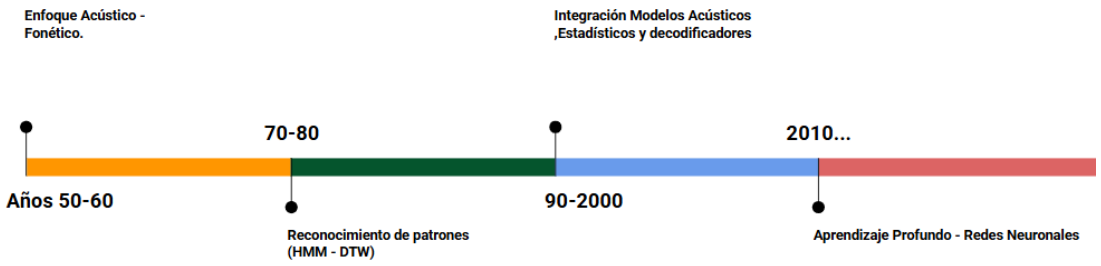


Fig 11. Línea de tiempo de la evolución ASR. Fuente: Elaboración propia con fines académicos.

2.3.2.3 Enfoques End-to-End

El gran cambio del ASR actual es el desarrollo de modelos end-to-end, que reemplazan la arquitectura modular tradicional (extracción de características, modelo acústico, modelo de lenguaje y decodificador) por una red neuronal única. Alharbi et al. (2021) destaca que la integración de redes neuronales profundas (DNN), y estos modelos junto con las técnicas de attention han redefinido el paradigma del reconocimiento del habla. Según Prabhavalkar et al. (2023), estos modelos se agrupan en tres principales enfoques:

- CTC (Connectionist Temporal Classification): introduce símbolos en blanco para manejar secuencias de distinta longitud y permite mapear directamente audio a texto.

- RNN-T (Recurrent Neural Network Transducer): extiende CTC al incorporar contexto, integrando predicción de símbolos y dependencias temporales.

- Modelos con atención (Attention-based Encoder-Decoder): utilizan mecanismos de atención global para relacionar fragmentos de audio con segmentos de texto, mejorando la fluidez en transcripción de habla continua.

Estos enfoques han demostrado reducciones de más del 50% en la tasa de error de palabras (WER) en comparación con los sistemas híbridos tradicionales. Además, la integración de grandes conjuntos de datos multilingües y técnicas de "data augmentation" ha permitido que los modelos E2E sean más robustos frente a acentos, ruido y variaciones espontáneas del habla.

2.3.2.4 Desafíos

Pese a los avances, el ASR enfrentó desafíos desde sus primeras etapas. Anusuya y Katti (2009) destacaron problemas estructurales como:

- Variabilidad entre hablantes: diferencias de acento, género, edad y estado físico.

- Condiciones acústicas adversas: ruido ambiental, reverberación y distorsiones de canal.

- Estilo de habla: desde voz pausada y leída hasta conversación espontánea con titubeos y autocorrecciones.

- Tamaño y cobertura del vocabulario: los sistemas deben adaptarse tanto a dominios restringidos (ej. dictado médico) como a dominios abiertos (transcripción de noticias o redes sociales).

Los modelos E2E mitigaron parcialmente estos problemas, pero requieren grandes cantidades de datos y potencia de cómputo significativa, lo que limita su adopción en entornos con menos recursos. Además, la transparencia de estos sistemas plantea un desafío: al ser cajas negras neuronales, es más difícil analizar los errores o garantizar explicabilidad en aplicaciones críticas.

2.3.2.5 Aplicaciones actuales

El ASR está presente hoy en una amplia gama de aplicaciones cotidianas y profesionales. Entre ellas se destacan:

- Asistentes virtuales y dispositivos inteligentes: Siri, Alexa y Google Assistant, que permiten interactuar mediante comandos de voz.

- Subtitulado y transcripción automática: empleados en plataformas como YouTube y en servicios de accesibilidad para personas sordas o con hipoacusia.

- Call centers y telecomunicaciones: análisis en tiempo real de conversaciones para detectar intenciones, clasificar llamadas o identificar fraudes.

- Medicina y justicia: transcripción de historias clínicas y declaraciones judiciales, reduciendo tiempos de documentación.

- Seguridad y ciberdefensa: detección temprana de patrones sospechosos en llamadas de voz, clave en el combate contra el vishing y otras estafas telefónicas.

De este modo, el ASR no solo constituye una tecnología de soporte a la interacción humano-máquina, sino también una herramienta estratégica en el ámbito de la seguridad digital.

2.3.2.6 Síntesis

El reconocimiento automático del habla ha pasado de intentos rudimentarios con reglas fonéticas en los años 50 a sofisticados modelos neuronales end-to-end en la actualidad. Su progreso refleja la convergencia entre avances en procesamiento digital de señales, lingüística computacional y aprendizaje profundo. Aunque los sistemas modernos han alcanzado niveles de precisión cercanos al reconocimiento humano en

entornos controlados, persisten retos vinculados a la variabilidad del habla y a condiciones adversas.

Para este trabajo, el ASR tiene especial relevancia dado que forma parte de las tecnologías involucradas en las llamadas telefónicas y mensajes de voz en los que se cometen fraudes. Por otro lado, sin la capacidad de transcribir de manera precisa y rápida, resultaría inviable aplicar modelos de inteligencia artificial orientados a la detección temprana de estafas digitales.

2.4 Voice Cloning

2.4.1 Definición

La clonación de voz (voice cloning) es una técnica de inteligencia artificial que permite generar voces sintéticas casi indistinguibles de la voz de una real. A partir de grabaciones de audio, los modelos de aprendizaje profundo (deep learning) extraen características como timbre, prosodia (estudio de rasgos fonéticos que afectan a la métrica de los acentos), tono y acento, para luego reproducirlas en nuevos enunciados. Según Genelza (2024), “la clonación de voz utiliza machine learning para imitar el habla de una persona real” y requiere un conjunto de datos de voz que capture los rasgos únicos del individuo (Ver figura 12).

Los avances recientes en arquitecturas neuronales como las redes recurrentes, “transformers” y redes generativas adversarias o GANs (generative adversarial networks), han reducido drásticamente la cantidad de datos necesarios, haciendo posible entrenar modelos con apenas segundos de voz de referencia (Jurcys, Fenwick & Liaudanskas, 2024). Esta evolución tecnológica conecta directamente con los

sistemas Text-to-Speech (TTS) y con los deepfakes de voz, ampliando sus aplicaciones y riesgos.



Fig. 12 Esquema básico de Voice Cloning. Fuente: Elaboración propia con fines académicos.

2.4.2 Aplicaciones legítimas

El voice cloning presenta beneficios significativos en distintos campos. En el ámbito médico y de accesibilidad, permite a pacientes con pérdida del habla recuperar su propia voz para comunicarse (Genelza, 2024). En educación y cultura, se emplea para recrear voces históricas y enriquecer experiencias didácticas, así como en la narración de audiolibros o materiales formativos. En entretenimiento, se utiliza para doblaje automático en múltiples idiomas y la producción de contenido personalizado. Finalmente, en servicios al cliente, los clones de voz ofrecen interacciones más naturales y personalizadas, optimizando la atención automatizada (Jurcys et al., 2024).

2.4.3 Riesgos

El riesgo de abuso es igualmente considerable a los aspectos positivos. Uno de los principales riesgos es su utilización en estafas telefónicas y vishing como hemos tratado. Con voces clonadas, los delincuentes pueden suplantar la identidad de familiares, directivos o empleados de confianza para inducir transferencias o revelar información sensible.

Kassis y Hengartner (2021) demostraron experimentalmente que muestras de voz generadas artificialmente pueden evadir mecanismos de autenticación en determinados escenarios.

En Argentina, se ha reportado recientemente el "fraude del sí", donde los atacantes graban a la víctima respondiendo afirmativamente y manipulan ese audio para validar operaciones fraudulentas (Escobar Ruiz, 2025). Este ejemplo ilustra cómo la clonación de voz puede combinarse con ingeniería social y técnicas de vishing para aumentar la efectividad de la estafa.

Otro foco crítico es la biometría de voz. Dado que muchas instituciones utilizan la voz como factor de autenticación, su clonación demuele la fiabilidad de estos sistemas.

Hutiri, Papakyriakopoulos y Xiang (2024) señalan que los generadores de voz amplifican riesgos de fraude, suplantación y robo de identidad, generando nuevos patrones de daño sociotécnico.

Así también, la clonación de voz favorece la creación de deepfakes políticos y mediáticos, capaces de difundir desinformación y manipular la opinión pública, con un impacto directo en la confianza social y en la estabilidad democrática.

2.4.5 Impacto social

La clonación de voz desafía la percepción humana de autenticidad. Investigaciones recientes demuestran que incluso oyentes experimentados tienen dificultades para distinguir entre voces reales y sintéticas (Hutiri et al., 2024). Esto deteriora la confianza en las interacciones cotidianas y puede generar problemas en ámbitos legales al intentar adjudicar valor probatorio a un audio teniendo que acudir técnicas forenses expertas .

Los dilemas éticos giran en torno a tres ejes principales: consentimiento, privacidad y dignidad digital. Jurcys et al. (2024) proponen la necesidad de un "nuevo contrato social" que establezca el consentimiento como regla por defecto y refuerce la transparencia en la generación de voces sintéticas. Aunque este trabajo ,como se adelantó, no profundizará en lo regulatorio, es clave reconocer que la discusión ética y legal es parte del contexto global de esta tecnología.

2.4.6 Síntesis

El voice cloning representa una dualidad: por un lado, es una herramienta transformadora en accesibilidad, educación, entretenimiento y servicios; por otro, se

consolida como un factor de riesgo en fraudes telefónicos, suplantación biométrica y campañas de desinformación. Su facilidad de uso y el realismo alcanzado incrementan la vulnerabilidad de usuarios y organizaciones frente a ataques de ingeniería social.

Si bien los marcos regulatorios son un debate abierto y en evolución, el foco de este trabajo está en reconocer las amenazas y riesgos actuales, como el fraude del "sí", y sentar las bases para explorar soluciones técnicas y modelos de detección temprana en capítulos posteriores.

2.5 Modelos de IA

2.5.1 Definición

Comprender el concepto de la inteligencia artificial (IA) no es algo fácil, dado que nuestro acercamiento al mismo depende del enfoque desde el cual partamos. Russell y Norvig (2021) destacan que existen cuatro grandes formas de concebirla: pensar como los humanos, actuar como los humanos, pensar racionalmente y actuar racionalmente. Cada una de estas perspectivas refleja diferentes corrientes intelectuales y metodológicas, lo que explica por qué la IA no tiene una definición única y definitiva, sino que se entiende como una obra en construcción en donde se encuentran trabajando diferentes disciplinas: la filosofía, la psicología, la matemática y las ciencias computacionales.

2.5.2 El Aporte de las disciplinas

En sus inicios la IA se nutrió de los avances y preguntas de otras áreas del conocimiento. Como se mencionó anteriormente entre las más influyentes tenemos (ver figura 13):

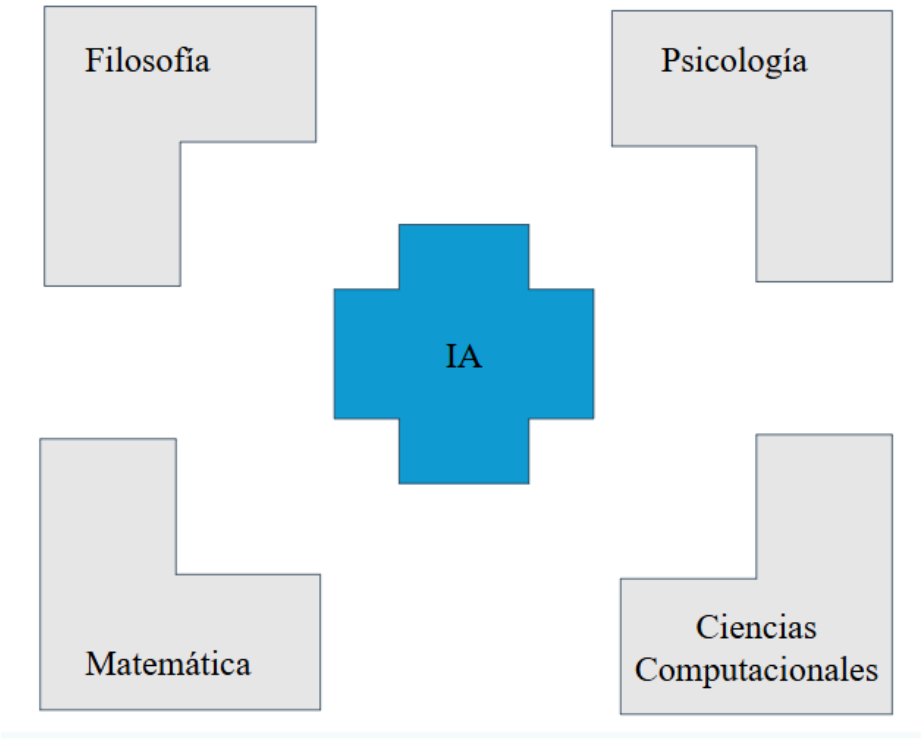


Fig. 13 Diagrama de disciplinas influyentes en la IA . Fuente: Elaboración propia con fines académicos.

- Filosofía: en la Antigüedad se plantearon interrogantes sobre la naturaleza de la mente, la lógica y la posibilidad de que el razonamiento humano pudiera ser formalizado. Desde el año 400 aC Sócrates, Platón, Aristóteles, Leibniz sentaron las bases conceptuales donde se basaría tiempo más tarde a la IA moderna. A través de sus aportes conjuntos, se concibió a la mente como una máquina que funciona a partir del conocimiento codificado en un lenguaje interno y que el pensamiento servía para determinar cuál era la acción correcta que había que emprender.

- Psicología: aportó modelos del comportamiento y teorías del aprendizaje. Experimentos sobre memoria, estímulo-respuesta y cognición crearon un marco para que los primeros sistemas pudiesen imitar procesos de percepción y decisión humana.

- Matemática: proporcionó herramientas esenciales como la lógica formal, la teoría de la probabilidad y la estadística, fundamentales para la representación del conocimiento y la inferencia. La programación matemática y la investigación operativa, por su parte, posibilitaron el diseño de algoritmos de optimización y planificación.

- Ciencias de la computación: hicieron posible la materialización de las ideas mencionadas en programas y máquinas. El surgimiento de la computación digital en los años 40 y 50, junto con la noción de máquina de Turing, mostró que era factible procesar información de manera simbólica y sentó el terreno para la IA como disciplina autónoma.

2.5.3 Breve Historia Contemporánea

En este apartado se pretende exponer algunos de los hitos más importantes de la historia contemporánea (ver figura 14).

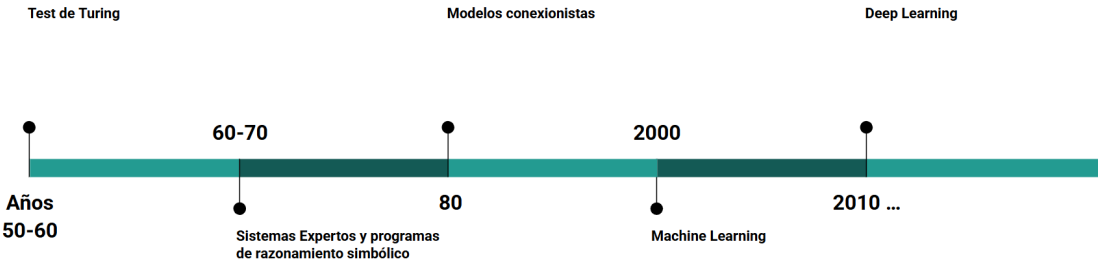


Fig 14 Línea de tiempo evolución de la IA en la historia contemporánea. Fuente: Elaboración propia con fines académicos.

Décadas de 1940 y 1950: Alan Turing propone su famoso "juego de la imitación" (Test de Turing) como criterio para evaluar la inteligencia de una máquina. Paralelamente, se desarrollan los primeros programas de ajedrez y razonamiento lógico.

Años 1960 y 1970: aparecen los primeros sistemas expertos y programas de razonamiento simbólico como ELIZA y SHRDLU, que buscaban imitar la conversación humana y la manipulación de objetos en entornos simulados.

Década de 1980: resurge el interés en los modelos conexionistas, con las redes neuronales artificiales capaces de aprender patrones complejos.

Años 2000 en adelante: se consolida el enfoque de machine learning, con modelos que aprenden de grandes volúmenes de datos.

Década de 2010: el deep learning revoluciona el campo con redes neuronales profundas que impulsan avances en reconocimiento de voz, visión por computadora y procesamiento de lenguaje natural.

Este breve recuento histórico muestra cómo los modelos de IA no son construcciones aisladas, sino que son resultado del trabajo hecho por investigadores de diferentes áreas. En cada generación podemos encontrar la incorporación de nuevos conceptos, técnicas y capacidades.

2.5.4 Clasificación de modelos de IA

El desarrollo de la IA ha dado lugar a distintos tipos de modelos, que pueden clasificarse en tres grandes categorías :

- Modelos simbólicos o clásicos: se basan en la representación explícita del conocimiento mediante reglas lógicas, árboles de decisión o sistemas expertos. Su ventaja es la interpretabilidad, pero presentan limitaciones al enfrentarse a entornos dinámicos o datos ambiguos.

- Modelos sub-simbólicos o conexionistas: utilizan técnicas estadísticas y redes neuronales para aprender directamente de los datos. No requieren conocimiento explícito, sino que extraen patrones a partir de grandes volúmenes de ejemplos. Este enfoque es la base de los avances recientes en aprendizaje profundo (deep learning).

- Modelos híbridos: combinan elementos simbólicos y conexionistas, con el objetivo de integrar la interpretabilidad de los primeros y la capacidad de generalización de los segundos.

Tipo de modelo	Descripción	Ejemplos	Ventajas	Limitaciones
Simbólicos (clásicos)	Basados en reglas, lógica formal y sistemas expertos. Representan conocimiento de manera explícita.	Sistemas expertos, árboles de decisión, razonamiento lógico.	Alta interpretabilidad, transparencia en decisiones.	Rígidos, poca adaptabilidad a entornos inciertos.
Sub-simbólicos (conexionistas)	Inspirados en redes neuronales y enfoques estadísticos. Aprenden patrones directamente de los datos.	Redes neuronales, deep learning, modelos probabilísticos.	Alta capacidad de generalización, manejo de datos complejos.	Opacos (cajas negras), requieren grandes volúmenes de datos.
Híbridos	Combinan elementos simbólicos y conexionistas, buscando interpretabilidad y capacidad de	Neuro-symbolic AI, modelos híbridos de razonamiento y aprendizaje.	Equilibran explicabilidad y potencia predictiva.	Mayor complejidad en diseño e implementación.

	generalización.			

Tabla 3. Clasificación de modelos AI. Fuente: Elaboración propia con fines académicos.

El libro Inteligencia Artificial: Iniciativa Latinoamericana de Libros de Texto Abiertos plantea que los modelos conexionistas, basados en el enfoque subsimbólico de la inteligencia artificial, han permitido el desarrollo de sistemas con capacidad de aprendizaje mediante redes neuronales, diferenciándose así de los enfoques simbólicos clásicos.

A partir de esta idea conceptual, puede entenderse que las técnicas modernas derivadas de este paradigma, como las Redes Neuronales Recurrentes (RNN), los Transformers y las Generative Adversarial Networks (GAN), han posibilitado avances significativos en áreas que durante décadas resultaron complejas para la IA tradicional, tales como el reconocimiento de voz, la visión por computadora y el procesamiento de lenguaje natural. Pero estos modelos serán examinados en el capítulo 2.6, donde se desarrollará cómo es que son aplicados.

2.5.5 Aplicaciones relevantes

Como se ha adelantado en este trabajo podemos encontrar que los modelos de IA aparecen relacionados con múltiples aplicaciones prácticas:

- Procesamiento de lenguaje natural (NLP en inglés): permite la comprensión y generación de texto en lenguajes humanos.

- Reconocimiento de voz y síntesis de habla: tecnologías como ASR y TTS se apoyan en modelos neuronales para transcribir y generar voz de manera natural.

- Clonación de voz y deepfakes: dependen de modelos generativos, como las Generative Adversarial Networks (GANs), capaces de imitar timbres y prosodias humanas.

- Sistemas de detección de fraudes: utilizan modelos supervisados para clasificar patrones de comportamiento y modelos no supervisados para identificar anomalías.(Estos modelos se desarrollarán en el apartado de "Machine Learning")

Estos modelos son la base técnica que sostiene tanto las amenazas como las soluciones que estamos abordando en este proyecto.

2.5.6 Desafíos

Si bien los modelos de IA a lo largo de los años han mostrado avances notables aún presentan limitaciones importantes. Una de ellas es transparencia: mientras los modelos simbólicos ofrecen una explicación sobre su lógica o razonamiento, los modelos neuronales suelen ser percibidos como "cajas negras". Así también, requieren grandes volúmenes de datos de calidad y altos recursos computacionales para su

entrenamiento. Estos factores condicionan su aplicabilidad en contextos de seguridad, donde la precisión debe ir acompañada de confianza e interpretabilidad.

2.5.7 Síntesis

En conclusión, los modelos de inteligencia artificial se conciben como el núcleo técnico de los avances recientes en el reconocimiento del habla, la síntesis de voz y la detección de fraudes digitales. Su clasificación en simbólicos, sub-simbólicos e híbridos refleja la evolución histórica de la disciplina y sus distintos enfoques de resolución de problemas.

En el marco de esta tesis, son elementos importantes de estudio dado que permiten entender cómo se construyen las herramientas que hoy facilitan tanto el desarrollo de estafas (como el voice cloning o el vishing) como las estrategias para detectarlas y mitigarlas.

2.6 Machine Learning: Aprendizaje supervisado y no supervisado

2.6.1 Definición

El machine learning (ML) es un subcampo de la inteligencia artificial el cual busca que las computadoras aprendan empíricamente sin necesidad de ser programadas para ello. Según Nilsen (2018), se trata de “descubrir reglas ocultas en los datos y utilizarlas para predecir o clasificar nuevos casos”.

Tradicionalmente, los sistemas computacionales se construían mediante programación explícita, es decir, el desarrollador diseñaba un algoritmo para recibir ciertos datos de entrada y producir una salida determinada (Ver figura 15). En este esquema clásico se debían anticipar todas las reglas necesarias para resolver un problema dado.

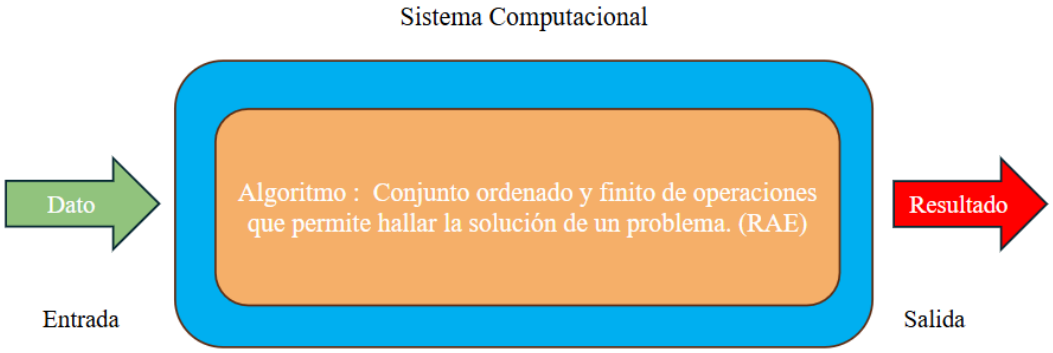


Fig 15 Esquema tradicional de programación. Fuente: Elaboración propia con fines académicos.

El ML invierte esta lógica. En vez de escribir manualmente el conjunto de pasos, se le proporciona al sistema un grupo de datos junto con las respuestas correctas (en el caso del aprendizaje supervisado), y el sistema desarrolla un modelo que explica la relación entre las entradas y las salidas (Ver figura 16). Como también resume Nilsen (2018), la idea central es que "el computador no recibe directamente las reglas, sino que las infiere de la experiencia".

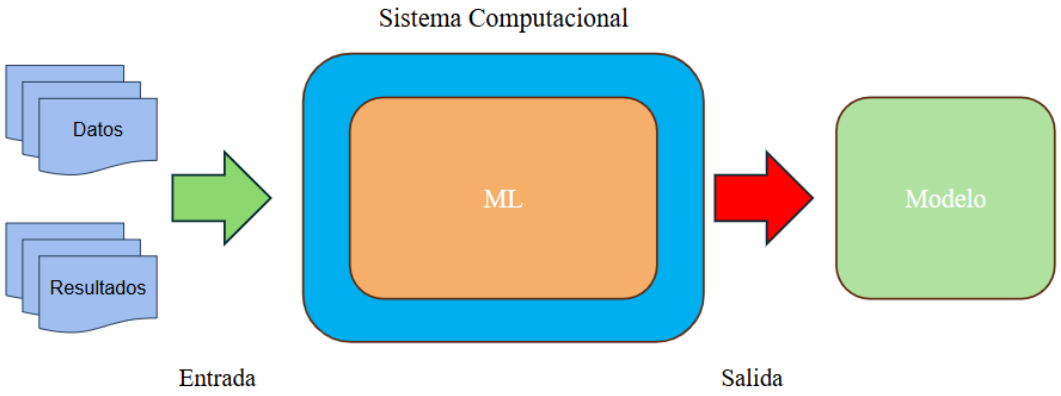


Fig. 16 Esquema básico de ML (Aprendizaje Supervisado). Fuente: Elaboración propia con fines académicos.

De esta manera, lo que antes era el resultado obtenido a través de pasos procedurales y de validación, ahora se convierte en el producto de un predicción realizada por un modelo que ajusta sus parámetros internos para minimizar errores y mejorar su capacidad de acierto.

Este cambio de paradigma planteado explica por qué el ML se ha convertido en una herramienta esencial para problemas complejos como la detección de fraudes, donde resulta inviable anticipar de antemano todas las posibles modalidades de ataque. También representa una evolución frente a los enfoques simbólicos, al privilegiar el aprendizaje a partir de datos masivos, propios del mundo digital contemporáneo.

El ML se divide en diferentes enfoques, entre los cuales se destaca el aprendizaje supervisado y el no supervisado. Ambos son relevantes para la detección de estafas telefónicas y digitales, aunque presentan ventajas y limitaciones que conviene analizar en detalle.

2.6.2 Aprendizaje supervisado

En el aprendizaje supervisado, el modelo se entrena con un conjunto de datos etiquetados: cada entrada está asociada a una salida correcta. El sistema aprende a predecir esas salidas y luego generaliza frente a nuevos casos. Choi, et al. (2020) explica que este paradigma busca "minimizar el error entre las predicciones del modelo y los valores reales, utilizando un conjunto de entrenamiento previamente etiquetado".

Dentro de los modelos de aprendizaje supervisados encontraremos a la regresión lineal, las support vector machines (SVM), los árboles de decisión y las redes neuronales artificiales (ANN).

Para mejorar la comprensión propongo imaginar un sistema que analizará 10.000 grabaciones de llamadas, de las cuales 5.000 serán marcadas como inofensivas y 5.000 como fraudulentas. Aquí la idea es que nuestro modelo aprenda a reconocer patrones de voz, frases recurrentes ("su tarjeta fue bloqueada", "necesitamos su clave") o anomalías en la entonación que se repitan normalmente en intentos de estafas. Todo esto con el fin de generar un clasificador capaz detectar las amenazas en un conjunto de registros nuevos sin etiquetar.

2.6.2.1 Regresión Logística

La regresión logística es uno de los algoritmos supervisados más utilizados en problemas de clasificación binaria. A través de una función sigmoide (función matemática no lineal cuya gráfica tiene forma de "S" y que transforma un número real en un valor entre 0 y 1), convierte una combinación lineal de variables en probabilidades (Ver figura 17).

$$P(y = 1|x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + ... + \beta_n x_n)}}$$

En nuestro caso por ejemplo el modelo nos permitiría calcular la probabilidad de que una llamada sea fraudulenta (teniendo en cuenta diferentes características como el horario, la duración o la presencia de expresiones de urgencia).

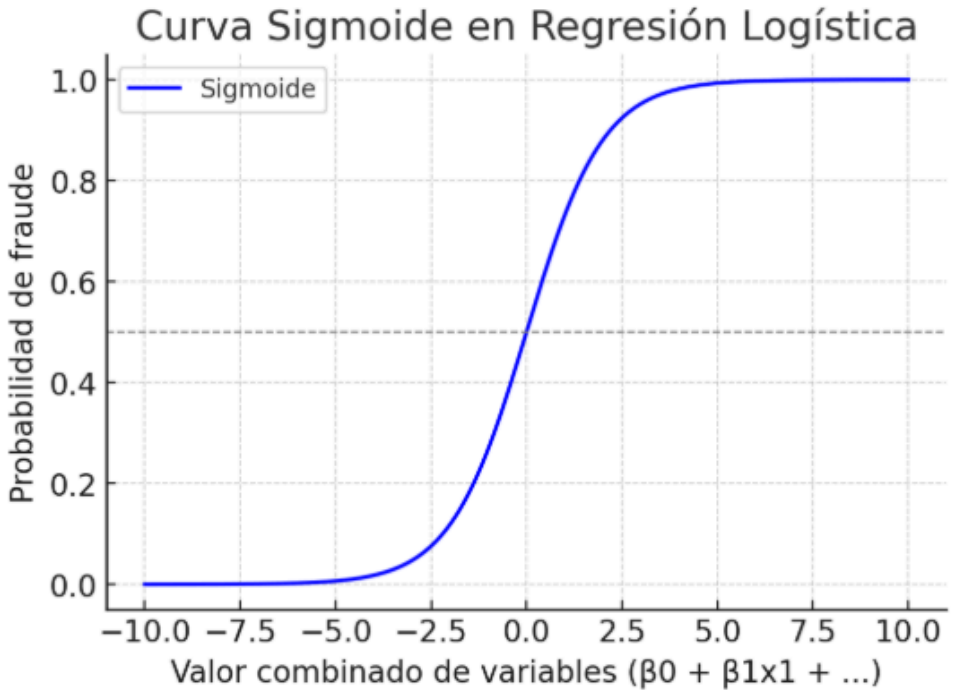


Fig. 17 Representación curva Sigmoide . Fuente: Elaboración propia con fines académicos mediante herramienta google colab.

2.6.2.2 Árboles de decisión

Los árboles de decisión permiten dividir los datos en ramas según reglas lógicas, lo que los hace muy interpretables. Cada nodo plantea una condición (ej. "¿el número es desconocido?"), y las hojas representan el resultado final (Ver figura 18).

Supongamos que queremos determinar cierta lógica para definir si un mensaje es fraudulento o no. Podríamos hacer uso de ciertas validaciones como si dentro del texto se encuentran enlaces a sitios externos sospechosos o entrecortados, si existe una redacción con tono de urgencia o si el remitente es desconocido. La manera en la cual prioricemos estas características nos generarán los cortes de división necesarios para clasificar la conversación.

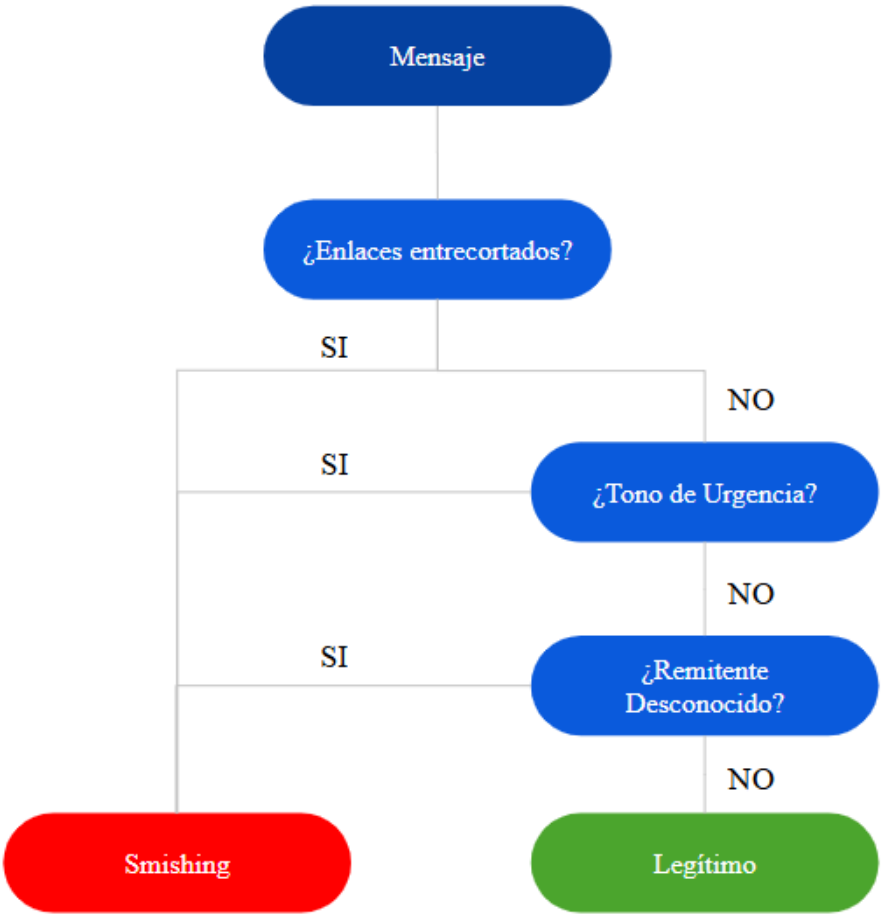


Fig. 18 Árbol de clasificación de Smishing simplificado. Fuente: Elaboración propia con fines académicos.

2.6.2.3 Redes neuronales artificiales (ANN)

Las redes neuronales artificiales (ANN por sus siglas en inglés) constituyen otra técnica supervisada inspirada en la biología. Cada ANN contiene nodos (similares a los cuerpos celulares) que se comunican entre sí mediante conexiones (similares a los axones y las dendritas). De forma muy similar a como se fortalecen las sinapsis entre neuronas cuando sus neuronas tienen salidas correlacionadas en una red neuronal biológica (la teoría hebbiana postula que "los nervios que se activan juntos, se

conectan"), las conexiones entre los nodos de una ANN se ponderan en función de su capacidad para proporcionar el resultado deseado (Choi, et al. 2020).

Se componen de (Ver figura 19):

- Una capa de entrada que recibe las variables.
- Varias capas ocultas que transforman la información.
- Una capa de salida que entrega la predicción.

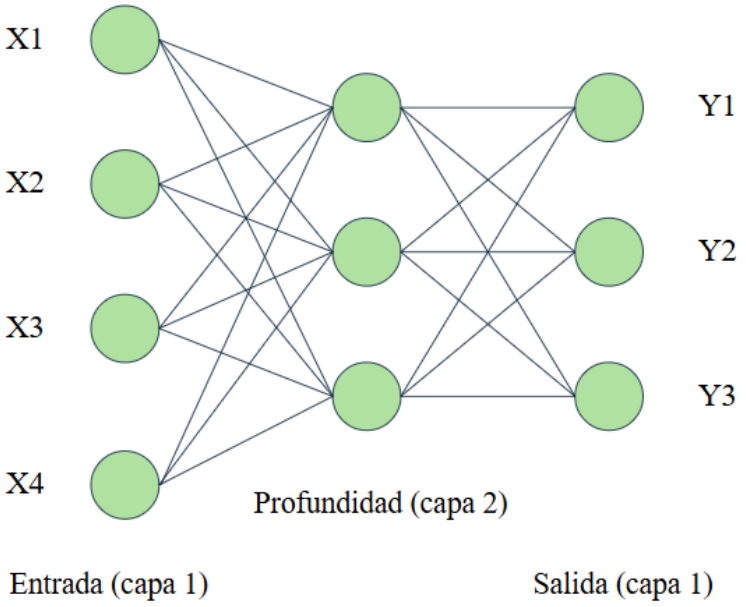


Fig. 19 Red neuronal artificial simplificada. Fuente: Elaboración propia con fines académicos.

El aprendizaje se realiza mediante retropropagación del error (backpropagation), que ajusta los pesos de las conexiones para minimizar la diferencia entre la salida que se ha predecido y la real.

Una ANN podría entrenarse con grabaciones de voz etiquetadas (fraude/no fraude) para identificar patrones de entonación sospechosos en tiempo real.

2.6.3 Deep Learning como evolución del supervisado

El deep learning (DL) surge como una extensión de las ANN, caracterizada por el uso de múltiples capas ocultas que permiten aprender representaciones jerárquicas de los datos. Gracias al auge del big data y las GPU (del inglés graphics processing unit), el DL se consolidó en la década de 2010 (Choi, et al. 2020).

Principales arquitecturas:

- CNN (Convolutional Neural Networks): Aunque se desarrollaron para visión por computadora, también se aplican en audio analizando espectrogramas de voz, lo que permite identificar anomalías sutiles permitiendo detectar fraudes.

- RNN (Recurrent Neural Networks): Las RNN procesan secuencias, ya que cada neurona retiene memoria de estados anteriores. Son útiles para datos de voz y texto, pero sufren problemas de gradientes en secuencias largas. Para resolver esto surgieron variantes como LSTM (Long Short-Term Memory) y GRU (Gated Recurrent Unit).

- Transformers: evolución de las RNN que emplea auto-atención para procesar secuencias largas de manera más eficiente.

- GAN (Generative Adversarial Networks): redes generativas capaces de crear datos sintéticos, incluyendo voces falsas.

Sangit et al. (2025) documentan cómo estas arquitecturas han logrado altos niveles de precisión en la detección de fraudes financieros, lo que demuestra su aplicabilidad a la identificación de estafas telefónicas y digitales.

2.6.3.1 Transformers

Los Transformers representan un avance fundamental en el campo de las redes neuronales para el procesamiento de secuencias. A diferencia de las RNN, que procesan la información de manera secuencial, los Transformers se basan en el mecanismo de auto-atención (self-attention), que evalúa las relaciones entre todos los elementos de una secuencia en paralelo. Esto les permite capturar dependencias a largo plazo y entrenar con mayor eficiencia (Choi, et al. 2020).

En el contexto del aprendizaje supervisado, los Transformers se entrenan con pares entrada/salida por ejemplo, pasandole el texto en un idioma y su traducción, o un fragmento de audio y su transcripción, alcanzando resultados superiores en tareas de procesamiento de lenguaje natural y reconocimiento automático del habla (ASR).

Sangit et al. (2025) destacan que los modelos de deep learning más recientes, incluidos los Transformers, han mostrado gran eficacia en la detección de fraudes financieros gracias a su capacidad para procesar secuencias de transacciones y detectar patrones sutiles. En el marco de esta tesis, estas capacidades poseen relación directa y aplicables a:

- Fraudes telefónicos: análisis de conversaciones para identificar frases, entonaciones o estructuras lingüísticas asociadas a ingeniería social.

- Phishing y smishing: clasificación automática de mensajes de texto o correos electrónicos fraudulentos frente a los legítimos.

- Voice cloning y deepfakes: como modelo generativo, también se utilizan para entrenar sistemas capaces de producir o detectar voces sintéticas.

Por estas razones, los Transformers se encuentran presentes como una arquitectura clave en la detección de estafas digitales.

2.6.4 Aprendizaje no supervisado

El aprendizaje no supervisado se aplica cuando los datos no están etiquetados. El objetivo no es predecir un resultado particular sino más bien descubrir estructuras ocultas, agrupamientos o relaciones internas.

Naeem et al. (2023) resaltan que este paradigma es especialmente útil cuando resulta costoso o imposible etiquetar los datos de manera manual, como ocurre con grandes volúmenes de registros telefónicos. Su principal aplicación en seguridad es la detección de anomalías, es decir, identificar comportamientos que se desvían de lo que consideramos “normal”.

Debajo mencionare algunas de las técnicas más utilizadas para lograr el propósito mencionado.

2.6.4.1 Clustering

Los algoritmos de clustering agrupan datos según similitud (Ver figura 20):

- k-means: busca formar k grupos de puntos similares.

- DBSCAN: detecta grupos de densidad y outliers.

Imaginemos un banco que analiza millones de transacciones con un algoritmo de clustering. Los clientes pueden agruparse por patrones de gasto, ubicación y horarios. Si se encontrara un grupo nuevo con transacciones inusuales (por ejemplo, compras nocturnas en otro país con un mismo número de tarjeta), el sistema marcaría esas operaciones como sospechosas.

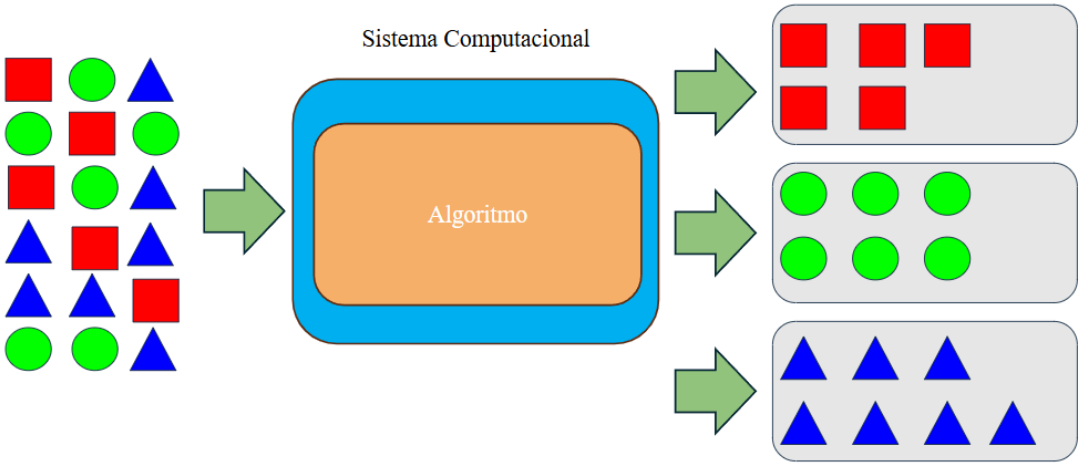


Fig. 20 Clustering simplificado. Fuente: Elaboración propia con fines académicos.

2.6.4.2 Reducción de dimensionalidad (PCA)

El Análisis de Componentes Principales (PCA) es un método lineal para proyectar datos de dimensiones altas a 2 o 3 dimensiones. Visualizar datos con este método permite identificar la similitud entre los puntos cercanos y la diferencia entre los puntos lejanos (Nilsen, V. 2018).

Esto reduce la complejidad de los datos extrayendo las características más relevantes. Por ejemplo, lo utilizaremos si desearamos simplificar variables acústicas en grabaciones de voz para detectar desviaciones sospechosas.

2.6.4.3 Autoencoders

Los autoencoders son redes neuronales que aprenden a reconstruir datos tras comprimirlos en un espacio reducido. Al comparar la entrada con la salida reconstruida, se pueden detectar anomalías.

Su implementación ayuda a detectar una voz manipulada dado que la reconstrucción de la misma difiere demasiado de la señal original.

2.6.4.4 GANs en no supervisado

Además de generar datos, las GANs permiten aumentar datos de entrenamiento (data augmentation) o detectar inconsistencias en señales falsas.

En nuestro caso si no tuviéramos las suficientes muestras necesarias para entrenar un sistema de seguridad podríamos realizarlo con ejemplos sintéticos de fraudes para mejorar su robustez.

Dicho lo anterior, entonces, los autoencoders permiten reconstruir datos y señalar desvíos, mientras que las Generative Adversarial Networks (GANs) se usan para generar datos sintéticos y, al mismo tiempo, para entrenar detectores capaces de diferenciar señales reales de falsificadas (Sangit et al., 2025).

2.6.5 Comparación de los enfoques

Cada enfoque presenta ventajas y limitaciones. El supervisado ofrece precisión, pero depende de conjuntos de datos bien etiquetados. El no supervisado es flexible y descubre fraudes desconocidos, aunque puede generar falsos positivos y su interpretación es más compleja (García Sandoval, 2024).

Paradigma	Datos requeridos	Ejemplos de algoritmos	Ventajas	Limitaciones	Ejemplo aplicado a fraudes
Supervisado	Datos etiquetados	Regresión logística, árboles, ANN, CNN, RNN, Transformers	Alta precisión, resultados claros	Requiere etiquetas, menos flexible	Clasificación de llamadas fraudulentas
No supervisado	Datos no etiquetados	k-means, DBSCAN, PCA, Autoencoders, GANs	Detecta patrones emergentes, útil en fraude desconocido	Difícil de interpretar, falsos positivos posibles	Detección de patrones de voz o transacciones atípicas

Tabla 4. Comparación entre aprendizaje supervisado y no supervisado. Fuente: Elaboración propia con fines académicos.

2.6.6 Síntesis

El aprendizaje supervisado y el no supervisado son dos enfoques complementarios dentro del ML. El primero se centra en predecir a partir de ejemplos previos y es útil para reconocer modalidades conocidas de fraude, mientras que el segundo permite detectar anomalías y descubrir estafas emergentes. El deep learning, con CNN, RNN, Transformers y GANs, potencia estas aproximaciones al aprender representaciones complejas de voz, texto y transacciones, ofreciendo herramientas capaces de procesar lenguaje y voz con gran realismo.

Dentro de esta tesis, ambas metodologías resultan esenciales para fundamentar la construcción de un modelo de detección temprana de estafas telefónicas y digitales: el supervisado aporta precisión sobre casos conocidos, mientras que el no supervisado brinda adaptabilidad frente a nuevas amenazas.

2.7 Aplicación móviles

2.7.1 Introducción

El teléfono móvil es el dispositivo central de la vida digital contemporánea. Como ejemplo local podemos encontrar que en Argentina, los usuarios pasan más de tres meses al año mirando la pantalla de su celular, lo que refleja tanto su importancia social como la escala de exposición a riesgos digitales(Asprea, 2025). A nivel global, informes de Datareportal confirman que más del 96% de los usuarios de internet acceden a la red mediante un smartphone, lo que convierte a este dispositivo en la puerta de entrada más usada para fraudes digitales y en simultáneo en la plataforma ideal para implementar medidas de protección (Datareportal, 2025).

2.7.2 Evolución de las aplicaciones móviles de seguridad

Las estafas digitales vistas en este trabajo (phishing,smishing o vishing) encuentran al smartphone como el canal de ataque predilecto. Para mitigar estos riesgos, han surgido aplicaciones de bloqueo de llamadas y mensajes sospechosos (por ejemplo Truecaller, Hiya), que se apoyan en listas negras y análisis de permisos del sistema operativo (Ver figura 21). No obstante, estudios técnicos muestran que estos bloqueadores presentan limitaciones relevantes: su dependencia de bases de datos estáticas reduce la eficacia frente a fraudes novedosos, y el acceso excesivo a permisos puede comprometer la privacidad del usuario (Busropan, 2020).

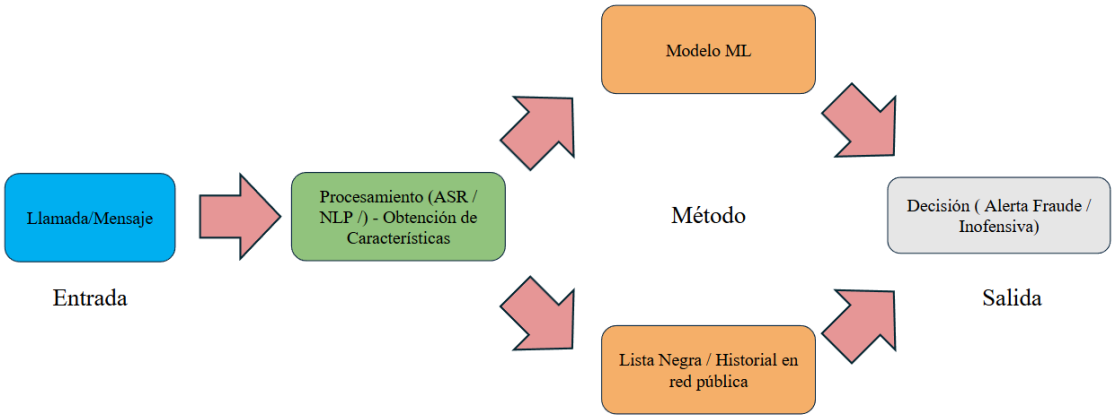


Fig. 21 Flujo simplificado detección de fraudes. Fuente: Elaboración propia con fines académicos.

La evolución hacia la era del 5G ha intensificado los desafíos. La mayor velocidad y capacidad de red favorecen la escalabilidad de fraudes telefónicos y

digitales, lo que exige soluciones más dinámicas y distribuidas. Pietri, Mamei y Colajanni (2025) destacan la necesidad de integrar detección en tiempo real directamente en el dispositivo, reduciendo la dependencia de infraestructuras centralizadas.

2.7.3 Aplicaciones móviles potenciadas por Machine Learning

La incorporación de modelos de aprendizaje automático en aplicaciones móviles ha abierto nuevas posibilidades de protección. Goel, Ahmad, Jain y Goel (2024) proponen un framework de detección de smishing basado en machine learning, capaz de analizar características lingüísticas y patrones de comportamiento en mensajes SMS . Así se busca lograr una reducción significativa de falsos positivos en comparación con enfoques basados solo en reglas.

De manera complementaria, Zimba, Phiri, Kashale y Phiri (2024) desarrollan un modelo que combina machine learning y procesamiento de lenguaje natural (NLP) para la detección de smishing en transacciones de dinero móvil. Sus resultados muestran que la integración de IA en dispositivos móviles no solo mejora la capacidad de respuesta frente a fraudes, sino que también puede adaptarse a contextos específicos como los servicios financieros digitales.

2.7.4 Síntesis

Las aplicaciones móviles representan hoy la primera línea de defensa contra las estafas telefónicas y digitales, debido a que concentran tanto el riesgo como la

posibilidad de respuesta inmediata. La evolución de estas soluciones pasó de enfoques rígidos, basados en procedimientos heurísticos, listas negras y permisos, hacia la integración de modelos de machine learning que ofrecen mayor adaptabilidad y precisión. No obstante, persisten desafíos asociados a la privacidad de datos, al consumo de recursos del dispositivo y a la necesidad de actualizaciones constantes frente a nuevas tácticas de fraude.

En este contexto se encuentra al teléfono móvil emergiendo no sólo como el principal punto de ataque, sino también como el entorno estratégico más adecuado para validar modelos de detección inteligente, como el que se propone en esta investigación.

3. MODELO

La siguiente sección describe el diseño, la arquitectura y los componentes del modelo propuesto de detección temprana de estafas telefónicas y digitales mediante inteligencia artificial, denominado “ProxIA”.

La propuesta no tiene como finalidad desarrollar una aplicación comercial completa, sino validar, a través del MVP, parte del núcleo técnico que sustenta su factibilidad.

3.1 Presentación del modelo

La evolución de las estafas telefónicas y digitales, presentada en este trabajo, nos permite inferir que las estrategias de detección y prevención tradicionales deben

tomar un camino más allá de los métodos reactivos si se busca obtener mejores resultados. Los delincuentes usan ,cada vez más, técnicas de ingeniería social con mayor sofisticación. La clonación de voz y automatización del engaño mediante herramientas basadas en inteligencia artificial (IA) son un ejemplo.

Este proyecto pone sobre la mesa la necesidad de crear sistemas de defensa más inteligentes, capaces de intervenir antes de que la víctima establezca un vínculo directo con el atacante.

3.1.1 Propuesta

Como respuesta a la problemática desarrollada en este trabajo se propone el diseño de una aplicación móvil basada en IA que actúe como un proxy entre el posible atacante y la víctima. Su propósito es la detección temprana de intentos de estafa y la protección de la identidad vocal del usuario (Ver figura 22).

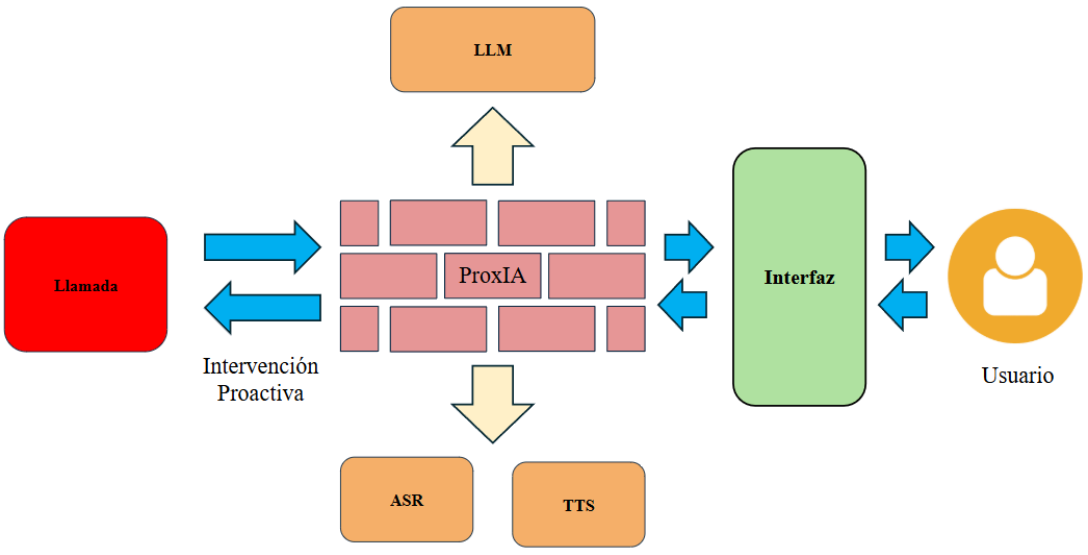


Fig. 22 Modelo simplificado aplicación para detección de fraudes .Fuente: Elaboración propia con fines académicos.

El aplicativo, de ahora en más llamado “ProxIA”, tiene como objetivo principal ser un intermediario capaz de interceptar una comunicación en directo, analizar y advertir al usuario sobre la posible presencia de un intento de fraude.

ProxIA se situará entre ambos interlocutores conformado por la combinación de tres componentes elementales:

- Reconocimiento Automático del Habla (ASR): Convirtiendo la voz del interlocutor en texto, permitiendo su análisis semántico y emocional.

- Procesamiento y clasificación con Modelos de IA: Detectando patrones en el discurso (por ejemplo de urgencia, presión, solicitud de datos personales o financieros).

- Síntesis de Voz (TTS): Convirtiendo en respuestas de voz los resultados que genere la lógica del modelo y así manteniendo la comunicación con el posible atacante.

El diseño plantea un entorno en el que la aplicación puede capturar y procesar el audio de una llamada telefónica en curso, operando así como una capa de seguridad preventiva en tiempo real (Ver figura 23).

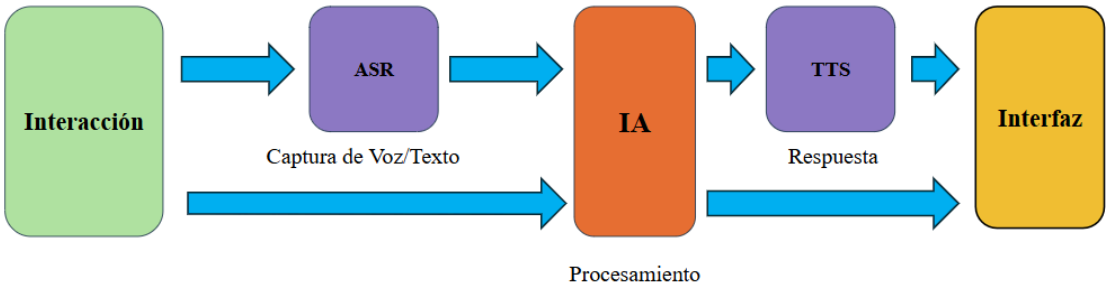


Fig. 23 Flujo general de operación de ProxIA. Fuente: Elaboración propia con fines académicos.

3.1.2 Limitaciones

En la actualidad existen limitaciones críticas que impiden implementar la aplicación de acuerdo a lo desarrollado en la anterior sección.

Tanto Android (Google ,2018) como iOS (Call Kit Apple, 2024) le restringen a aplicaciones de terceros el acceso al canal de audio de llamadas nativas (GSM o VoLTE), por motivos de privacidad y protección de datos personales. Estas políticas, reforzadas desde Android 9 (Pie) y los términos de Google Play (Google Play Console Help, 2024), prohíben la grabación, escucha o modificación de llamadas sin consentimiento explícito de todos los participantes y sin ser una aplicación del sistema o del operador.

Por todo lo expresado y en consecuencia, el modelo se reestructuró (Ver Figura 24) para adaptarse a los limitantes pero con la intención y el cuidado de mantener intacto su propósito original cambiando solo su entorno de ejecución. En vez de actuar como intermediario en llamadas móviles, el sistema se plantea para operar en plataformas VoIP (SIP/WebRTC), donde la comunicación se realiza dentro de una aplicación propia o homologada.

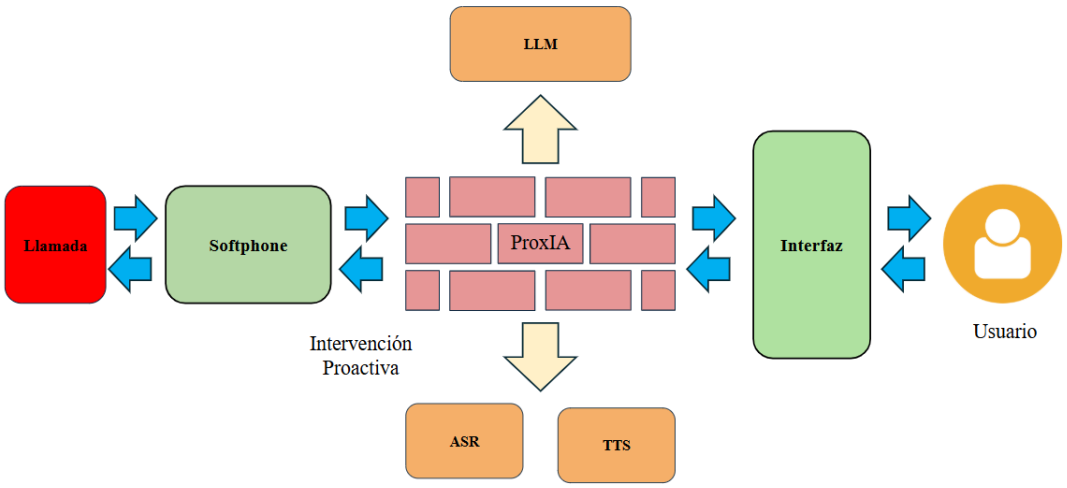


Fig. 24 Modelo reestructurado de aplicación para detección de fraudes en tiempo real. Fuente: Elaboración propia con fines académicos.

En estos ambientes, el flujo de audio puede ser procesado legítimamente dentro del cliente, lo que habilita la integración del modelo de inteligencia artificial, el motor de reconocimiento de voz y el sintetizador de voz sin vulnerar políticas de privacidad.

También sería posible incluir un módulo de análisis de texto que procese mensajes recibidos en servicios de mensajería instantánea, siempre que dichos textos

sean accesibles a través del sistema de notificaciones del dispositivo (por ejemplo, WhatsApp).

En este último caso, el sistema solo puede leer el contenido mostrado en pantalla y emitir una notificación de alerta, sin modificar ni responder los mensajes.

La reestructuración mantiene el principio de procesamiento local (edge-only), dado que el análisis del audio y del texto continúa realizándose en el dispositivo del usuario, sin transmisión del contenido hacia servicios externos..

Más allá de lo expuesto se reconoce que el enfoque original podría volverse viable en el futuro si se produjeran cambios en las políticas de acceso a llamadas, en la infraestructura de los operadores o en la legislación sobre tratamiento de datos en tiempo real.

Por este motivo, la implementación del modelo como proxy directo de comunicación se plantea como una línea de investigación futura.

3.1.3 Factores del diseño

Se establece que el diseño del modelo resultante esté basado en los siguientes factores:

- Prevención proactiva: Identificar el riesgo antes de que el usuario comparta información sensible o quede expuesto a manipulación emocional.

- Privacidad: La información procesada (audio, texto, patrones de voz) será analizada de manera local y cuando no sea posible deben aplicarse medidas de encriptación o cifrado.

- Accesibilidad y escalabilidad: Permitir que la solución pueda instalarse en dispositivos móviles con bajos requerimientos técnicos para que sea compatible con la mayor cantidad posible.

El clasificador a desarrollar como prototipo (MVP) se enfocará exclusivamente en el caso de uso más crítico: las llamadas telefónicas sospechosas (vishing), donde se emplea la voz como canal principal de engaño y en donde la intervención automática puede tener un impacto directo en la prevención del fraude.

Con lo planteado no se busca reemplazar el juicio humano o la verificación manual, sino que se intenta ofrecer una herramienta de apoyo que complemente las estrategias de seguridad personal a través de un componente de asistencia.

3.2 Arquitectura del modelo

El propósito de este apartado es presentar el diseño y arquitectura del prototipo de solución. Se realizarán justificaciones asociadas al camino elegido y herramientas recomendadas mientras se desarrollarán las definiciones de las necesarias..

3.2.1 Visión del Diseño

El diseño a desarrollar busca materializar la idea original del modelo pero bajo un marco técnico realista y compatible con las políticas actuales de los sistemas operativos móviles.

ProxIA se plantea preliminarmente como una arquitectura modular. Un sistema con una estructura compuesta por diferentes componentes capaces de operar y adaptarse según el entorno de ejecución (En el dispositivo, nube o híbrido). Los elementos que constituyen el aplicativo son:

- Captura de audio o texto, originados en una comunicación entrante.
- Reconocimiento de voz (ASR), encargado de transformar el audio en texto.
- Clasificación mediante inteligencia artificial, para evaluar el riesgo del mensaje.
- Síntesis de voz (TTS) o alertas visuales, para advertir al usuario o responder de forma automática.
- Módulo de control y notificación, que gestione alertas hacia el usuario y/o contactos de confianza.
- Capa de seguridad y privacidad, que asegure el cifrado, anonimización y eliminación de datos sensibles.

Se consideran datos sensibles aquellos que pueden permitir la identificación directa o indirecta de una persona, como fragmentos de voz, transcripciones de comunicaciones o metadatos asociados a la interacción.

3.2.1.1 Criterios de diseño

El diseño de la arquitectura a implementar fundara sus bases en diferentes criterios de diseño arquitectónico, vinculados a atributos de calidad, buscando equilibrar las diferentes propiedades claves del sistema:

- **Privacidad**: Al operar con comunicaciones personales será importante considerar una configuración de elementos que permitan aplicar mecanismos de seguridad en la transmisión de datos.

- **Latencia y disponibilidad**: Al tratarse de comunicaciones en tiempo real, es importante poder operar en condiciones donde la conexión a servicios externos pueda verse afectada y poder responder con rapidez.

- **Eficiencia operativa y coste**: Deben seleccionarse tecnologías que permitan hacer uso eficiente de los servicios necesarios para la ejecución de la aplicación . De esta manera poder reducir costos variables del consumo.

- **Escalabilidad y adaptabilidad**: El diseño debe poder admitir su evolución si fuera necesario agregar funcionalidades o cubrir escenarios más complejos a los planteados en este proyecto.

3.2.2 Comparativa de enfoques arquitectónicos

De acuerdo a los criterios planteados en la anterior sección se realizó un investigación sobre de las distintas arquitecturas disponibles para la implementación de la aplicación .

Como resultado se encontró la existencia de tres paradigmas: Cloud Computing, Edge Computing (On-Device) y modelos híbridos. A continuación se pasarán a definir los mismos con el fin de tener una idea clara de los beneficios y desventajas de cada uno.

3.2.2.1 Cloud Computing

El cloud computing se consolidó por su capacidad de ofrecer recursos elásticos de procesamiento y almacenamiento mediante servidores centralizados. Grandes proveedores como Google Cloud, AWS y Azure han masificado este modelo al permitir el acceso bajo demanda a servicios de IA, ASR y TTS (Hua et al., 2023).

Sin embargo, con el crecimiento del IoT (Internet de las cosas por sus siglas en inglés) y la generación masiva de datos, el procesamiento exclusivo en la nube comenzó a exhibir limitaciones: alta latencia, cuellos de banda, costos de transmisión y riesgos de privacidad (Hua et al., 2023). Estos problemas claramente aumentan en

escenarios de interacción en tiempo real, como una llamada telefónica en curso, donde cada milisegundo impacta la experiencia del usuario.

Cherukuri (2023) agrega que, además de los costes de almacenamiento y cómputo, las arquitecturas cloud implican un gasto energético elevado y dependencia constante de la conectividad. En entornos de alta concurrencia o cobertura irregular, esto limita la autonomía de los sistemas sensibles a la latencia.

Como conclusión podemos observar que el enfoque “Cloud-Only” (del inglés, solo en la nube) es adecuado para sistemas con requerimientos de entrenamiento masivo y escalabilidad global, pero no es óptimo para aplicaciones que deben reaccionar localmente con baja latencia y sin exponer información sensible.

3.2.2.2 Edge Computing (On-Device)

El modelo edge delega parte del procesamiento y almacenamiento en los dispositivos cercanos al usuario o “nodos de borde”. Según Hua et al. (2023 p. 184:2), el núcleo del Edge Computing consiste en “desplazar las operaciones de procesamiento, almacenamiento y cómputo que antes requerían del cloud hacia el borde de la red cercano a los dispositivos terminales” .

Este paradigma reduce el tiempo de transmisión, la latencia y la congestión de red, al tiempo que mejora la privacidad, dado que los datos permanecen en el dispositivo (Hua et al., 2023). De la investigación se extrae que la computación en el borde nació como respuesta a mitigar los limitantes de la nube mencionados.

Cherukuri (2023) complementa que el Edge Computing “minimiza los retrasos al procesar los datos en el punto de origen, disminuyendo el uso de ancho de banda y permitiendo respuestas casi instantáneas”. Esta característica es clave entonces para sistemas de IA que deben analizar voz o texto en tiempo real como el de este trabajo.

El modelo on-device ofrece una ventaja importante ya que procesar voz y texto localmente nos garantiza también la no exposición de datos sensibles y reduce la latencia de respuesta . También nos permite eliminar la dependencia de servicios pagos por uso (ASR/TTS en la nube). Los estudios de Hua et al. (2023) también destacan que la IA ejecutada en el borde mejora la estabilidad y confiabilidad del sistema al evitar los puntos únicos de falla propios del modelo centralizado.

Ahora bien, este enfoque nos presenta el desafío de tener que manejar y lidiar con las limitaciones de memoria y energía que pudieran tenerse en los dispositivos móviles.

3.2.2.3 Arquitectura Híbrida (Edge - Cloud)

La arquitectura híbrida busca combinar ambas tecnologías con el fin de aprovechar los beneficios que estas pueden dar y minimizar sus desventajas. Por ejemplo, Hua et al. (2023) identifican tres modos de distribuir el uso de IA entre nube y borde :

- Entrenamiento y razonamiento en la nube: los datos masivos se pre procesan y luego se suben a la nube para el entrenamiento y razonamiento de la IA.

- Entrenamiento en la nube y razonamiento en el borde (modelo más común en IA móvil): las tareas de razonamiento de IA se realizan en el borde o en el extremo, mientras que las tareas de entrenamiento de modelos todavía se realizan en la nube.

- Entrenamiento y razonamiento parcial o total en el borde : ofrece un equilibrio entre latencia, precisión y costos. Permite operar localmente por defecto, solo delega trabajo a la nube cuando el modelo local presenta baja confianza o necesita refinamiento. También aumenta el procesamiento y consumo de energía en el borde.

A modo de resumen y ejemplo se elabora y se deja la siguiente tabla comparativa de los 3 enfoques (Ver Tabla 5) :

Criterio	**Cloud-Only**	**Edge (On-Device)**	**Híbrido Edge-Cloud**
Latencia	Alta	Muy baja	Baja-media
Privacidad	Baja	Muy alta	Alta con control
Coste operativo	Alto (por uso API)	Bajo	Moderado
Escalabilidad	Muy alta	Limitada	Alta
Energía local	Bajo consumo	Alto consumo	Variable

Compatibilidad con el proyecto	Dependiente de la red	Respuestas rápidas	Posible evolución futura

Tabla 5. Comparación entre enfoques de arquitectura. Fuente: Elaboración propia con fines académicos.

3.2.2.4 Punto de vista económico : costos de los enfoques

Más allá de la comparativa funcional y no funcional de los enfoques es necesario realizar un análisis económico que nos permita seleccionar la opción más adecuada para la implementación del prototipo. De acuerdo con Hua et al. (2023) y Cherukuri (2024), se puede inferir que los costos asociados al despliegue de sistemas de Inteligencia Artificial varían de acuerdo al punto desde donde se ejecutan los procesos de análisis y clasificación: en la nube o directamente en el dispositivo.

En nuestro caso se ha planteado que la detección de estafas telefónicas depende de la velocidad de respuesta y trabajo en tiempo real. Esta necesidad debe equilibrarse con el costo económico. Procesar la información localmente puede implicar una inversión inicial en desarrollo, pero baja los costos variables por uso de APIs o minutos de procesamiento de infraestructura en la nube.

El paradigma “Cloud-Only” centraliza todo el procesamiento (ASR, IA, TTS) en servicios remotos ofrecidos en la nube. AWS ,por ejemplo, los comercializa bajo el modelo de pago por uso, lo que garantiza escalabilidad pero sus cargos variables

pueden complejizar la tarea de realizar un presupuesto acertado (Ver Tabla 6). Así también presenta latencias más altas (Cherukuri, 2024).

Componente	Proveedor / Tecnología	Unidad de cobro	Costo (USD)	Descripción
ASR	AWS Transcribe	Por minuto de audio	0.024 USD / minuto	Transcripción estándar
TTS	Amazon Polly	Por millón de caracteres	4 o 16 USD por Millón de caracteres	Dependiendo de la voz (estándar o neuronal).
IA	OpenAI GPT-4o mini / Azure OpenAI	Por millón de tokens	0.15 USD input / 0.60 USD output	Evaluación y clasificación de mensajes sospechosos.
Storage	AWS S3	Por GB	0.02–0.12 USD / GB	Logs, audios y datos temporales.
Orquestación	AWS Lambda + API Gateway	Por millón de solicitudes	0.20 USD (aprox.)	Manejo de las peticiones

Tabla 6. Costos de referencia - Servicios en la nube. Fuente: Elaboración propia con fines académicos basados en datos obtenidos en noviembre de 2025.

Ejemplo estimativo (10000 llamadas de 3 minutos x mes):

- ASR Cloud = 10000 × 3 × 0.024 = 720 USD x mes
- TTS aproximado 5 USD x mes
- Infraestructura y almacenamiento aproximado 50 USD x mes
- IA = 10 000 × 150 tokens = (1.5 millones x 0.15) +(1.5 millones x 0.6) = 1,125 USD
- Total aproximado: 775 USD x mes

En el enfoque híbrido se combina procesamiento local en el dispositivo con servicios en la nube únicamente cuando la confianza del modelo local es baja o se

requiere validación adicional. Según la simulación realizada esto puede reducir severamente el uso de recursos cloud

Ejemplo estimativo (10000 llamadas de 3 minutos x mes, enviando solo el 20% de los casos):

- ASR (720 x 20 %): 144 USD.
- TTS (5 x 20 %) :1 USD
- IA (1,125 x 20 %) :0.23 USD
- Infraestructura (Lambda, S3, Gateway) : 30 USD
- Total aproximado: 175 USD x mes

En el enfoque "Edge-Only" delegamos la totalidad del procesamiento al dispositivo móvil. Los componentes de reconocimiento de voz, análisis de texto e interacción vocal funcionan localmente sin conexión a internet.

Al seleccionar componentes disponibles en el dispositivo móvil como así también softwares libres el costo de uso que tendríamos sería 0 USD por tal motivo no se realiza un ejemplo estimativo.

3.2.2.5 Definición del enfoque

Basado en la evaluación realizada la propuesta adoptará un enfoque "Edge-Only". Esta decisión está asociada a las ventajas que ofrece sobre las necesidades expresadas como la velocidad de respuesta en la intervención de casos de vishing o el alineamiento con los principios declarados en el trabajo. Otro motivo es la minimización de riesgos de exposición de datos sensibles y el control de los costos.

Como parte de una evolución futura, es posible considerar la adopción de un esquema híbrido que permita mejorar la precisión del sistema y ampliar su capacidad de respuesta, sin incurrir en los costos operativos de un servicio completamente basado en la nube.

3.2.3 Flujo operacional del modelo

Como se estableció en la propuesta, el modelo no se limita a la clasificación de un mensaje aislado, sino que opera como un intermediario entre el usuario y el interlocutor durante una comunicación VoIP . Acorde a esto, ProxIA actúa como una capa de mediación que recibe el flujo de audio entrante, lo procesa de manera incremental y toma decisiones dinámicas en función del riesgo estimado

La operación se basa en la captura del stream (del inglés flujo) de audio en segmentos cortos . Cada segmento es transcrito mediante el módulo de ASR para luego ser evaluado por el clasificador de inteligencia artificial. A partir de cada evaluación el sistema actualiza un estado de riesgo asociado a la sesión completa, acumulando evidencia a medida que la conversación avanza.

Cada ventana de captura tendrá una duración que dependerá de factores como la latencia del ASR, el tiempo de inferencia del modelo y capacidades del dispositivo. La calibración de estos parámetros requieren pruebas controladas en un entorno de desarrollo real. Debido al carácter conceptual de este apartado se asume un procesamiento en intervalos cortos para mantener una demora en la respuesta compatible con una interacción telefónica natural.

En función del nivel de riesgo estimado, el modelo adopta diferentes comportamientos:

- Riesgo bajo: Al detectar señales consistentes de legitimidad la llamada se redirige al usuario.
- Riesgo intermedio: el sistema mantiene una comunicación breve mediante respuestas y preguntas automatizadas a través del módulo de TTS con el objetivo de tener más contexto.
- Riesgo alto : Se genera un alerta al usuario y se limita a la interacción automática hasta que la comunicación finalice.

La conversación automatizada se limita a un conjunto acotado de respuestas coherentes y controladas orientadas a ganar tiempo y reducir la exposición al usuario, siempre con la premisa de no reemplazar una interacción humana. El procesamiento se realiza casi en tiempo real debido a la latencia de transcripción y recursos del dispositivo.

Para minimizar demoras en comunicaciones legítimas el modelo incorpora un mecanismo de paso rápido hacia el usuario. Esta función se activa al detectar señales consistente a la legitimidad, por ejemplo, coincidencia con una lista permitida de contactos o presencia de contenido textual catalogado como una frase habilitada.

En la figura 25 se permite apreciar la combinación del ASR, el clasificador, el módulo de control o decisión y el de TTS.

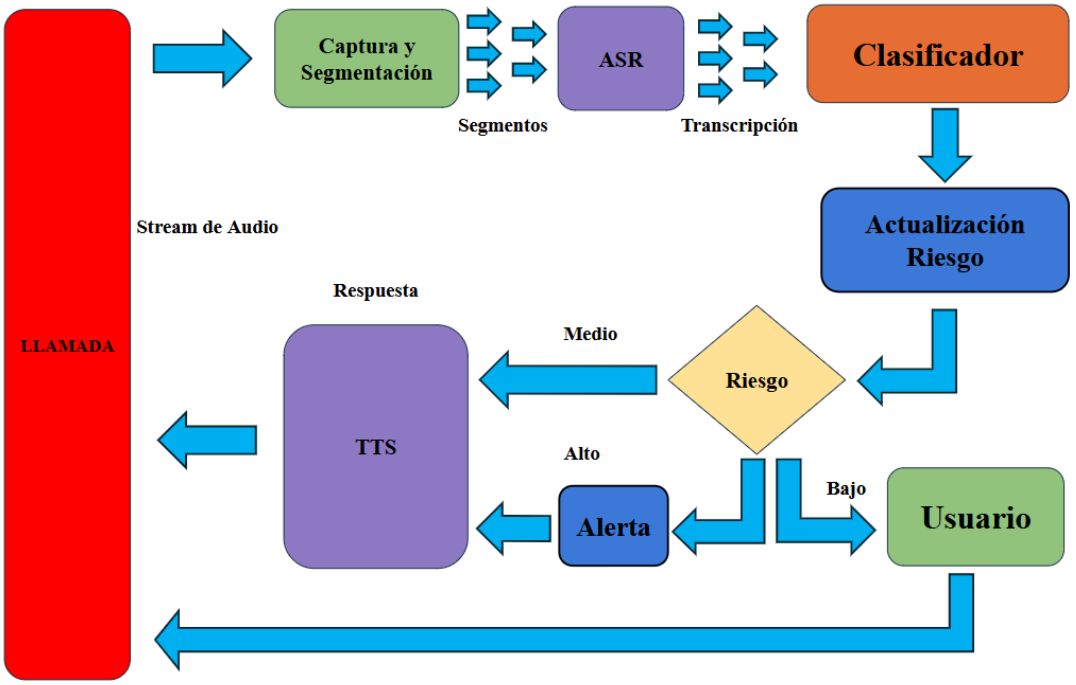


Fig. 25 Flujo de operación del modelo. Fuente: Elaboración propia con fines académicos.

La Figura 26 presenta el diagrama de componentes preliminar del sistema ProxIA, ilustrando la interacción entre los módulos internos y su rol como intermediario conversacional dentro de una arquitectura Edge-Only.

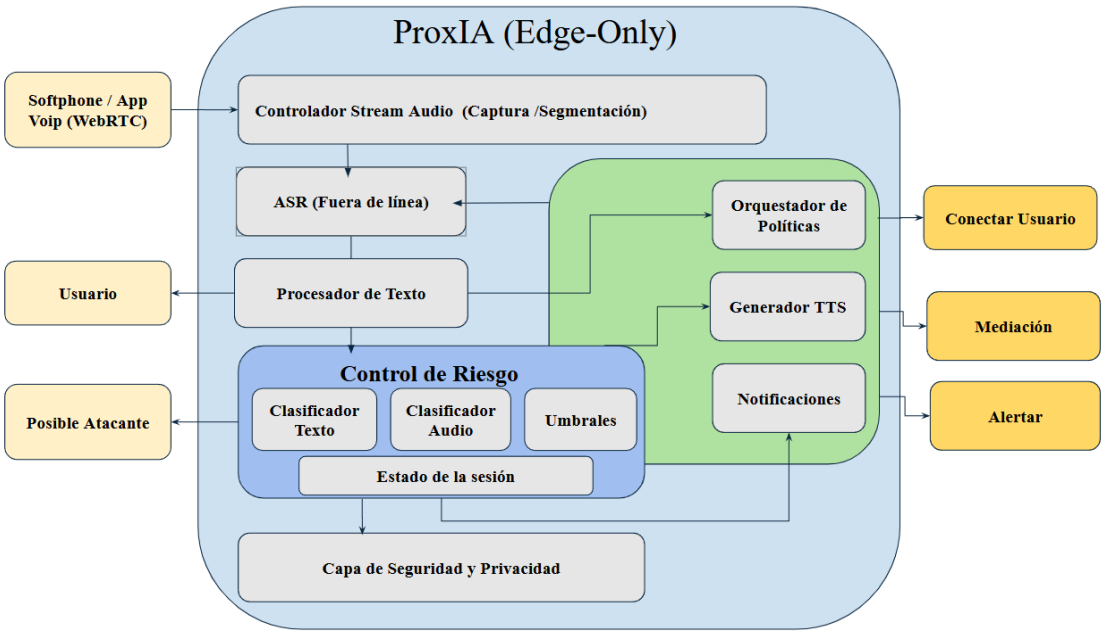


Fig. 26 Diagrama de Componentes preliminar ProxIA. Fuente: Elaboración propia con fines académicos.

3.3 Selección tecnológica de los módulos

El propósito de esta sección no es ejemplificar una implementación particular sino más bien identificar las herramientas de código libre / gratuito o comerciales que permitan materializar los componentes que conforman la propuesta de solución. La premisa es que el sistema operé bajo el enfoque Edge-Only, con compatibilidad hacia aplicaciones VoIP (SIP/WebRTC).

Este análisis, además, busca garantizar que los elementos puedan integrarse de una manera coherente en una futura aplicación operativa.

3.3.1 Criterios para el análisis

Los criterios a continuación no se plantean como métricas formales de calidad de software sino como requerimientos no funcionales y restricciones de diseño que orientan la selección tecnológica de los módulos en coherencia con la filosofía arquitectónica del modelo que se propone.

La selección se realiza a través de los siguientes criterios:

- Compatibilidad lingüística: soporte completo para el idioma español, preferentemente con adaptaciones al español latinoamericano.

- Ejecución local: capacidad de funcionar sin conexión constante a Internet, en línea con el principio de privacidad del modelo.

- Eficiencia: consumo reducido de recursos (CPU, RAM, energía), con tiempos de respuesta adecuados al análisis en tiempo real.

- Licenciamiento y accesibilidad: disponibilidad bajo licencias abiertas o comerciales de bajo costo.

- Interoperabilidad: posibilidad de integrarse en un flujo modular, utilizando formatos estándar (JSON, REST, WebSocket, API local).

- Mantenibilidad: existencia de documentación, comunidad activa y soporte multiplataforma.

Estas características permiten que la arquitectura pueda implementarse con herramientas tanto para un entorno de pruebas viable como para un prototipo con intención comercial.

3.3.2 Tecnologías analizadas

Alineado con los criterios mencionados, se identificaron diferentes alternativas tecnológicas para cada componente del modelo. A continuación se hará revisión de las mismas

3.3.2.1 Captura de audio y entorno de ejecución (VoIP / WebRTC)

Debido a que los sistemas Android e iOS restringen el acceso al canal de voz de llamadas nativas, la aplicación se orienta a operar en un entorno VoIP, donde la gestión del flujo de audio se realiza dentro del propio cliente.

Opciones evaluadas:

- WebRTC, estándar del World Wide Web Consortium (W3C): Define las APIs de comunicación de voz y vídeo en tiempo real. Permite capturar, procesar y cifrar flujos de audio mediante SRTP dentro de una aplicación.

- Linphone / Liblinphone SDK: Biblioteca de código libre compatible con Android e iOS que implementa el protocolo SIP y ofrece acceso controlado al flujo PCM dentro de la aplicación.

Características:

- Acceso bidireccional al audio del cliente.

- Integración nativa con SRTP/TLS para cifrado.

- APIs documentadas para control de llamadas.

3.3.2.2 Reconocimiento automático del habla (ASR)

El sistema requiere poder convertir audio en texto en varios idiomas de forma offline, manteniendo así la privacidad y baja latencia.

Opciones evaluadas:

- Vosk Speech Recognition Toolkit (AlphaCephei): Motor de código libre que funciona sin conexión a internet y ofrece modelos pre entrenados en español.

- Whisper.cpp: Implementación ligera/liviana en C/C++ del modelo Whisper de OpenAI, optimizada para ejecución local.

- Coqui STT: Es un kit de herramientas de aprendizaje profundo de código abierto para entrenar e implementar modelos de conversión de voz a texto.

Características:

- Ejecución local.

- Modelos en español y soporte multilingüe.

- Latencia moderada.

Resumen:

Los tres motores poseen comunidad, repositorios activos y documentados; sus modelos se ejecutan en dispositivos móviles con sistema operativo Android. Esto permite viabilidad dentro del enfoque seleccionado.

3.3.2.3 Clasificador basado en inteligencia artificial

El clasificador será el encargado de analizar el texto transcrito y determinar el nivel de riesgo de la comunicación. Para la compatibilidad con el enfoque seleccionado se requiere que tenga soporte para una ejecución en dispositivos móviles con recursos moderados.

Opciones evaluadas:

- TensorFlow Lite : Es un conjunto de herramientas que ayuda a desarrolladores a ejecutar modelos en dispositivos incorporados, móviles o de IoT, y permite implementar aprendizaje automático integrado en el dispositivo.

- ExecuTorch : Es la solución de PyTorch para la inferencia de IA en dispositivos de borde, como teléfonos móviles o sistemas integrados.

Características:

- Compatibilidad con CPU ARM y aceleradores.

- Soporte Android/iOS y APIs estables.

Resumen y aclaraciones:

Estos frameworks permiten desplegar modelos livianos con alta precisión y sin conectividad.

Dado que el módulo textual depende del componente de transcripción automática, se contempla el fallo parcial o total del ASR. En dichos casos, el sistema deberá operar en modo degradado, priorizando el análisis acústico y registrando un indicador de confianza asociado a la transcripción. Esta estrategia evitará bloquear la detección por dependencia exclusiva del texto y permite mantener continuidad operativa, aunque con potencial impacto en falsos positivos/negativos.

3.3.2.4 Síntesis de voz (TTS)

El TTS convierte texto en audio dando respuesta en una comunicación. Así como se realizó con el motor de ASR se evaluaron motores que den soporte multilenguaje y operación en el dispositivo.

Opciones evaluadas:

- **Android TextToSpeech API** : Es el motor nativo integrado en Android.

- **Piper TTS**, sistema neuronal local con voces naturales.

- **eSpeak NG**, eSpeak NG text-to-speech es un sintetizador de voz de software de código abierto compacto que admite una gran cantidad de idiomas.

Características:

- Funcionamiento offline.

- APIs para Android/Linux.

Resumen:

Android TTS nos ofrece una compatibilidad inmediata con ese sistema operativo; Piper y eSpeak NG proporcionan alternativas libres de red y personalizables. Al ser tecnologías que se ejecutan directamente en el dispositivo y

según la documentación observada nos permite inferir que la latencia baja de las mismas nos permitirá operar casi en tiempo real de una manera fluida.

3.3.2.5 Notificaciones

El módulo de notificaciones gestiona alertas locales o remotas al usuario o contactos designados.

Opciones evaluadas:

- **Firebase Cloud Messaging (FCM):** Es una plataforma gratuita para notificaciones en Android.

- **Android Notification API:** API nativa para notificaciones locales.

Características:

- Envío asíncrono y cifrado de datos.

- Integración directa con Android.

- Bajo consumo de energía.

Resumen:

Ambas soluciones son estables y ampliamente utilizadas. Permiten implementar alertas y comunicación segura sin almacenar datos personales ni depender de servidores propios.

3.3.2.6 Seguridad y privacidad

El aplicativo debe garantizar que los datos que sean procesados queden en el dispositivo, aplicando cifrado estándar.

Opciones evaluadas:

- **Android Keystore System**, sistema oficial para almacenamiento seguro de claves y certificados.

- **AES-256 (Advanced Encryption Standard)** : Estándar de encriptación definido por el Instituto nacional de estándares y tecnología (National Institute of Standards and Technology).

Características:

- Cifrado de datos en reposo.

- Claves no exportables.

- Compatibilidad con Android

Resumen:

Ambas tecnologías son nativas para Android, conocidas, robustas y de libre implementación

3.3.3 Conclusiones

Las tecnologías identificadas son existentes, documentadas y accesibles.Su compatibilidad con el enfoque Edge-Only permite su ejecución fuera de línea, además de poseer costo nulo debido a que son de código libre .

Ahora bien, aunque varias de las opciones analizadas poseen bibliotecas compatibles con iOS (como WebRTC, Linphone o frameworks de inferencia como ExecuTorch), podemos observar que la plataforma presenta restricciones adicionales que limitan la implementación práctica del enfoque propuesto en dispositivos Apple.

Estas limitaciones no son técnicas sino que están asociadas a las políticas de uso y al diseño del propio sistema operativo.

iOS no permite acceso directo al flujo de audio de llamadas. Según la documentación oficial de CallKit, la aplicación solo permite gestionar el estado de la llamada.

Más allá de que la ejecución de modelos locales de IA es técnicamente posible con las herramientas enlistadas en este trabajo, sus capacidades limitadas a trabajar procesando el audio de manera incremental, con un comportamiento cercano al tiempo real siguiendo el flujo de la figura 23.

Apple aplica políticas estrictas sobre interacción con APIs de micrófono y notificaciones, y no contamos con las mismas herramientas que Android posee para el manejo de las notificaciones de aplicaciones como WhatsApp, y que nos da acceso a los textos visibles.

Por estas razones, al momento de escribir este documento encontramos que si bien la arquitectura conceptual del modelo ProxIA es independiente de la plataforma, su implementación práctica resulta evidentemente más viable en Android, dado que :

- Android nos permite integrar un softphone VoIP con acceso al flujo PCM (vía WebRTC o Liblinphone).

- Android permite ejecutar ASR, clasificación y TTS en el dispositivo sin restricciones adicionales.

- Android permite leer el contenido visible de las notificaciones (por ejemplo, WhatsApp), posibilitando análisis textual no intrusivo.

El sistema podría extenderse a iOS en caso de que Apple otorgue flexibilidad para el acceso al canal de audio en aplicaciones .

3.3.4 Aclaraciones adicionales

Las tecnologías que fueron citadas en este apartado tienen su correspondencia con documentación oficial, repositorios públicos y estándares reconocidos, disponibles en línea y verificables. Se incluyen en el apartado de referencia sus enlaces con fines de mantener la transparencia académica.

Por último se remarca que estas herramientas no forman parte de una selección final, sino que son resultado de la búsqueda de una compatibilidad tecnológica sobre la cual podría desarrollarse la implementación de la solución propuesta para este trabajo o para integraciones futuras.

3.4 Viabilidad del modelo

Se evalúa la viabilidad del modelo a través de las siguientes perspectivas: técnica, económica, regulatoria y de madurez tecnológica.

Debido a que el objetivo principal del proyecto no es construir una aplicación completa sino proponer un modelo de solución a la problemática planteada y atrás uno), la evaluación se centra en la factibilidad de implementar el sistema bajo las condiciones y tecnologías seleccionadas, sin asumir capacidades no comprobadas ni requerir pruebas de laboratorio completas.

3.4.1 Viabilidad técnica

El análisis de viabilidad técnica se fundamenta en las capacidades requeridas por los diferentes módulos del modelo, particularmente aquellos asociados al procesamiento local del audio y la inferencia del modelo de inteligencia artificial.

El reconocimiento de voz offline, la clasificación automática de mensajes y la síntesis de voz implican una carga de cómputo y memoria que excede las capacidades de dispositivos móviles antiguos o de gama baja.

En consecuencia, la implementación resulta recomendable en dispositivos con arquitectura ARM de 64 bits, al menos 4 GB de memoria RAM y versiones recientes del sistema operativo Android, donde las APIs de audio, seguridad y ejecución de modelos de IA se encuentran más maduras y estables. Estas especificaciones no se plantean como requisitos mínimos garantizados, sino como una estimación técnica conservadora de acuerdo con la documentación disponible de las herramientas seleccionadas y sus requerimientos de ejecución en entornos móviles.

El modelo propuesto se basa en una arquitectura Edge-Only, donde todos los procesos se ejecutan en el dispositivo del usuario y apuntada a entornos VoIP.

Esto es técnicamente viable dado que:

Todas las tecnologías seleccionadas existen, están documentadas y son compatibles con este tipo de arquitectura.

- WebRTC y Liblinphone permiten la gestión de audio en aplicaciones VoIP sin depender del sistema de telefonía móvil.

- Motores ASR como Vosk, Whisper.cpp y Coqui STT proporcionan reconocimiento de voz offline en español.

- Frameworks como TensorFlow Lite y ExecuTorch posibilitan la ejecución local de modelos livianos de clasificación.

- Motores TTS como Android TextToSpeech, Piper o eSpeak-NG permiten síntesis sin conexión.

- Android Keystore y AES-256 aseguran el almacenamiento seguro y el cifrado local.

El procesamiento local evita dependencias de infraestructura externa. El audio se procesa de manera incremental en tiempo casi real, mediante técnicas de transcripción por segmentos y evaluación continua del contenido semántico.(Ver figura 27)

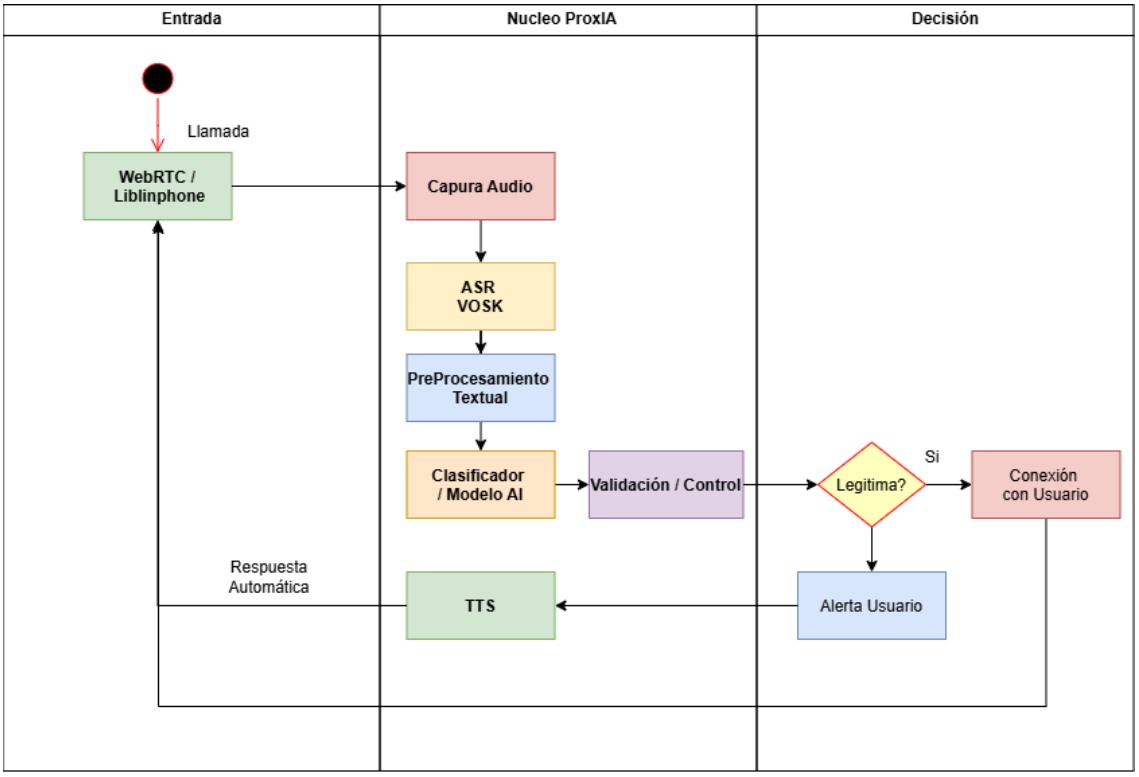


Fig. 27 Diagrama para toma de decisión del modelo. Fuente: Elaboración propia con fines académicos con herramienta Draw.io

Como ningún componente crítico requiere conectividad permanente, el funcionamiento del modelo se limita a los recursos del dispositivo, lo que reduce la complejidad y evita latencias propias de la nube.

La arquitectura modular permite que cada componente funcione de manera independiente.

Aunque iOS presenta restricciones adicionales el diseño se mantiene válido y la implementación inicial se centra en Android, donde las restricciones son menores y el acceso a ASR, TTS y VoIP es más flexible.

Sobre el mantenimiento y evolución del modelo, la arquitectura modular permite actualizar de forma independiente los distintos componentes, tales como los motores de reconocimiento de voz, los modelos de clasificación o los sistemas de síntesis. Asimismo, la detección de capacidades del dispositivo permite limitar la ejecución del sistema a entornos compatibles, evitando la instalación en hardware no apto. De este modo, el mantenimiento se orienta a la actualización controlada de módulos y modelos, sin comprometer la estabilidad del conjunto ni requerir infraestructura centralizada.

Conclusión

El diseño es técnicamente viable dentro del alcance definido, siempre que la implementación se realice en entornos VoIP controlados. Además su arquitectura modular la hace extensible y flexible a la incorporación de nuevos casos de uso mediante la aplicación de principios de ingeniería y diseño de software. Por ejemplo, se podrían implementar mecanismos de validación por palabra clave para permitir el acceso directo al usuario final, políticas de transferencia condicional o listas de números previamente autorizados.

3.4.2 Viabilidad económica

La selección de los componentes valora las herramientas de código libre o gratuitas. Según el análisis también expresado en este trabajo demuestra que esto reduciría prácticamente a cero los costos de implementación de un prototipo.

El enfoque de Edge-Only no necesita servidores de procesamiento ni recursos en la nube, evitando gastos en:

- Transcripción remota
- Clasificación de IA por API,
- Almacenamiento en nube
- Servicios SaaS.

Los costos que podrían encontrarse son los siguientes:

- Dispositivos de prueba, como móviles o computadoras personales con emuladores.
- Servicios VoIP provistos por proveedores de telefonía locales o internacionales.
- Tiempo invertido en el desarrollo.

Conclusión:

La solución es económicamente accesible y no implica costos recurrentes significativos.

3.4.3 Viabilidad regulatoria y de privacidad

Al comienzo de este trabajo se han mencionado aspectos legislativos y regulatorios que, potencialmente, podrían generar conflictos con el tipo de solución propuesta. También se estableció que la exploración profunda de estos marcos jurídicos no formarían parte del alcance de esta investigación. No obstante se han seleccionado herramientas que permitan diseñar un modelo alineado con normativas y políticas de privacidad. Es por eso que se deja constancia que la solución :

- No se interceptan llamadas del sistema móvil.
- No se accede a datos de aplicaciones cerradas.
- Todo el procesamiento ocurre en el dispositivo.
- Solo se implementará a través de aplicaciones VoIP o del texto visible de notificaciones .
- Se solicitará y respetará el consentimiento del usuario.
- Se utilizan mecanismos de encriptación.

Conclusión:

El modelo es viable dentro del entorno y alcance definidos, dado que no evade restricciones ni requiere acceder a datos protegidos.

3.4.4 Viabilidad operativa

La propuesta se encuentra dentro de un nivel de madurez tecnológica coherente debido a que el diseño propuesto puede llegar a un prototipo real. A continuación se explaya los puntos claves que me permiten llegar a esta afirmación.

- Solidez conceptual: El diseño expuesto integra componentes existentes, probados y usados por la comunidad.

- Escalable: El diseño modular permite integrar componentes más avanzados sin redefinir la base de la arquitectura.

- Operación viable: La ejecución completa propuesta sería más factible en Android, lo cual es acorde a las herramientas seleccionadas.

- Bajo desarrollo : Es posible utilizar aplicativos ya desarrollos como softphones trabajando con sus bibliotecas.

- Implementación parcial: El clasificador a crear como "MVP" permitirá comprobar la efectividad de la inteligencia artificial en el reconocimiento de comunicaciones riesgosas.

Conclusión:

La propuesta se encuentra en un nivel adecuado para su validación y puede evolucionar hacia prototipos más avanzados sin modificar su estructura base.

3.4.5 Conclusión integral

Como síntesis se llega la conclusión que la propuesta es viable dentro del alcance planteado porque:

- Usa tecnologías existentes y accesibles.
- Respeta las restricciones técnicas y normativas.
- Puede ser demostrada parcialmente mediante un MVP del clasificador.
- Su arquitectura es modular, clara y escalable.
- No depende de infraestructura externa ni de capacidades no disponibles.

Se vuelve a mencionar que la limitada compatibilidad con iOS no invalida ni debilita el modelo, sino que acota el ámbito operacional de la primera implementación, lo cual es aceptable y esperable para esta tesis que tiene como objetivo la propuesta de solución y no así su desarrollo.

4. MVP : CLASIFICADOR DE AUDIOS

La siguiente sección presentará el alcance y las características propias del producto mínimo viable (MVP por sus siglas en inglés) elegido como elemento práctico de validación para las afirmaciones expuestas en este trabajo.

Cabe aclarar que los siguientes apartados no tienen como objetivo describir en detalle el funcionamiento interno de las herramientas, librerías o plataformas utilizadas, ni justificar exhaustivamente cada decisión técnica a nivel de implementación. El foco del trabajo se encuentra en exponer de manera clara el proceso seguido, las decisiones metodológicas adoptadas y los resultados obtenidos en el desarrollo.

Las tecnologías empleadas ,como los entornos de ejecución, frameworks y bibliotecas de procesamiento, cuentan con amplia documentación, comunidades activas y literatura disponible, por lo que su explicación detallada excede el alcance de esta tesis. En línea a lo expresado, se optó por una descripción orientada a explicar su uso dentro del experimento, haciendo énfasis en la comprensión del tópico general por sobre el detalle técnico de cada componente.

4.1. Objetivo del MVP

El objetivo del MVP es desarrollar un prototipo funcional que permita evaluar la factibilidad de poder clasificar llamadas o partes de audio como “riesgosos o sospechosos” o lo contrario. Todo esto dentro del contexto ya presentado de estafas telefónicas basadas en ingeniería social o “Vishing”.

El modelo operará como experimento en entorno controlado y no se aplicará para detectar estafas en tiempo real ni operará sobre dispositivos móviles. Con él se buscará demostrar de manera práctica que existen patrones identificables que un algoritmo pueda aprender.

Esto quiere decir que el resultado servirá como base empírica para justificar las etapas posteriores de una posible implementación más avanzada.

Lo expresado anteriormente se alinea con la concepción del MVP como instrumento de experimentación temprana para reducir incertidumbre y validar hipótesis en contextos tecnológicos (Moogk, 2012).

4.2 Alcance

El MVP se enfoca únicamente en las etapas fundamentales necesarias para demostrar la viabilidad técnica mínima del proyecto, es por eso que sus objetivos secundarios serán:

- Entrenar clasificadores en un entorno aislado y reproducible (Google Colab).

- Aplicar técnicas estándar de procesamiento de audio y, de forma exploratoria, análisis textual a partir de transcripciones automáticas.

- Utilizar frameworks consolidados como TensorFlow y Keras, orientados al prototipado rápido y a la investigación aplicada.

- Generar métricas cuantitativas de evaluación (accuracy, F1-score y matriz de confusión) para analizar el desempeño del modelo.

- Comparar distintos enfoques de entrada (audio crudo, representaciones acústicas mediante MFCC y señales textuales) con el fin de observar su impacto en la clasificación.

En este sentido, el MVP no pretende reproducir el funcionamiento completo del sistema propuesto, sino validar el núcleo técnico de clasificación sobre el cual se sustenta la arquitectura conceptual presentada en el capítulo anterior.

Queda fuera de alcance:

- El procesamiento en tiempo real.

- El acceso a audio de llamadas reales en aplicaciones oficiales para este fin en móviles (no viable en iOS/Android por restricciones OS).

- Una integración con arquitectura móvil.

- La identificación de personas o locutores específicos.

El MVP se implementará utilizando audios propios, simulados o provenientes de datasets públicos debidamente etiquetados y con consentimiento de uso. Esta delimitación permite mantener el control experimental, respetar consideraciones éticas y concentrar el análisis en el comportamiento del modelo bajo condiciones conocidas.

Desde esta perspectiva, el producto resultante cumple el rol de validar de manera parcial y controlada la propuesta técnica de la tesis, actuando como puente entre el marco teórico desarrollado en capítulos anteriores y las discusiones sobre resultados, limitaciones y posibles extensiones que se presentan en las secciones siguientes.

4.3 Enfoque del MVP

Para una mejor comprensión sobre los caminos elegidos en este proceso de clasificación de audios vinculados a estafas telefónicas esta sección enlista los aspectos técnicos específicos que fundamentan el enfoque adoptado en el MVP.

Es importante destacar que en capítulos anteriores ya se han introducido conceptos generales como inteligencia artificial, aprendizaje automático, ingeniería social y modalidades de fraude. Por tal motivo ,y con el fin de evitar cualquier tipo de redundancia, no serán desarrollados con la misma profundidad.

4.3.1 Ingeniería social telefónica y su impacto en el análisis de audio

Como se trató previamente, la ingeniería social es un proceso en la comunicación que puede producirse a través de tácticas intimidatorias como también en imitaciones verosímiles de entidades legítimas (hacerse pasar por el representante de un banco). Particularmente en el vishing, los ataques no solo dependen del tono o la emocionalidad del atacante. Las estafas pueden ser ejecutadas con una comunicación neutral, profesional y sin parámetros acústicos que indiquen una amenaza .

Esto quiere decir que:

- Los indicadores acústicos no son suficientes para identificar una estafa. Aunque es verdad que ciertos tipos sí presentan patrones prosódicos detectables como la presión temporal, urgencia, nerviosismo, agresividad.

- El contenido de la comunicación es clave dado que encontraremos señales particulares que nos permitirán identificar una amenaza (por ejemplo, solicitudes de contraseñas, tokens, códigos SMS; algo prohibido en protocolos bancarios).

El acercamiento planteado como entregable no intenta resolver todos estos escenarios, sino que su propósito es validar, en un entorno controlado, la existencia de patrones acústicos y textuales que puedan ser aprendidos por un modelo automático, y analizar en qué medida cada enfoque contribuye a dicha tarea.

4.3.2 Justificación del enfoque acústico y textual

El enfoque adoptado se definió a partir de una serie de supuestos iniciales vinculados a la naturaleza de las estafas que estamos tratando. En particular, se partió de la premisa de que este tipo de fraudes podría manifestar señales tanto en la forma de la comunicación como en su contenido, y que ninguna de estas dimensiones consideradas de manera aislada, garantiza por sí misma una identificación definitiva y acertada.

En una primera instancia, se decidió priorizar el análisis acústico del audio, dado que este permite trabajar directamente sobre la señal de voz sin necesidad de interpretar el significado del mensaje. Este enfoque fue considerado adecuado para una etapa exploratoria, en la medida en que facilita la extracción de características objetivas y reduce la dependencia de procesos externos como la transcripción automática.

Paralelamente, se asumió que el contenido de la comunicación podría aportar información relevante en determinados escenarios, particularmente en aquellos en los que las señales acústicas resultan ambiguas o poco distintivas. A partir de este supuesto, se sumó al diseño del clasificador la incorporación de un análisis textual de carácter exploratorio, basado en transcripciones automáticas, con el objetivo de evaluar su contribución al proceso de clasificación.

Todo esto responde a una lógica incremental, en la que se implementaron y evaluaron distintos tipos de representaciones de entrada de manera controlada. Los resultados obtenidos a partir de estas decisiones serán presentados y analizados en las secciones correspondientes, permitiendo discutir el alcance y las limitaciones de lo propuesto.

4.4 Herramientas y entorno de ejecución

Para el desarrollo y la ejecución de los experimentos se utilizó Google Colab (Plataforma gratuita basada en la nube de Google que permite escribir y ejecutar código Python en un navegador web) como herramienta de trabajo. Esta elección permitió tener a disposición un entorno para operar con soporte nativo de bibliotecas de aprendizaje automático y acceso a recursos de cómputo, sin requerir configuraciones locales adicionales.

La ejecución se basó en Python versión 3.1, utilizando TensorFlow como framework (entorno de trabajo) principal para el entrenamiento de los modelos, junto con su API Keras para la definición de arquitecturas y la gestión del proceso de aprendizaje.

En las tareas de procesamiento y análisis se emplearon bibliotecas complementarias como Librosa para el manejo de señales de audio, NumPy para la manipulación de datos numéricos, Matplotlib para la visualización de resultados y Scikit-learn para el cálculo de métricas de evaluación. El uso de Google Drive permitió organizar el almacenamiento y acceso a los archivos de audio utilizados durante las pruebas.

Los recursos ofrecidos por la herramienta “Colab” resultaron suficientes para el entrenamiento de redes neuronales convolucionales de complejidad moderada, sin afectar la viabilidad ni la reproducibilidad de los experimentos.

La preparación del entorno incluyó la instalación de dependencias, y la configuración de rutas de acceso a los datos, estableciendo una base adecuada para la ejecución de las distintas pruebas realizadas.

4.5 El dataset (Conjunto de datos)

Uno de los desafíos a superar en esta investigación fue la dificultad de encontrar ejemplos públicos de estafas telefónicas en español.

Esto se debe a restricciones legales, riesgos de privacidad y obligaciones de consentimiento explícito. Dichas consideraciones están ampliamente documentadas en análisis éticos recientes sobre datasets de voz, como el trabajo de Papakyriakopoulos et al. (2023), que expone la necesidad de metadatos, documentación ampliada y procesos de gobernanza para mantener estándares éticos en datos de habla.

Por lo anterior mencionado se optó por generar datos simulados . La utilización de datos sintéticos es metodológicamente válida en etapas exploratorias, especialmente bajo restricciones éticas. Raghunathan (2021) establece que los datasets sintéticos permiten evaluar comportamiento del modelo y reproducibilidad sin exponer datos sensibles.

El dataset simulado fue generado mediante de las siguientes tareas :

- Crear 100 guiones divididos en 50 frases que podemos encontrar en estafas y 50 en comunicaciones legítimas.

- Grabar los guiones como audios de diferentes voluntarios para lograr introducir diversidad de voces, entonaciones y estilos.

El uso de 100 audios responde al nivel exploratorio del MVP. No se buscó construir un conjunto de datos representativo del fenómeno real, sino contar con un dataset controlado, balanceado y etiquetado con precisión que permitiera evaluar la viabilidad del enfoque propuesto.

El resultado permitió tener:

- Control total sobre las etiquetas.
- Trabajar con diferentes voces.
- Un dataset balanceado y adecuado para el MVP.

La combinación de guiones estructurados y múltiples voces contribuyó a generar un conjunto de muestras suficientes para una primera aproximación experimental.

4.6 Preprocesamiento de audio

Los audios utilizados fueron registrados inicialmente mediante aplicaciones de mensajería y almacenados en su mayoría en formato Opus y MP4. Con el fin de garantizar consistencia en el procesamiento y compatibilidad con las bibliotecas empleadas, se realizó una conversión previa de todos los archivos a formato WAV, en canal mono y con una frecuencia de muestreo uniforme de 16 kHz. El propósito de la estandarización fue bajar la diversidad no deseada en la entrada del modelo.

Adicionalmente, se llevó a cabo una normalización del volumen de los audios con el propósito de evitar que las diferencias en amplitud influyeran de manera indebida en el proceso de aprendizaje.

Por último para asegurar entradas de dimensión consistente al modelo, se estableció una duración fija para los audios mediante técnicas de padding o trimming. Esta normalización temporal permitió construir tensores (estructuras de datos básica utilizada en entornos de aprendizaje automático como TensorFlow) de tamaño homogéneo y facilitando el entrenamiento de la red neuronal convolucional.

A partir de los audios preprocesados se extrajeron coeficientes cepstrales en las frecuencias de Mel (MFCC) (ver figura 28), utilizados como representación acústica de entrada para el modelo. Los MFCC constituyen una técnica de extracción de características ampliamente utilizada en reconocimiento automático del habla, basada en la percepción auditiva humana y diseñada para representar de forma compacta el espectro de potencia de corto plazo de una señal de voz (Singh & Rani, 2014).

El proceso de extracción de MFCC comprende etapas de pre-énfasis, segmentación temporal en ventanas cortas, aplicación de funciones de ventana, transformación al dominio frecuencial mediante la transformada rápida de Fourier, filtrado a través de un banco de filtros distribuidos según la escala Mel y, finalmente, una transformación cepstral discreta que genera un conjunto reducido de coeficientes representativos de la envolvente espectral del sonido (Singh & Rani, 2014). Esta representación permite capturar información fonética relevante de manera eficiente, reduciendo la influencia de ruido y variaciones no deseadas.

Esta técnica es comúnmente adoptada en tareas de reconocimiento y clasificación de voz por su capacidad para capturar información fonética relevante de manera compacta y eficiente (Ittichaichareon et al., 2012). Asimismo los MFCC ofrecen un mejor equilibrio entre capacidad discriminativa, complejidad computacional y requerimientos de datos, resultando particularmente apropiados para sistemas de clasificación automática basados en aprendizaje profundo (Singh et al., 2012).

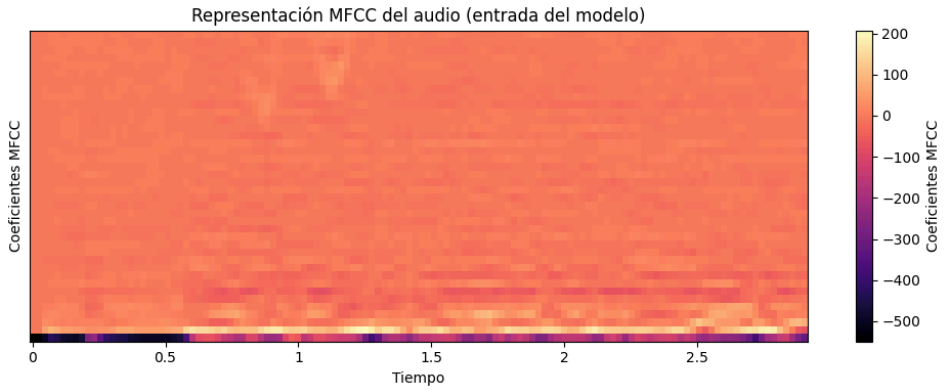


Fig. 28 Representación MFCC utilizada como entrada del modelo. Fuente: Elaboración propia con fines académicos.

En este trabajo se utilizaron 40 coeficientes MFCC por muestra, valor comúnmente empleado para modelos convolucionales aplicados a datasets de tamaño reducido. La elección otorgó una representación adecuada para el entrenamiento del clasificador.

4.7 Pipeline

El pipeline fue diseñado buscando equilibrar la simplicidad y la ejecución de las tareas necesarias para la clasificación de audio.

En primer paso, los audios fueron cargados desde el repositorio remoto para aplicar el preprocesamiento mencionado en anterior sección, y así dar con las condiciones necesarias de entrada. Luego, se continuó con la extracción de características acústicas mediante coeficientes MFCC, generando representaciones de dimensión fija adecuadas para su utilización como entrada del modelo.

En paralelo, y con fines exploratorios, se incorporó un flujo complementario basado en las transcripciones automáticas del audio. Estas permitieron construir representaciones textuales simples, utilizadas para evaluar de manera preliminar el aporte del contenido semántico al proceso de clasificación. Este componente fue

integrado al flujo de trabajo sin modificar la estructura general del experimento, conservando la posibilidad de tener una comparación entre enfoques.

Las representaciones obtenidas fueron organizadas en tensores y divididas en conjuntos de entrenamiento, validación y prueba, siguiendo una proporción fija. Posteriormente, se llevó a cabo el entrenamiento del modelo utilizando las configuraciones definidas, registrando métricas estándar para evaluar su desempeño.

Finalmente, el proceso incluyó una etapa de validación sobre el conjunto de pruebas y evaluaciones de inferencia sobre audios no utilizados durante el entrenamiento, con el objetivo de observar el comportamiento general del modelo frente a nuevas muestras. Esto permitió analizar de manera ordenada el impacto de las distintas representaciones de entrada como también obtener la interpretación de los resultados presentados en secciones posteriores.

La figura 29 es una representación del pipeline utilizado, resumiendo las principales etapas del proceso.

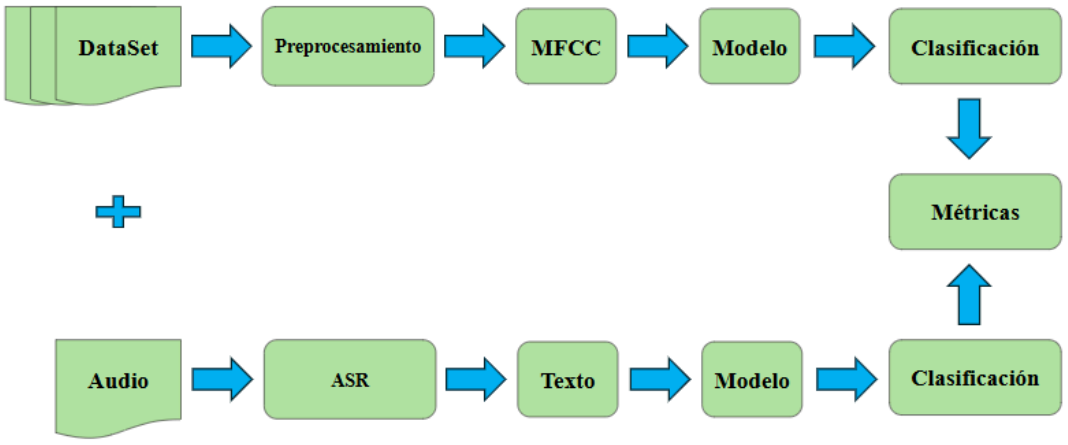


Fig. 29 Representación de pipeline MPV. Fuente: Elaboración propia con fines académicos.

4.8 Modelo implementado

Para la clasificación de audio se utilizó un modelo de red neuronal convolucional bidimensional (CNN 2D), seleccionado por su adecuación para procesar representaciones acústicas en forma de matrices tiempo-frecuencia, como las obtenidas a partir de coeficientes MFCC. En particular, Reggiswarashari y Sihwi (2022) aplican este tipo de arquitectura sobre representaciones MFCC segmentadas para reconocimiento en señales de voz, utilizando dichas matrices como entrada bidimensional al modelo.

En las representaciones MFCC, cada muestra de entrada puede interpretarse como una imagen acústica, donde un eje corresponde a la evolución temporal de la señal y el otro a las distintas bandas cepstrales, permitiendo a la red aprender patrones locales tanto en el dominio temporal como en el frecuencial.

Las redes convolucionales 2D son ampliamente utilizadas en tareas de clasificación de audio y reconocimiento de patrones acústicos. lo que lo convierte en una opción adecuada para una etapa exploratoria. En contraste, las arquitecturas convolucionales unidimensionales suelen emplearse cuando la entrada corresponde a señales temporales crudas, lo que no es el caso en este trabajo, dado que se realizó una etapa previa de extracción de características.

El modelo fue entrenado como un clasificador binario, diferenciando entre audios etiquetados como sospechosos y audios legítimos. Para el proceso de aprendizaje se utilizaron configuraciones estándar de entrenamiento empleadas habitualmente en este tipo de problemas, priorizando la estabilidad y la reproducibilidad por sobre la optimización fina de hiper parámetros.

Para el análisis semántico se empleó un modelo base compuesto por vectorización TF-IDF seguida de un clasificador de Regresión Logística. La elección de este enfoque responde a su desempeño competitivo en datasets pequeños, su bajo costo computacional y su capacidad de generar probabilidades calibradas útiles uso en esquemas con la función multimodal, como en este caso.

La vectorización TF-IDF (Term Frequency–Inverse Document Frequency) transforma el texto transcripto en una representación numérica ponderada, destacando términos relevantes y reduciendo la influencia de palabras frecuentes de bajo contenido informativo..Esta representación resulta adecuada en tareas de minería de texto,dado que la ponderación de términos impacta directamente en la capacidad discriminativa del clasificador (Flores Caceres, 2024). En este caso se utilizaron unigramas y bigramas con un máximo de 5000 características

Posteriormente, se entrenó un modelo de Regresión Logística con balanceo de clases, adecuado para problemas binarios y estable ante conjuntos de datos de tamaño reducido. Este clasificador permite obtener probabilidades asociadas a cada clase, lo cual facilita el análisis comparativo con el enfoque acústico.

Diversos trabajos de clasificación automática de texto han demostrado que la combinación de TF-IDF + Regresión constituye una estrategia robusta y estable en contextos de alta dimensionalidad y matrices dispersas, características propias de la representación documento-terminó (Flores Caceres, 2024).

La arquitectura implementada fue elegida por su baja complejidad, en relación con el tamaño del dataset y con el objetivo del experimento. No se buscó optimizar el rendimiento del modelo ni explorar arquitecturas profundas, sino contar con la estructura suficiente para evaluar la viabilidad del enfoque propuesto.

Durante el entrenamiento se realizó la observación de las métricas básicas de desempeño con el fin de visualizar el comportamiento general del modelo y detectar posibles problemas de ajuste. Los resultados obtenidos a partir de este proceso se presentan y analizan en la sección siguiente.

La figura 30 resume el pipeline ya mencionado con los respectivos modelos implementados.

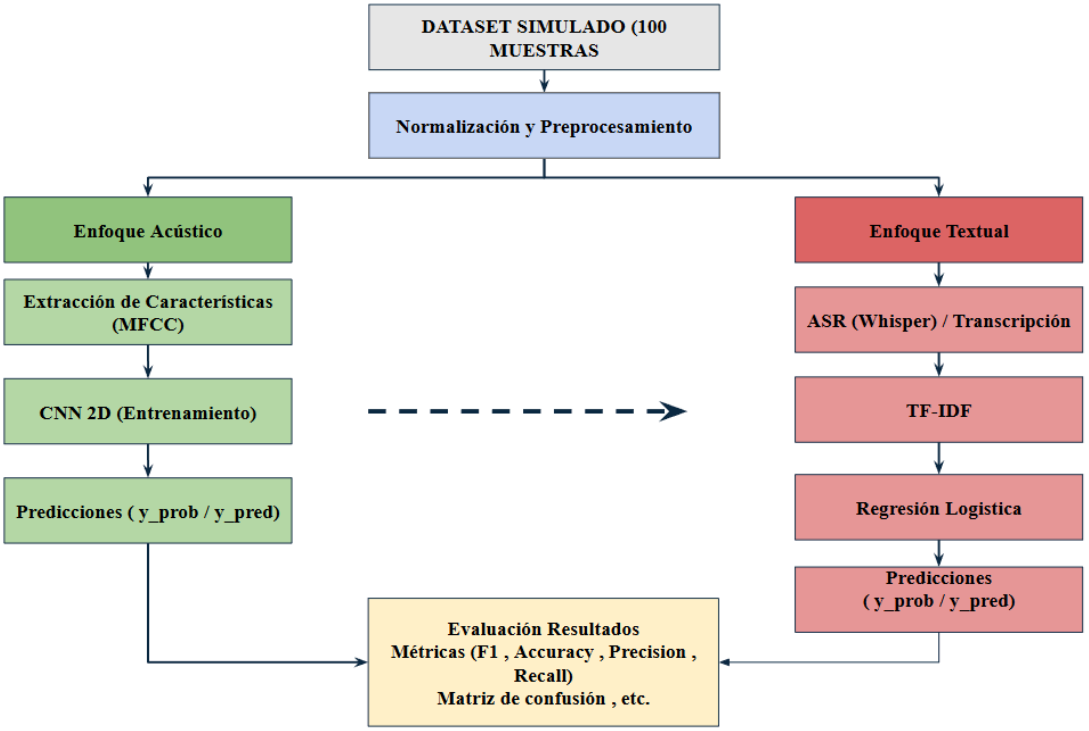


Fig. 30 Presentación de modelos usados en pipeline MPV. Fuente: Elaboración propia con fines académicos.

4.9 Resultados experimentales

En esta sección se presentan los resultados obtenidos a partir de los experimentos realizados con el clasificador implementado. Las métricas corresponden a evaluaciones sobre la base de prueba, utilizando los conjuntos de entrenamiento y validación definidos previamente.

Para la interpretación de los resultados obtenidos se debe tomar en consideración el tamaño del dataset, la naturaleza de los datos, el carácter exploratorio del enfoque adoptado y el contexto de este trabajo.

4.9.1 Métricas de evaluación

Para evaluar el desempeño del modelo se utilizaron métricas estándar para problemas de clasificación binaria, incluyendo accuracy, precisión, recall y F1-score. Estas métricas entregan una visión general del comportamiento del clasificador y de su capacidad para distinguir entre audios sospechosos y audios legítimos.

En este caso, el modelo fue ajustado para priorizar la detección de audios sospechosos, lo que influyó directamente en la distribución de errores observada.(Ver Tabla 7)

Clase	Precisión	Recall	F1-score	Muestras
Legítimo	0.00	0.00	0.00	8
Sospechoso	0.47	1.00	0.64	7
Accuracy			0.47	15
Macro avg	0.23	0.50	0.32	15
Weighted avg	0.22	0.47	0.30	15

Tabla 7. Métricas del clasificador acústico. Elaboración propia con fines académicos basados en resultados obtenidos durante la ejecución.

La matriz de confusión relacionada muestra, sobre los elementos de validación, que el modelo clasifica correctamente la totalidad de los audios etiquetados como sospechosos, pero presenta un alto número de falsos positivos, clasificando audios legítimos como sospechosos. (Ver figura 31)

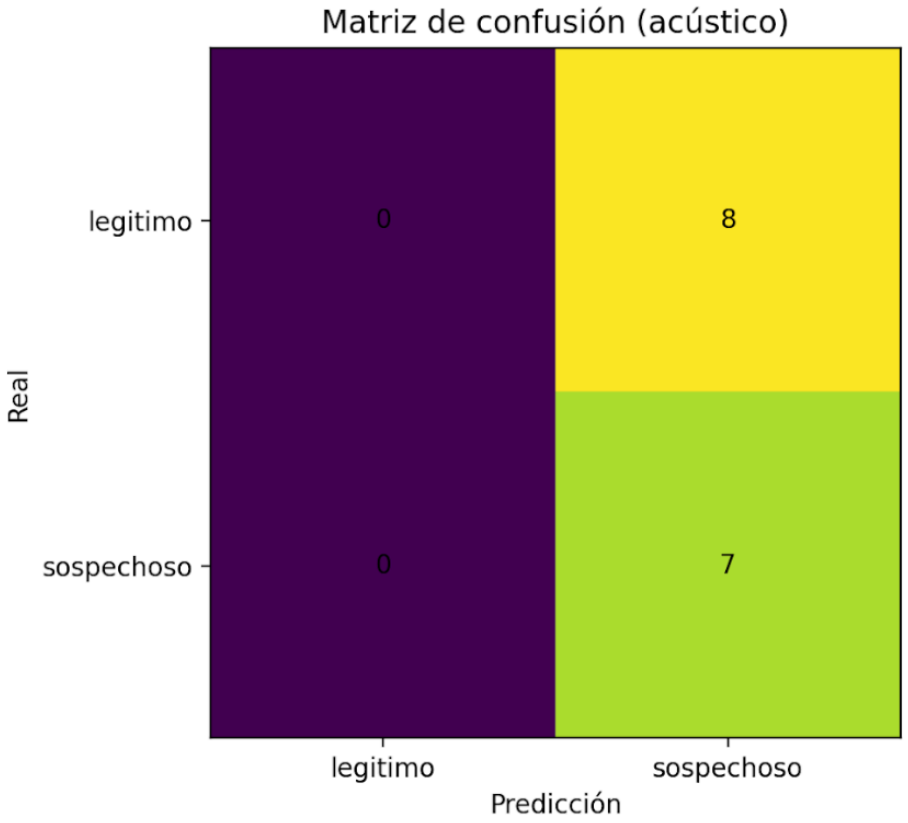


Fig. 31 Matriz de confusión del clasificador acústico. Elaboración propia con fines académicos basados en resultados obtenidos durante la ejecución.

Este comportamiento es coherente con el criterio de ajuste adoptado, que puso como objetivo minimizar los falsos negativos en la clase sospechosa. Si bien esta estrategia reduce el riesgo de no detectar una estafa, incrementa la cantidad de clasificaciones erróneas sobre comunicaciones legítimas, lo cual limita el uso del enfoque acústico como mecanismo único de detección.

4.9.2 Resultados del clasificador de texto

El clasificador basado en señales textuales fue evaluado utilizando el mismo set de datos y las mismas métricas que el enfoque acústico, con el objetivo de analizar el aporte del contenido semántico al proceso de clasificación.

Clase	Precisión	Recall	F1-score	Muestras
Legítimo	1.00	0.88	0.93	8
Sospechoso	0.88	1.00	0.93	7
Accuracy			0.93	15
Macro avg	0.94	0.94	0.93	15
Weighted avg	0.94	0.93	0.93	15

Tabla 8. Métricas del clasificador de texto. Elaboración propia con fines académicos basados en resultados obtenidos durante la ejecución.

La matriz de confusión asociada al clasificador textual se presenta en la Figura 32 y se detalla a continuación:

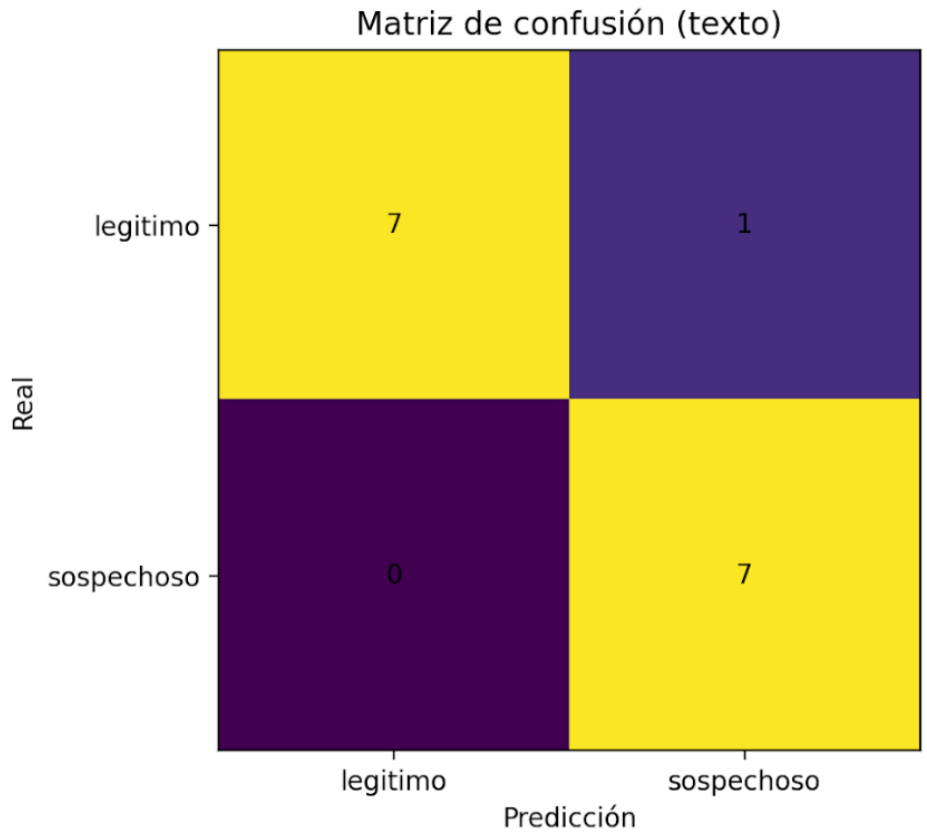


Fig. 32 Matriz de confusión del clasificador de texto. Elaboración propia con fines académicos basados en resultados obtenidos durante la ejecución.

Este resultado indica que el modelo clasifica correctamente la totalidad de los audios sospechosos presentes en el conjunto de prueba, y presenta un único error de clasificación en la clase legítima. En comparación con el enfoque acústico, se observa una reducción significativa de los falsos positivos, lo que deja ver que el contenido semántico aporta información relevante para la tarea de etiquetado y en consecuencia de detección.

El desempeño entonces obtenido refleja un comportamiento más equilibrado entre ambas clases, manteniendo altos valores de precisión y recall.

4.10 Análisis de resultados preliminares

El clasificador acústico presentó un comportamiento orientado a la detección de audios sospechosos, logrando identificar correctamente la totalidad de los casos correspondientes a esta clase en el set de test. La sensibilidad elevada se obtuvo a costa de una capacidad limitada para reconocer comunicaciones legítimas, lo que se tradujo en un número considerable de falsos positivos.

En contraste, el clasificador basado en texto mostró un desempeño más equilibrado entre ambas clases, reduciendo de manera significativa los errores sobre audios legítimos y manteniendo una alta tasa de detección de comunicaciones sospechosas. Esto comprobó el supuesto de que el contenido semántico aportaba señales discriminantes que permitieran mejorar la clasificación cuando las características acústicas resultarán insuficientes.

Cada enfoque aportó miradas diferentes del problema. El análisis acústico resulto efectivo como mecanismo sensible al riesgo, mientras que el análisis textual contribuyó a mejorar la discriminación entre clases. Vistos en conjunto, estos resultados indicaron que el uso combinado de ambas fuentes de información permitiría mitigar las limitaciones observadas en cada enfoque por separado.

4.11 Validaciones

Durante el proceso de validación se analizaron de manera cualitativa las predicciones realizadas por los clasificadores sobre muestras individuales, con el objetivo de observar patrones de acierto y error más allá de las métricas agregadas.

En el caso del clasificador acústico, se llegó a observar que las predicciones correctas sobre audios sospechosos se concentraron principalmente en muestras que presentaban variaciones prosódicas marcadas, como cambios abruptos en la entonación o énfasis en determinadas frases. Por el contrario, las clasificaciones erróneas se produjeron en gran medida sobre audios legítimos con características acústicas similares a las de las estafas simuladas, lo que explica la presencia de falsos positivos observada en las métricas globales (Ver figura 33).

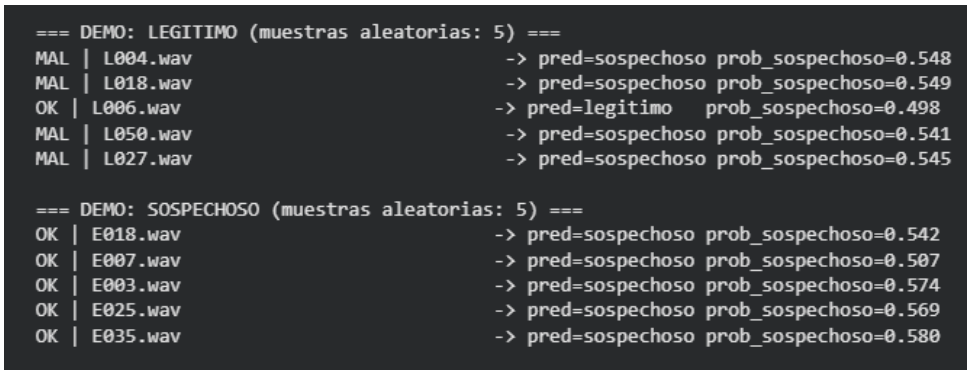


Fig. 33 Validaciones del modelo acústico. Elaboración propia con fines académicos basados en resultados obtenidos durante la ejecución.

En relación con el clasificador basado en texto, las predicciones correctas estuvieron fuertemente asociadas a la presencia de expresiones y estructuras típicas de la ingeniería social, tales como solicitudes de información sensible o referencias a

acciones urgentes. El único error registrado en el conjunto de pruebas correspondió a una comunicación legítima con una formulación ambigua, lo que sugiere que el modelo puede verse afectado por la ausencia de contexto adicional en ciertos casos.

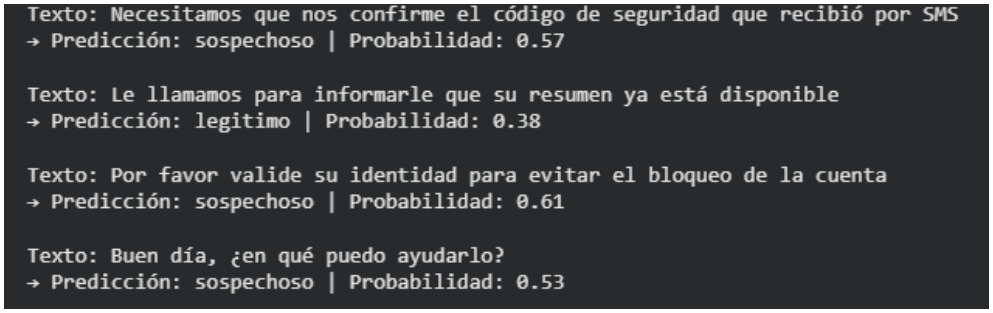


Fig. 34 Validaciones del modelo acústico. Elaboración propia con fines académicos basados en resultados obtenidos durante la ejecución.

Estas observaciones permiten complementar el análisis y refuerzan la interpretación de los resultados, evidenciando que los errores cometidos por los modelos responden a características razonables de las muestras y no a comportamientos aleatorios.

4.12 Síntesis

El desarrollo y evaluación del MVP permitió validar la factibilidad técnica del enfoque propuesto para la detección de estafas telefónicas basadas en ingeniería social. Los resultados obtenidos evidencian que el análisis acústico y el análisis textual capturan dimensiones complementarias del problema, aportando información relevante para su identificación.

Si bien el alcance entregado es limitado, su desarrollo y el análisis realizado permiten cumplir con el objetivo de proporcionar evidencia empírica inicial que sustenta las conclusiones del trabajo y orienta las líneas futuras de investigación

5. CONCLUSIONES

El presente trabajo partió de una hipótesis que planteaba que la inteligencia artificial puede contribuir a la detección temprana de estafas telefónicas basadas en ingeniería social, mediante el análisis automatizado de señales acústicas y textuales. A partir de esta premisa, se inició un proceso de investigación que permitió dar con herramientas, marcos metodológicos y enfoques de modelado adecuados para abordar el problema.

Sobre esta base, se propuso una arquitectura conceptual orientada a trabajar casi en tiempo real bajo un enfoque edge-only. El diseño se combinó con la implementación de un producto mínimo viable (MVP) cuyo objetivo fue validar experimentalmente el núcleo técnico de clasificación. El MVP no reprodujo la integración completa del sistema en un entorno operativo, sino que evaluó, en condiciones controladas, si los componentes centrales eran capaces de discriminar entre comunicaciones sospechosas y legítimas.

Los resultados obtenidos permiten afirmar que la hipótesis inicial se encuentra correctamente orientada: es técnicamente viable aplicar modelos de aprendizaje automático para identificar patrones asociados a estafas telefónicas. Si bien el experimento fue acotado, evidenció que tanto el análisis acústico como el análisis

textual aportan información relevante, y que su combinación fortalece la capacidad de clasificación.

En relación con el tamaño del conjunto de datos utilizado (100 muestras), se reconoce que este número resulta suficiente para una validación exploratoria del enfoque, pero no para afirmar robustez estadística ni generalización amplia. El objetivo del MVP no fue realizar una prueba de rendimiento comercial ni un benchmark (punto de referencia para comparativas) a gran escala, sino demostrar factibilidad técnica bajo condiciones controladas. La ampliación del dataset, ya sea mediante generación sintética o mediante recolección real con consentimiento explícito, constituye una condición necesaria para fortalecer futuras etapas del proyecto.

Asimismo, la arquitectura propuesta no debe interpretarse como un producto final completamente desarrollado, sino como un marco técnico que podría ser implementado respetando principios de ingeniería de software, modularidad y escalabilidad. Funcionalidades adicionales como validación por palabra clave, políticas de transferencia condicional o gestión avanzada de interacción, forman parte de posibles extensiones pero no constituyen el núcleo validado por esta tesis.

En resumen, el trabajo demuestra que la integración de inteligencia artificial en mecanismos de protección frente a estafas telefónicas es viable desde el punto de vista técnico, aunque su implementación real requiere consideraciones adicionales de ingeniería, regulación y experiencia de usuario que excedan el alcance del presente estudio.

6. FUTURAS LÍNEAS DE INVESTIGACIÓN

A partir de los resultados obtenidos y de las limitaciones identificadas, se abren diversas líneas de investigación que permitirían profundizar y ampliar el alcance del modelo propuesto.

En primer lugar, resulta necesario escalar el volumen y diversidad del conjunto de datos, incorporando mayor variabilidad de locutores, escenarios acústicos y modalidades de fraude. El análisis de las métricas resultantes permitiría evaluar el impacto del incremento de muestras en la estabilidad y generalización del modelo.

En segundo lugar, se plantea la exploración de arquitecturas que integren de manera más sofisticada señales acústicas, semánticas y contextuales. La incorporación de modelos de lenguaje de mayor complejidad podría mejorar la detección de patrones discursivos característicos de la ingeniería social.

Desde el punto de vista de despliegue, futuras investigaciones podrían analizar el comportamiento del sistema en dispositivos reales, evaluando consumo energético, latencia, compatibilidad y estrategias de actualización del modelo bajo un esquema edge-only.

En tercer lugar, resulta pertinente abrir el debate sobre los riesgos asociados a la adopción de soluciones basadas en inteligencia artificial en este dominio. La dependencia tecnológica, la existencia de márgenes de error inherentes a los modelos y

la percepción de confianza por parte de los usuarios constituyen variables críticas que deben ser estudiadas desde perspectivas técnicas, sociales y éticas.

Finalmente, la evolución del marco regulatorio y de la infraestructura de telecomunicaciones podría habilitar escenarios de mayor integración en el futuro. Cambios en políticas de acceso a audio, en capacidades de operadores o en legislación sobre tratamiento de datos en tiempo real podrían ampliar las posibilidades de implementación directa sobre llamadas nativas. En tal caso, el modelo conceptual desarrollado en esta tesis podría adaptarse a un entorno de mayor alcance operativo.

7. BIBLIOGRAFÍA

7.1 Referencias Bibliográficas

Ponce Gallegos, J. C., Torres Soto, A., Quezada Aguilera, F. S., Silva Sprock, A., Martínez Flor, E. U., Casali, A., ... & Pedreño, O. (2014). Inteligencia artificial (páginas 16 a 32). Iniciativa Latinoamericana de Libros de Texto Abiertos (LATIn). Disponible en https://rephip.unr.edu.ar/server/api/core/bitstreams/bb5e5b0c-01b6-482c-a3a4-a469f994c92b/content

Russell, S., & Norvig, P. (2004). Inteligencia artificial: Un enfoque moderno ((2ª ed., páginas 2 a 19). Pearson Educación.

Mitnick, K., & Simon, W. (2002). The art of deception: Controlling the human element of security (Capitulo 1). John Wiley & Sons.

7.2 Documentos Electrónicos

Abu Hweidi, R. F., & Eleyan, D. (2022). Social engineering attack concepts, frameworks, and awareness: A systematic literature review. International Journal of Computing and Digital Systems. https://www.researchgate.net/publication/372052065_Social_Engineering_Attack_concepts_frameworks_and_Awareness_A_Systematic_Literature_Review

Ahmad, H. A., & Rashid, T. A. (2024). Planning the development of text-to-speech synthesis models and datasets with dynamic deep learning. Journal of King Saud University - Computer and Information Sciences, 36(7), 102131. https://www.sciencedirect.com/science/article/pii/S1319157824002209

Alharbi, S., Alrazgan, M., Alrashed, A., Alnomasi, T., Almojel, R., Alharbi, R., ... & Almojil, M. (2021). Automatic speech recognition: Systematic literature review. Ieee Access, 9, 131858-131876. https://ieeexplore.ieee.org/document/9536732

Alpha Cephei. (2024). Vosk Speech Recognition Toolkit. https://alphacephei.com/vosk/

Amazon Web Services. (2024). Amazon Transcribe Pricing. https://aws.amazon.com/transcribe/pricing/

Amazon Web Services. (2024). Amazon Polly Pricing. https://aws.amazon.com/polly/pricing/

Amazon Web Services. (2024). Amazon S3 Pricing. https://aws.amazon.com/s3/pricing/

Ámbito. (2024, 14 de febrero). La DGI alertó por correos electrónicos que buscan estafar vía phishing a los contribuyentes. https://www.ambito.com/uruguay/la-dgi-alerto-correos-electronicos-que-buscan-estafar-via-phishing-los-contribuyentes-n5945083

Android Developers. (2024). Android Keystore System. https://developer.android.com/training/articles/keystore

Android Developers. (2024). Notifications overview. https://developer.android.com/develop/ui/views/notifications

Android Developers. (2024). TextToSpeech API.https://developer.android.com/reference/android/speech/tts/TextToSpeech

Anusuya, M. A., & Katti, S. K. (2009). Speech recognition by machine: A review. International Journal of Computer Science and Information Security, 6(3), 181–205. https://arxiv.org/abs/1001.2267

Apple. (2024). CallKit framework reference. Apple Developer. https://developer.apple.com/documentation/callkit

Asprea, G. (2025, abril 5). Los argentinos no sueltan el teléfono: más de tres meses del año se van mirando el celular. TN. https://tn.com.ar/sociedad/2025/04/05/los-argentinos-no-sueltan-el-telefono-mas-de-tres-meses-del-ano-se-van-mirando-el-celular/

Awah Buo, S. (2020). The emerging threats of deepfake attacks and countermeasures. arXiv Preprint. https://arxiv.org/abs/2012.07989

Bañuelos Capistrán, J. (2020). Deepfake: La imagen en tiempos de la posverdad. Revista Panamericana de Comunicación, 1(2), 51–61. https://www.redalyc.org/pdf/6649/664970407007.pdf

Bello, O., Sharma, S., Kar, A., & Gupta, M. (2025). Emerging trends in phishing: A look at smishing, vishing, and quishing. Computers & Security, 142, 103578. https://www.ijter.org/article/212507308755/emerging-trends-in-phishing-a-look-at-smishing-vishing-and-quishing

Benavides, E., Fuertes, W., & Sánchez, S. (2020). Caracterización de los ataques de phishing y técnicas para mitigarlos: Una revisión sistemática de la literatura. Ciencia y Tecnología, 13(1), 97–104.

https://www.researchgate.net/publication/343311739

Brijith, A. (2023). Unmasking scam calls: Analysing and detecting scammers using AI. Asia University. https://www.researchgate.net/publication/375003901

Busropan, C. (2020). Analysis of components in the manifest file of spam call blocking applications on Android.https://repository.tudelft.nl/file/File_3c6e3207-8157-4c96-a12a-c94394944436

Cherukuri, B. R. (2024). Edge computing vs. cloud computing: A comparative analysis for real-time ai applications. Int. J. Multidiscip. Res, 6, 29316. https://www.researchgate.net/profile/Bangar-Raju-Cherukuri/publication/386426266_Edge_Computing_vs_Cloud_Computing_A_Comparative_Analysis_for_Real-Time_AI_Applications/links/67513145876bd177783c16c0/Edge-Computing-vs-Cloud-Computing-A-Comparative-Analysis-for-Real-Time-AI-Applications.pdf

Choi, R. Y., Coyner, A. S., Kalpathy-Cramer, J., Chiang, M. F., & Campbell, J. P. (2020). Introduction to machine learning, neural networks, and deep learning. Translational Vision Science & Technology, 9(2), 14. https://iovs.arvojournals.org/article.aspx?articleid=2762344

Cialdini, R. B. (2009). Influence: Science and practice (5th ed.). Pearson. https://www.researchgate.net/publication/229067982

Coqui AI. (2024). Coqui STT: Speech-to-Text Toolkit.https://github.com/coqui-ai/STT

Datareportal. (2025). Global digital overview 2025. https://datareportal.com/global-digital-overview

Dida, H. A., Chakravarthy, D. S. K., & Rabbi, F. (2023). ChatGPT and big data: Enhancing text-to-speech conversion. Mesopotamian Journal of Big Data, 2023, 31–35. https://mesopotamian.press/journals/index.php/bigdata/article/view/47

Dutoit, T. (1997). High-quality text-to-speech synthesis: An overview. Journal of Electrical and Electronics Engineering Australia, 17(1), 25–36. https://www.academia.edu/download/3436726/Dutoit.pdf

Escobar Ruiz, D. (2025, septiembre 3). En qué consiste el fraude del "sí" al contestar una llamada telefónica y cómo evitar caer en la trampa. Infobae. https://www.infobae.com/tecno/2025/09/03/en-que-consiste-el-fraude-del-si-al-contestar-una-llamada-telefonica-y-como-evitar-caer-en-la-trampa/

eSpeak NG Project. (2024). eSpeak NG text-to-speech. https://github.com/espeak-ng/espeak-ng

Federal Trade Commission. (2025, 10 de marzo). Las principales estafas del 2024.
https://consumidor.ftc.gov/alertas-para-consumidores/2025/03/las-principales-estafas-

de-2024

Fenwick, M., Jurcys, P., Liaudanskas, A., Earp, B. D., Porsdam Mann, S., & Kozuka, S. (2024). Voice cloning in an age of generative AI: Mapping the limits of the law & principles for a new social contract with technology. SSRN. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4850866

Firebase. (2024). Firebase Cloud Messaging documentation. https://firebase.google.com/docs/cloud-messaging

Flores Cáceres, F. A. (2024). Clasificación de datos textuales provenientes de un streaming aplicando el método de representación de texto TF-IDF en una Regresión Logística. https://repositorio.lamolina.edu.pe/items/723321a1-f572-4088-9e21-7f2f128d5c73

FS-ISAC. (2024). Deepfakes in the financial sector: Understanding the threats, managing the risks. Artificial Intelligence Risk Working Group. https://www.fsisac.com/hubfs/Knowledge/AI/DeepfakesInTheFinancialSector-UnderstandingTheThreatsManagingTheRisks.pdf

García Sandoval, A. (2024). Aplicación del aprendizaje supervisado para reconocer patrones de datos de ciberamenazas y aprendizaje no supervisado para analizar y agrupar conjuntos de datos.

https://www.researchgate.net/publication/380293092

Genelza, G. G. (2024). A systematic literature review on AI voice cloning generator: A game-changer or a threat? Journal of Emerging Technologies, 4(2), 54–61. https://www.researchgate.net/publication/385394275_A_Systematic_Literature_Review_on_AI_Voice_Cloning_Generator_A_Game-changer_or_a_Threat

Gerganov, G. (2024). *whisper.cpp: Port of OpenAI Whisper model in C/C++*.https://github.com/ggerganov/whisper.cpp

Goel, D., Ahmad, H., Jain, A. K., & Goel, N. K. (2024). Machine learning driven smishing detection framework for mobile security. arXiv preprint arXiv:2412.09641.https://arxiv.org/abs/2412.09641

Google. (2018). Privacy changes in Android 9 (Pie). Android Developers. https://developer.android.com/about/versions/pie/android-9.0-changes-all#privacy-all

Google Play Console Help. (2024). Permissions and APIs that access sensitive information (SMS and Call Log permission policy). https://support.google.com/googleplay/android-developer/answer/10208820

Google. (2024). TensorFlow Lite documentation. https://www.tensorflow.org/lite

Google Security Blog. (2025, 18 de marzo). New AI-powered scam detection features. https://security.googleblog.com/2025/03/new-ai-powered-scam-detection-features.html

Hua, H., Li, Y., Wang, T., Dong, N., Li, W., & Cao, J. (2023). Edge computing with artificial intelligence: A machine learning perspective. ACM Computing Surveys, 55(9), 1-35.https://dl.acm.org/doi/abs/10.1145/3555802

Huang, J. J., & Leanos, J. J. A. (2018). Aclnet: efficient end-to-end audio classification cnn. arXiv preprint arXiv:1811.06669. https://arxiv.org/abs/1811.06669

Hutiri, W., Papakyriakopoulos, O., & Xiang, A. (2024). Not my voice! A taxonomy of ethical and safety harms of speech generators. In Proceedings of the 2024 ACM Conference on Fairness, Accountability, and Transparency (pp. 359–376). https://doi.org/10.1145/3630106.3658911

Infobae. (2025, mayo 18). Tras la denuncia del PRO, la Justicia pidió a la red social X que elimine las publicaciones con el video hecho con IA de Mauricio Macri. Infobae. https://www.infobae.com/politica/2025/05/18/que-dice-la-denuncia-presentada-por-el-pro-tras-la-difusion-de-un-video-hecho-con-ia/

Ittichaichareon, C., Suksri, S., & Yingthawornsuk, T. (2012, July). Speech recognition using MFCC. In International conference on computer graphics,

simulation and modeling (Vol. 9, p. 2012). https://www.academia.edu/download/85360099/9_20712576.pdf

Kaspersky. (2024, abril 3). Kaspersky reports nearly 900 million phishing attempts in 2024 as cyber threats increase. https://www.kaspersky.com/about/press-releases/kaspersky-reports-nearly-900-million-phishing-attempts-in-2024-as-cyber-threats-increase

Kassis, A., & Hengartner, U. (2021). Practical attacks on voice spoofing countermeasures. arXiv. https://arxiv.org/abs/2107.14642

Khan, D. (2025, enero-febrero). The evolution of AI in fraud detection: Technical frameworks and cross-sector applications. International Journal of Scientific Research in Computer Science, Engineering and Information Technology, 11(1), 964–973. https://ijsrcseit.com/index.php/home/article/view/CSEIT25111298/CSEIT25111298

Linphone. (2024). *Liblinphone SDK documentation*.https://gitlab.linphone.org/BC/public/liblinphone

López Grande, C. E., & Guadrón, R. S. (2015). Ingeniería social: El ataque silencioso. Revista Tecnológica, 8. ITCA-FEPADE. https://www.itca.edu.sv/wp-content/themes/elaniin-itca/docs/RevistaTec_N8_Digital.pdf

Maseno, E. M. (2017). Vishing attack detection model for mobile users (Master's thesis, KCA University). http://41.89.49.13:8080/xmlui/handle/123456789/1276

Meta Platforms, Inc. (2024). *ExecuTorch: On-device inference framework*. https://pytorch.org/mobile/home/

Moogk, D. R. (2012). Minimum viable product and the importance of experimentation in technology startups. Technology Innovation Management Review, 2(3). https://timreview.ca/sites/default/files/article_PDF/RancicMoogk_TIMReview_March2012.pdf

Naeem, M., et al. (2023). An unsupervised machine learning algorithms comprehensive review. ResearchGate. https://www.researchgate.net/publication/368983958

National Institute of Standards and Technology. (2001). Recommendation for Block Cipher Modes of Operation (SP 800-38A). https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38a.pdf

Nilsen, V. (2018). An introduction to machine learning. https://bafflerbach.github.io/test_software_carpentry/files/Module%201%20Lecture%20Notes.pdf

Noriega, V. (s.f.). Deepfakes: generación de imágenes y videos sintéticos

mediante autoencoders. https://victornoriega.github.io/deepfakes/

OpenAI. (2025). GPT-4o mini Model Pricing. https://platform.openai.com/pricing

Papakyriakopoulos, O., Choi, A. S. G., Thong, W., Zhao, D., Andrews, J., Bourke, R., ... & Koenecke, A. (2023, June). Augmented datasheets for speech datasets and ethical decision-making. In Proceedings of the 2023 ACM Conference on Fairness, Accountability, and Transparency (pp. 881-904).https://dl.acm.org/doi/abs/10.1145/3593013.3594049

Pietri, M., Mamei, M., & Colajanni, M. (2025). Telecom spam and scams in the 5G and artificial intelligence era: analyzing economic implications, technical challenges and global regulatory efforts. International Journal of Information Security, 24(3), 139. https://link.springer.com/article/10.1007/s10207-025-01062-8

Prabhavalkar, R., Sainath, T. N., & Wu, Y. (2023). End-to-end speech recognition: A survey. arXiv preprint arXiv:2303.03329. https://arxiv.org/abs/2303.03329

Raghunathan, T. E. (2021). Synthetic data. Annual review of statistics and its application, 8(1), 129-140. https://www.annualreviews.org/content/journals/10.1146/annurev-statistics-040720-031848

Reggiswarashari, F., & Sihwi, S. W. (2022). Speech emotion recognition using 2D-convolutional neural network. International Journal of Electrical and Computer Engineering, 12(6), 6594. https://www.academia.edu/download/92952678/87_1570742518_28439_EMr_28may22_31aug21_N.pdf

Rhasspy Project. (2024). *Piper: A fast, local neural text to speech system*. https://github.com/rhasspy/piper

Sangit, K., Prusty, R., & Sharma, V. (2025). The role of deep learning in credit card fraud. Cureus Journals. https://assets.cureusjournals.com/artifacts/upload/review_article/pdf/4298/20250911-80668-hzr0r6.pdf

Spina, M. L. (2013). Los grupos vulnerables y las tecnologías de la información y comunicaciones. 42 JAIIO – SID 2013. https://sedici.unlp.edu.ar/handle/10915/94148

Singh, P. P., & Rani, P. (2014). An approach to extract feature using MFCC. IOSR Journal of Engineering, 4(8), 21-25. https://www.academia.edu/download/34658262/D04812125.pdf

Singh, N., Khan, R. A., & Shree, R. (2012). MFCC and prosodic feature extraction techniques: a comparative study. International Journal of Computer Applications, 54(1), 9-13.

https://www.academia.edu/download/73945011/b22a5b9deaf2ab92d77ac137d1de4da00860.pdf

TikTok. (s. f.). Filtro de actores. TikTok. https://www.tiktok.com/discover/filtro-de-actores

Westerlund, M. (2019). The emergence of deepfake technology: A review. Technology Innovation Management Review, 9(11), 40–53. https://www.researchgate.net/publication/337644519_The_Emergence_of_Deepfake_Technology_A_Review

Yeboah-Boateng, E. O., & Amanor, P. M. (2014). Phishing, SMiShing & Vishing: An assessment of threats against mobile devices. Journal of Emerging Trends in Computing and Information Sciences, 5(4), 297–307. https://vbn.aau.dk/en/publications/phishing-smishing-amp-vishing-an-assessment-of-threats-against-mo/

Zscaler ThreatLabz. (2024, marzo 28). Phishing attacks rise 58% in a year with help of AI: ThreatLabz 2024 phishing report. Zscaler Blog. https://www.zscaler.es/blogs/security-research/phishing-attacks-rise-58-year-ai-threatlabz-2024-phishing-report

Zimba, A., Phiri, K. O., Kashale, C., & Phiri, M. N. (2024). A machine learning and natural language processing-based smishing detection model for mobile money transactions. International Journal on Information Technologies & Security,

16(3).https://ijits-bg.com/sites/default/files/archive/2024%28vol.16%29/No3/contents/2024-N3-07.pdf

Zucconi, A. (2018, marzo 14). Understanding the technology behind deepfakes. https://www.alanzucconi.com/2018/03/14/understanding-the-technology-behind-deepfakes/

World Wide Web Consortium. (2023). *WebRTC 1.0: Real-Time Communication Between Browsers*.https://www.w3.org/TR/webrtc/

7.3 Marco Normativo Argentino

Constitución de la Nación Argentina. (1994). Constitución de la Nación Argentina.

Ley N° 25.520. (2001). Ley de Inteligencia Nacional. Boletín Oficial de la República Argentina.

Ley N° 27.078. (2014). Ley Argentina Digital. Boletín Oficial de la República Argentina.

Código Procesal Penal de la Nación Argentina. (2015).

8. Anexo

8.1 Procedimiento Completo para la Reproducción del MVP ProxIA

8.1.1 Introducción

El presente anexo describe el procedimiento técnico necesario para reproducir el MVP desarrollado en esta investigación.

Se incluyen:

- Requisitos técnicos
- Herramientas utilizadas
- Configuración inicial del entorno
- Preparación del dataset
- Ejecución del modelo acústico
- Ejecución del modelo textual
- Fusión multimodal
- Generación de resultados

El objetivo es garantizar la reproducibilidad experimental del modelo propuesto.

8.1.2 Requisitos Previos

Requisitos Técnicos

- Cuenta activa de Google *
- Acceso a Google Drive
- Acceso a Google Colab
- Navegador web actualizado
- Conexión a Internet estable
- Audios de prueba en formato digital

No se requiere instalación local de software adicional, ya que todo el entorno se ejecuta en la nube.

*Para crear una cuenta dirigirse a https://www.google.com/intl/es/account/about/

(Ver Imagen 1)

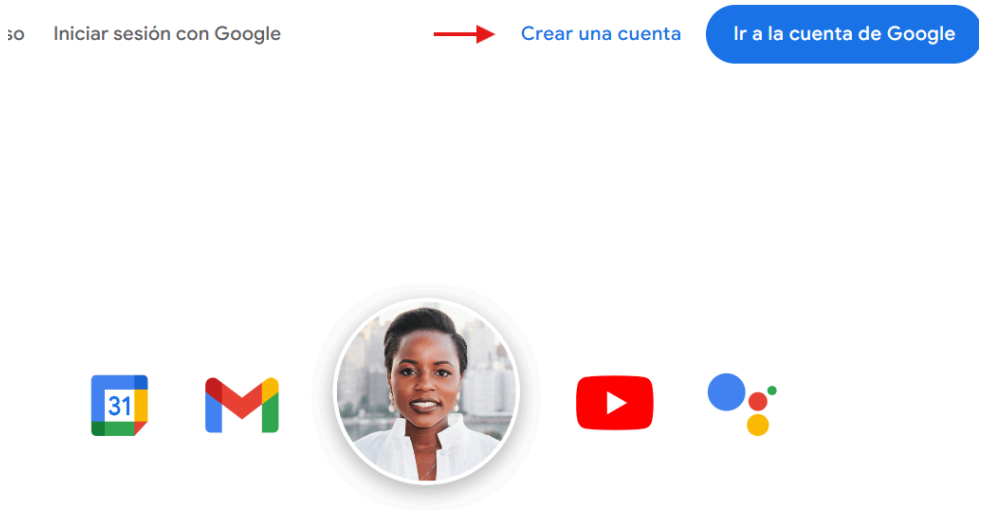


Imagen 1 . Crear cuenta Google

8.1.3. Herramientas Utilizadas

Las herramientas empleadas en el MVP fueron:

- Google Drive (almacenamiento del dataset y archivos)
- Google Colab (entorno de ejecución Python)
- Python 3.1
- TensorFlow / Keras
- Librosa
- Scikit-learn
- Whisper (modelo ASR)
- FFmpeg (conversión de audio)

Estas herramientas permiten construir un entorno reproducible sin dependencias locales.

8.1.4 Creación del Entorno de Trabajo

8.1.4.1 Acceso a Google Drive

1. Ingresar a https://drive.google.com (Ver imagen 2)
2. Iniciar sesión con cuenta Google. (Ver imagen 3)

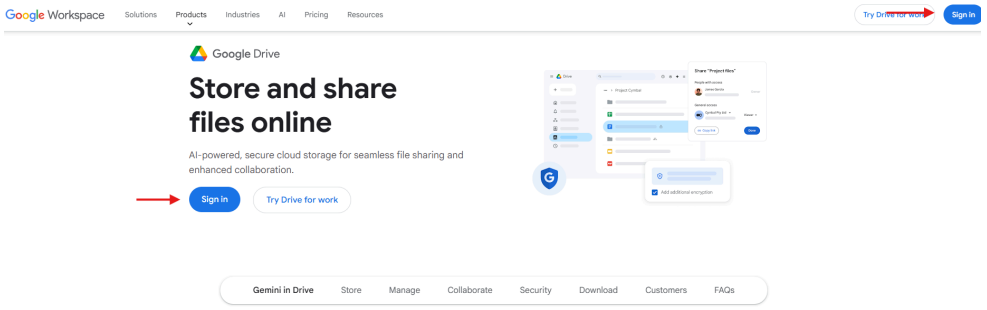


Imagen 2. Ingreso Google Drive

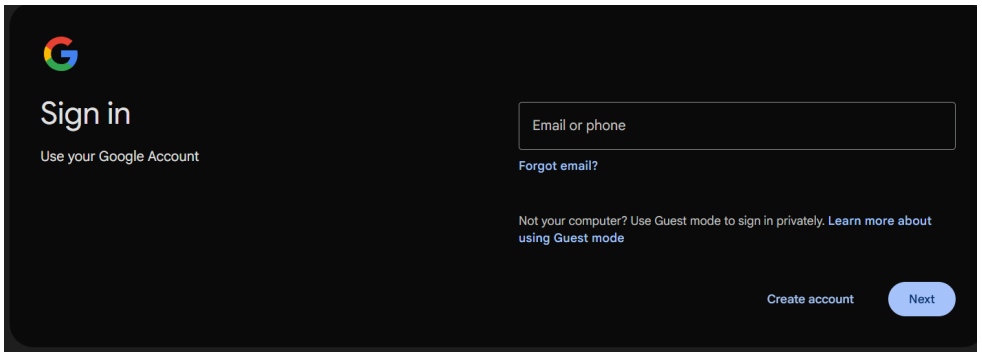


Imagen 3. Inicio Sesión Google

8.1.4.2 Creación del Proyecto

Dentro de “Mi unidad”, crear la carpeta principal: MVP (Ver imagen 4 y 5)

Dentro de ella, crear la siguiente estructura(Ver imagen 6)

/MVP/

- Notebooks/
- outputs/
- AudioDataset/
  - legitimo_sinconvertir/
  - sospechoso_sinconvertir/

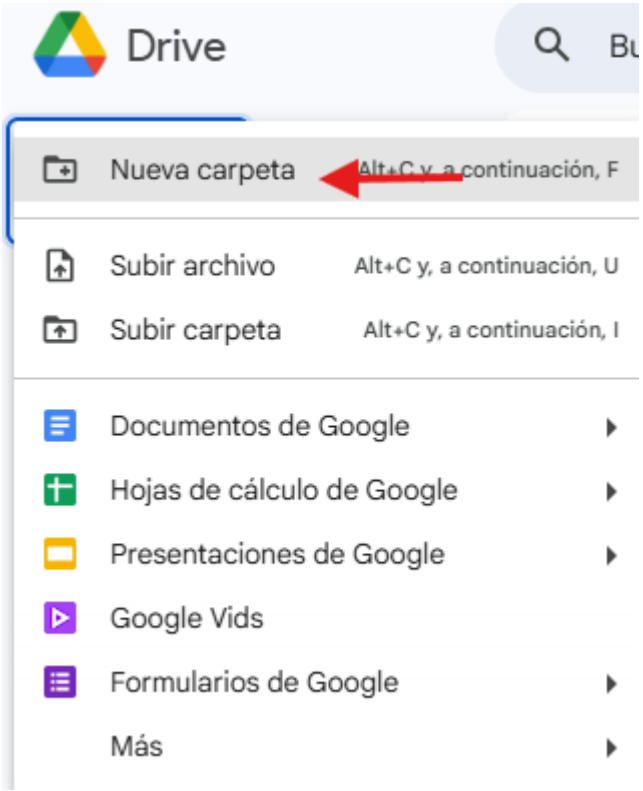


Imagen 4 . Crear Carpeta

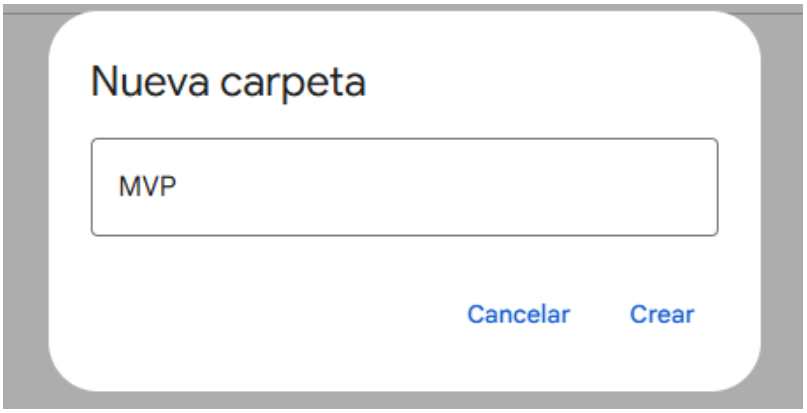


Imagen 5. Carpeta MVP

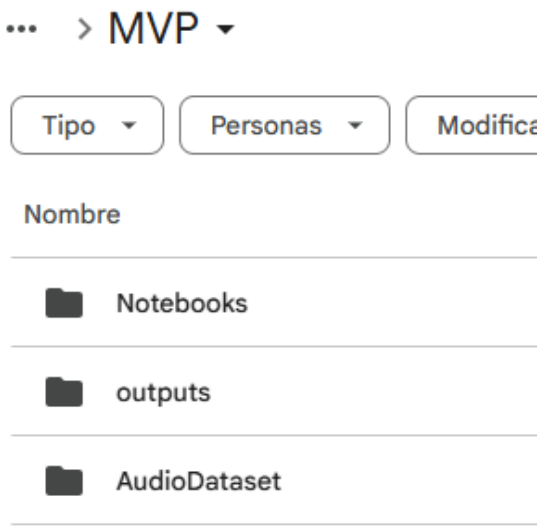


Imagen 6. Árbol Carpeta MVP

8.1.5 Preparación del Dataset

8.1.5.1 Grabación de Audios

Para reproducir el experimento se requieren:

- Audios simulando conversaciones legítimas

- Audios simulando intentos de estafa

Recomendaciones:

- Grabación en entorno silencioso
- Duración aproximada entre 3 y 6 segundos
- Claridad en la voz
- Guardar en formato estándar (opus, ogg, m4a, mp4 o wav)

8.1.5.2 Carga de Audios

Copiar los audios en sus respectivas carpetas (ver imagen 7):

- /MVP/AudioDataset/legitimo_sinconvertir/
- /MVP/AudioDataset/sospechoso_sinconvertir/



Imagen 7 Audios cargados en carpeta

8.1.6 Configuración en Google Colab

8.1.6.1 Acceso a Colab

- Ingresar a https://colab.research.google.com (ver imagen 8)
- Iniciar sesión (ver imagen 3)
- Crear nuevo notebook (ver imagen 9)

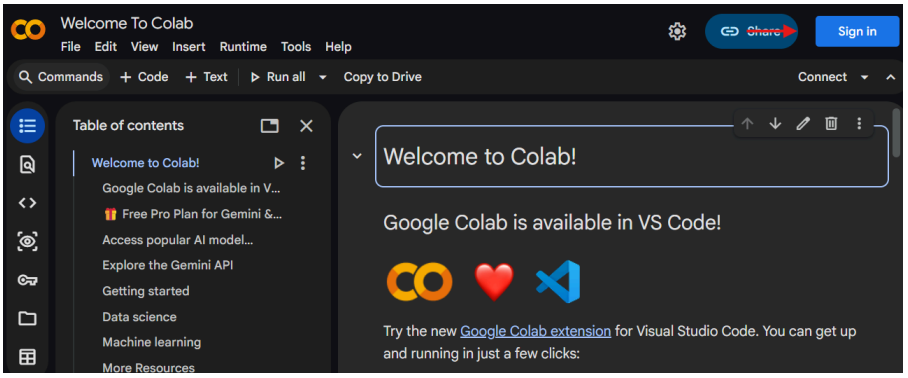


Imagen 8 Pantalla Inicio Colab

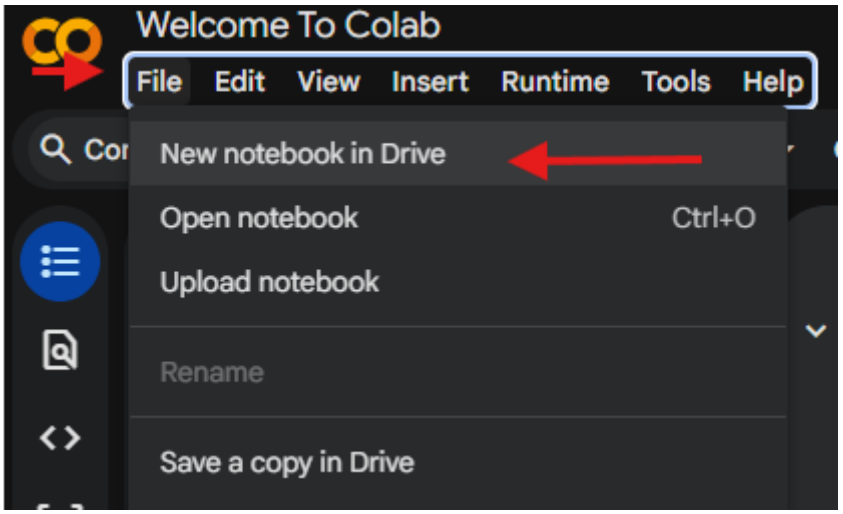


Imagen 9 Crear notebook en Colab

Guardar el archivo dentro de:

/MVP/Notebooks/

8.1.6.2 Montaje de Drive

Para permitir que el entorno de ejecución acceda a los archivos almacenados en Drive, es necesario montar la unidad en el sistema de archivos virtual de Colab.

Ejecutar:

```
from google.colab import drive
drive.mount('/content/drive')
```

Este procedimiento:

1. Solicita autorización al usuario.(Ver imagen 10 y 11)
2. Crea el punto de montaje /content/drive.
3. Permite acceder a los archivos mediante la ruta:

/content/drive/MyDrive/

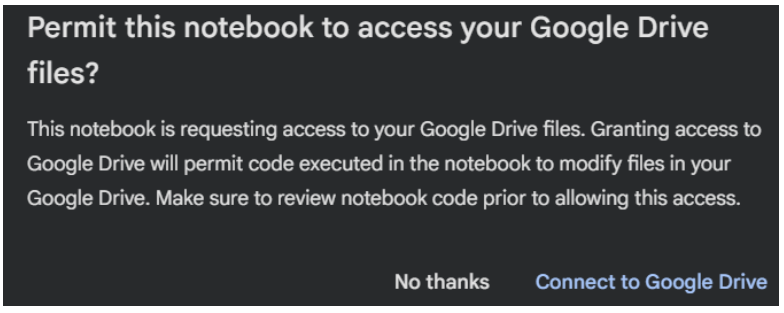


Imagen 10 Permisos montaje drive

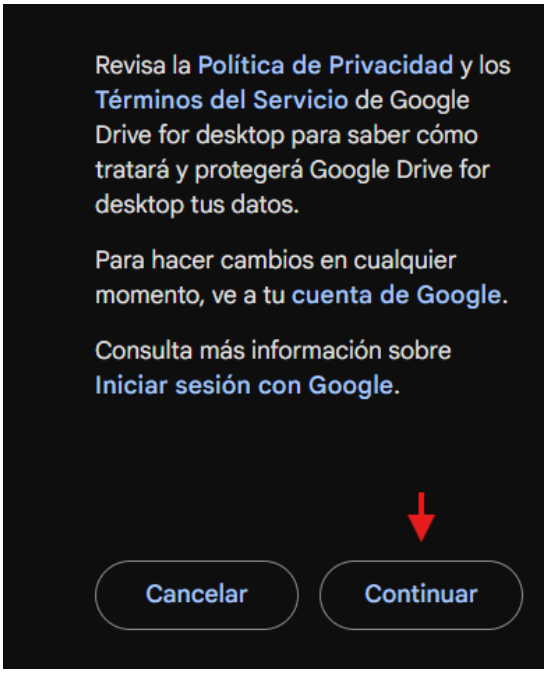


Imagen 11 Confirmar permisos montaje drive

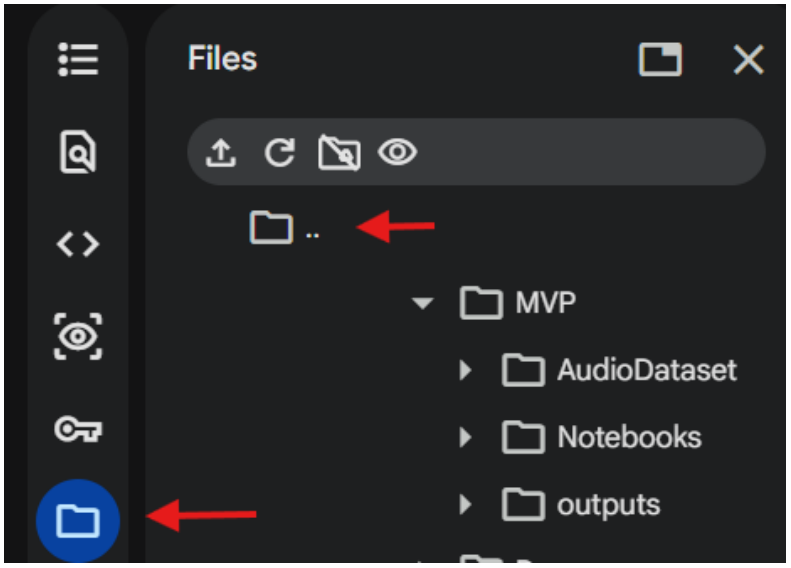


Imagen 12 Montaje finalizado

8.1.7 Ejecución del Modelo Acústico

Notebook: Notebook_1.ipynb (ver imagen 13)

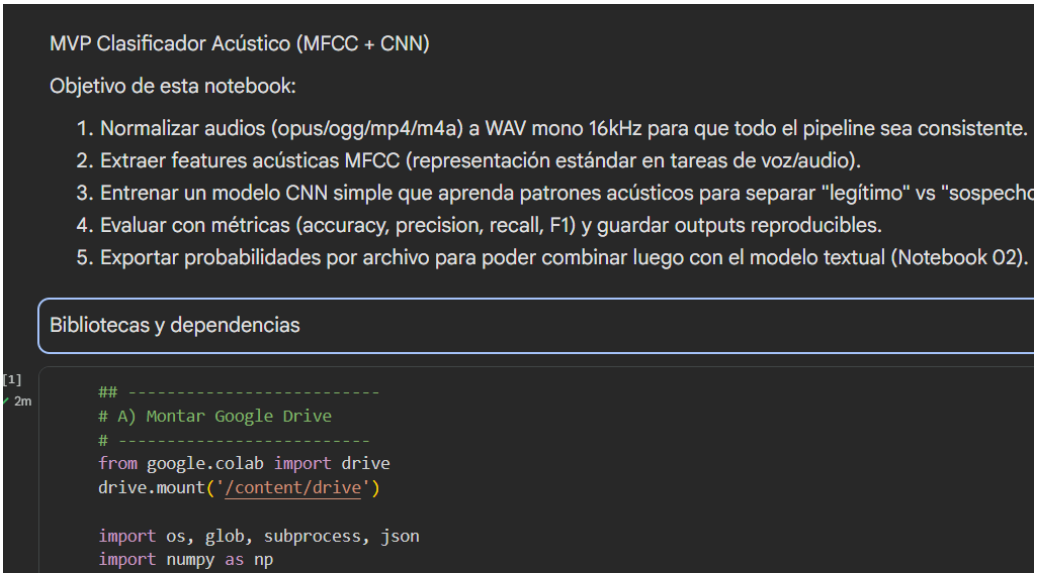


Imagen 13 Notebook_1.ipynb

8.1.7.1 Conversión de Audio

El notebook identifica automáticamente archivos en:

- legitimo_sinconvertir
- sospechoso_sinconvertir

Y los convierte mediante FFmpeg a (ver imagen 14):

- WAV
- Mono
- 16 kHz

Se generan las carpetas (ver imagen 15):

- legitimo_wav/
- sospechoso_wav/



Imagen 14. Conversión ejecutada.



Imagen 15. Carpetas WAV generadas.

8.1.7.2 Extracción de Características (MFCC)

Cada archivo WAV:

- Está normalizado.
- Está ajustado a 4 segundos (padding o truncado).
- Se transforma en una matriz de 40 coeficientes MFCC.

Se construyen tensores (ver imagen 16):

- X: (N, 40, T, 1)
- y: (N,)

Carga del Dataset y armado de tensores

```
# ========================
# 5) CARGAR DATASET Y ARMAR TENSORES
# ========================
# - Además de X e y, guardamos "paths" para saber qué predicción co
# - Esto me permite despues combinar audio + texto en la Notebook 0

def list_wavs(folder: str):
    return sorted(glob.glob(os.path.join(folder, "*.wav")))

pos_files = list_wavs(POS_DIR)
neg_files = list_wavs(NEG_DIR)

print("Sospechosos:", len(pos_files))
print("Legítimos:", len(neg_files))

# Balance: si hubiera mínimas diferencias, truncamos al mínimo comú
n = min(len(pos_files), len(neg_files))
pos_files = pos_files[:n]
neg_files = neg_files[:n]
print("Usando balanceado:", n, "por clase (total:", 2*n, ")")
```

Imagen 16. Dimensiones del dataset generado.

8.1.7.3 División del Dataset

El dataset se divide manteniendo proporciones por clase. en:

- 70% entrenamiento
- 15% validación
- 15% test

División Entrenamiento / Validación / Prueba

```
# ========================
# 6) SPLIT TRAIN / VAL / TEST
# ========================
X_trainval, X_test, y_trainval, y_test, paths_trainval, paths_test = train_test_split(
    X, y, paths, test_size=0.15, random_state=RANDOM_SEED, stratify=y
)

X_train, X_val, y_train, y_val, paths_train, paths_val = train_test_split(
    X_trainval, y_trainval, paths_trainval,
    test_size=0.1765,  # ~0.15 del total
    random_state=RANDOM_SEED, stratify=y_trainval
)

print("Train:", X_train.shape, "Val:", X_val.shape, "Test:", X_test.shape)
```

```
Train: (69, 40, 126, 1) Val: (16, 40, 126, 1) Test: (15, 40, 126, 1)
```

Imagen 17. División del dataset.

8.1.7.4 Arquitectura del Modelo CNN

El modelo implementado contiene (ver imagen 18):

- 3 bloques convolucionales
- Batch Normalization
- ReLU
- MaxPooling
- Dropout
- Capa densa
- Salida sigmoide binaria

Layer (type)	Output Shape	Param #
conv2d_39 (Conv2D)	(None, 40, 126, 32)	320
batch_normalization_39 (BatchNormalization)	(None, 40, 126, 32)	128
activation_39 (Activation)	(None, 40, 126, 32)	0
max_pooling2d_39 (MaxPooling2D)	(None, 20, 63, 32)	0
dropout_52 (Dropout)	(None, 20, 63, 32)	0
conv2d_40 (Conv2D)	(None, 20, 63, 64)	18,496
batch_normalization_40 (BatchNormalization)	(None, 20, 63, 64)	256
activation_40 (Activation)	(None, 20, 63, 64)	0
max_pooling2d_40 (MaxPooling2D)	(None, 10, 31, 64)	0
dropout_53 (Dropout)	(None, 10, 31, 64)	0
conv2d_41 (Conv2D)	(None, 10, 31, 128)	73,856
batch_normalization_41 (BatchNormalization)	(None, 10, 31, 128)	512
activation_41 (Activation)	(None, 10, 31, 128)	0
max_pooling2d_41 (MaxPooling2D)	(None, 5, 15, 128)	0
dropout_54 (Dropout)	(None, 5, 15, 128)	0
flatten_13 (Flatten)	(None, 9600)	0
dense_26 (Dense)	(None, 128)	1,228,928
dropout_55 (Dropout)	(None, 128)	0

Imagen 18. Resumen del modelo (model.summary).

8.1.7.5 Entrenamiento

Se emplean:

- Adam
- Binary Crossentropy
- EarlyStopping
- ReduceLROnPlateau
- Class weighting para priorizar clase sospechosa

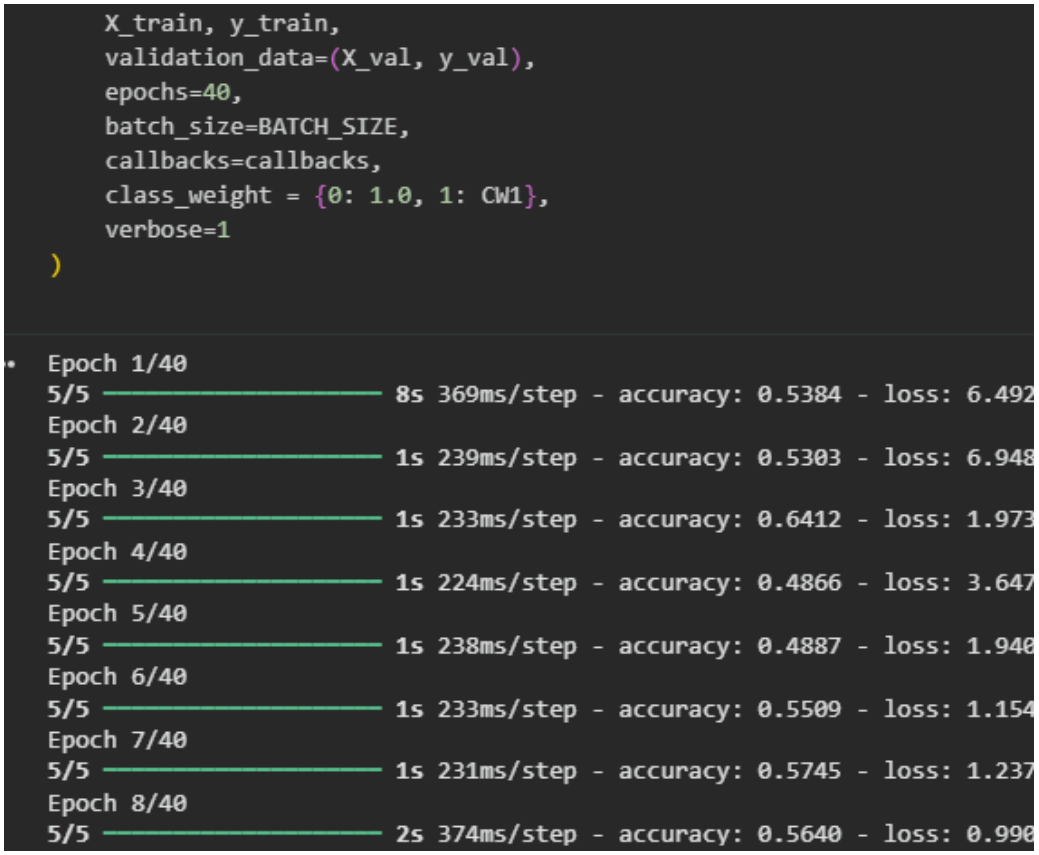


Imagen 19. Proceso de entrenamiento por épocas.

8.1.7.6 Evaluación y Resultados

Se calculan:

- Accuracy
- Precision
- Recall
- F1
- Matriz de confusión

Se generan (ver imagen 21):

- confusion_matrix.png

- curve_loss.png
- curve_accuracy.png
- model_mvp.keras
- preds_acoustic_all.csv

```
#Para analizar cual fue la mejor ejecución
import pandas as pd, os
LOG_PATH = os.path.join(BASE_OUT, "outputs", "runs_log.csv")
df = pd.read_csv(LOG_PATH)
df.sort_values(["recall_sospechoso", "f1_sospechoso"], ascending=False).head(10)
```

	run_id	run_name	out_dir	sr	dur_sec	n_mfcc	hop_length
0	20251220_200100	A_baseline	/content/drive/MyDrive/UP/18-TESIS/Tesis/MVP/o...	16000	4.0	40	512
3	20251220_201029	B_lr_low	/content/drive/MyDrive/UP/18-TESIS/Tesis/MVP/o...	16000	4.0	40	512
4	20251220_201347	B_lr_low	/content/drive/MyDrive/UP/18-TESIS/Tesis/MVP/o...	16000	3.0	40	512
5	20251220_204449	Modelo_A	/content/drive/MyDrive/UP/18-TESIS/Tesis/MVP/o...	16000	4.0	40	512
6	20251221_014723	Modelo_A	/content/drive/MyDrive/UP/18-TESIS/Tesis/MVP/o...	16000	4.0	40	512
1	20251220_200624	B_lr_low	/content/drive/MyDrive/UP/18-TESIS/Tesis/MVP/o...	16000	4.0	40	512
2	20251220_200907	B_lr_low	/content/drive/MyDrive/UP/18-TESIS/Tesis/MVP/o...	16000	4.0	40	512

7 rows × 23 columns

Imagen 20. Reporte de clasificación.

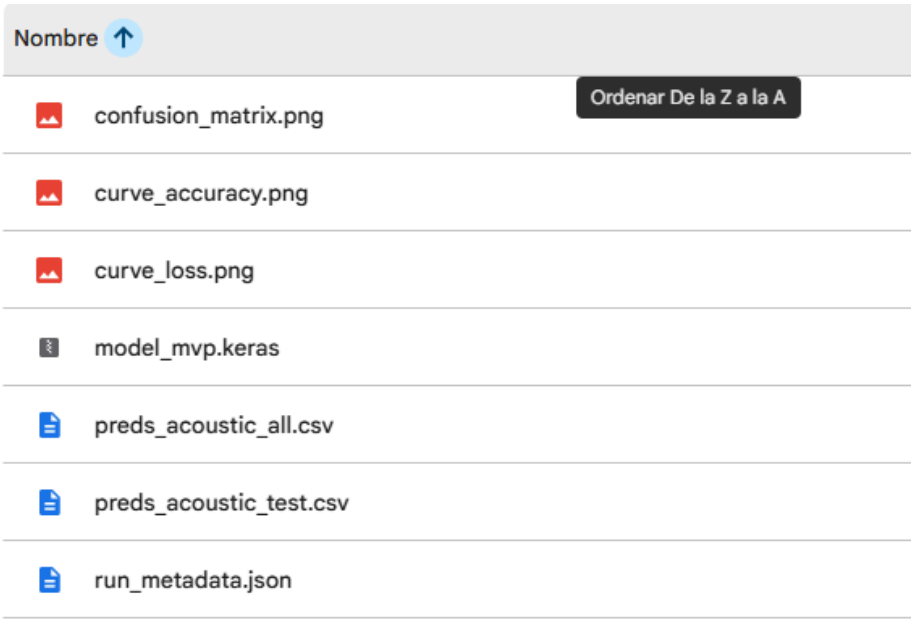


Imagen 21. Carpeta run generada en outputs.

8.1.8 Ejecución del Modelo Textual

Notebook: Notebook_02.ipynb (ver imagen 22)

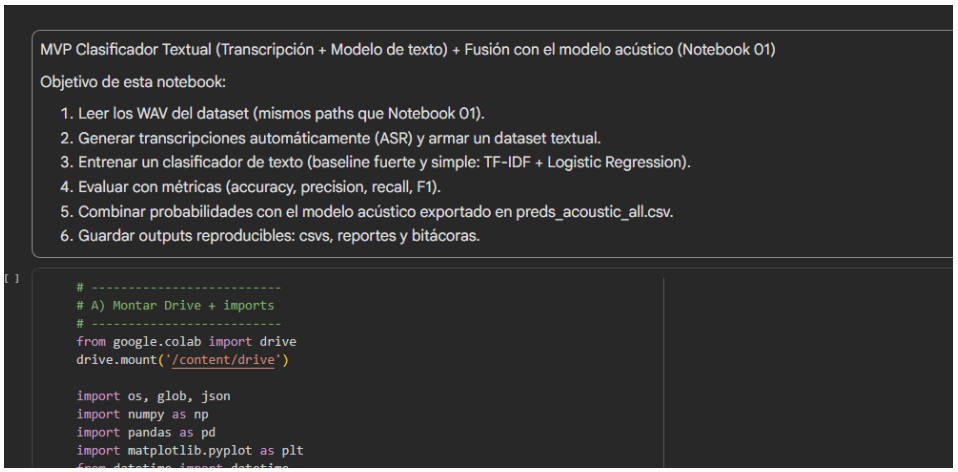


Imagen 22. Notebook_02.ipynb.

8.1.8.1 Instalación y Configuración de Whisper

El notebook instala y carga el modelo Whisper (small) (ver imagen 23).

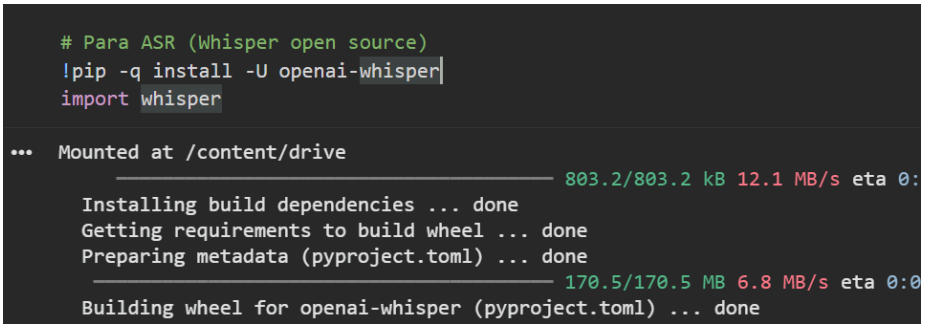


Imagen 23. Instalación de Whisper.

8.1.8.2 Transcripción Automática

Cada archivo WAV es transcripto y almacenado en (ver imagen 24):

- transcriptions_cache.csv

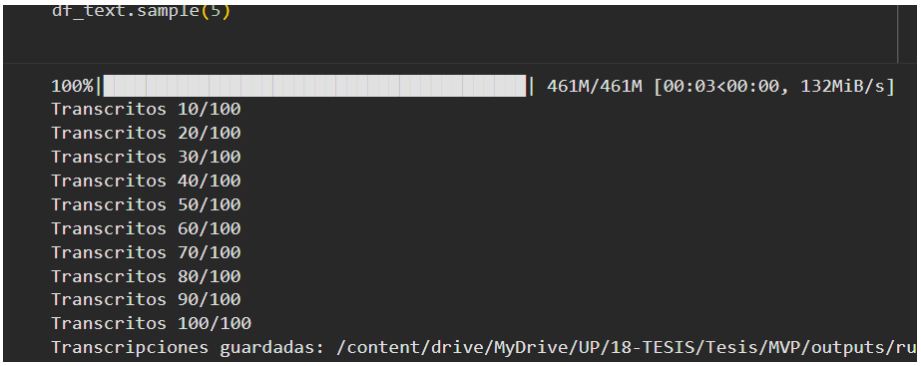


Imagen 24. Proceso de transcripción.

8.1.8.3 Entrenamiento del Modelo Textual

Se utiliza:

- TF-IDF (5000 features)
- N-grams (1,2)
- Logistic Regression con class_weight balanced

```
# ============================================================
# F) Modelo textual (TF-IDF + Logistic Regression)
# ============================================================
# Por qué este baseline:
# - Para datasets chicos suele ser muy competitivo.
# - Entrena rápido.
# - Da probabilidades para fusionar con acústico.
# - No requiere GPU.

RUN_NAME = "T_tfidf_logreg"
MAX_FEATS = 5000
NGRAM_RANGE = (1,2)

vectorizer = TfidfVectorizer(
    lowercase=True,
    max_features=MAX_FEATS,
    ngram_range=NGRAM_RANGE
)

Xtr = vectorizer.fit_transform(X_train)
Xva = vectorizer.transform(X_val)
Xte = vectorizer.transform(X_test)

clf = LogisticRegression(max_iter=200, class_weight="balanced")
```

Imagen 25. Métricas del modelo textual.

8.1.9 Fusión Multimodal

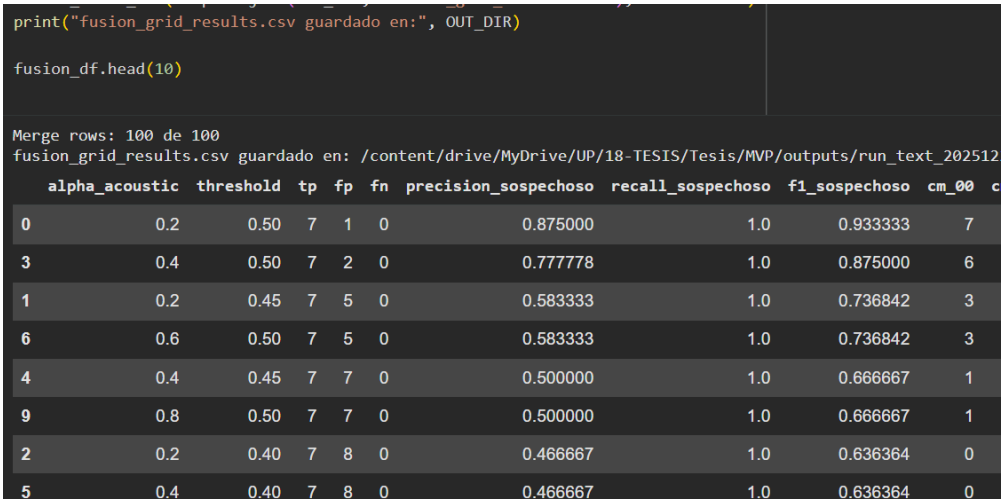
Se combinan probabilidades mediante:

- P_final = α * P_acoustic + (1 - α) * P_text

Se evalúan múltiples combinaciones de α y threshold.

Se genera:

- fusion_grid_results.csv
- runs_log_fusion.csv



```
print("fusion_grid_results.csv guardado en:", OUT_DIR)

fusion_df.head(10)
```

```
Merge rows: 100 de 100
fusion_grid_results.csv guardado en: /content/drive/MyDrive/UP/18-TESIS/Tesis/MVP/outputs/run_text_202512
```

	alpha_acoustic	threshold	tp	fp	fn	precision_sospechoso	recall_sospechoso	f1_sospechoso	cm_00	c
0	0.2	0.50	7	1	0	0.875000	1.0	0.933333	7	
3	0.4	0.50	7	2	0	0.777778	1.0	0.875000	6	
1	0.2	0.45	7	5	0	0.583333	1.0	0.736842	3	
6	0.6	0.50	7	5	0	0.583333	1.0	0.736842	3	
4	0.4	0.45	7	7	0	0.500000	1.0	0.666667	1	
9	0.8	0.50	7	7	0	0.500000	1.0	0.666667	1	
2	0.2	0.40	7	8	0	0.466667	1.0	0.636364	0	
5	0.4	0.40	7	8	0	0.466667	1.0	0.636364	0	

Imagen 26 Resultado óptimo de fusión.

8.1.10 Reproducción con Nuevos Audios

Para repetir el experimento:

1. Agregar nuevos audios en carpetas sinconvertir.
2. Ejecutar Notebook 01.
3. Ejecutar Notebook 02.
4. Verificar nuevos resultados en /outputs/.

Cada ejecución genera una carpeta con timestamp, evitando sobrescritura.

8.1.11 Ubicación Final de Resultados

Todos los resultados se almacenan en (ver imagen 27):

/MVP/outputs/

Incluyendo:

- Runs acústicos
- Runs textuales
- Logs acumulados
- Resultados de fusión

Esto garantiza trazabilidad experimental.

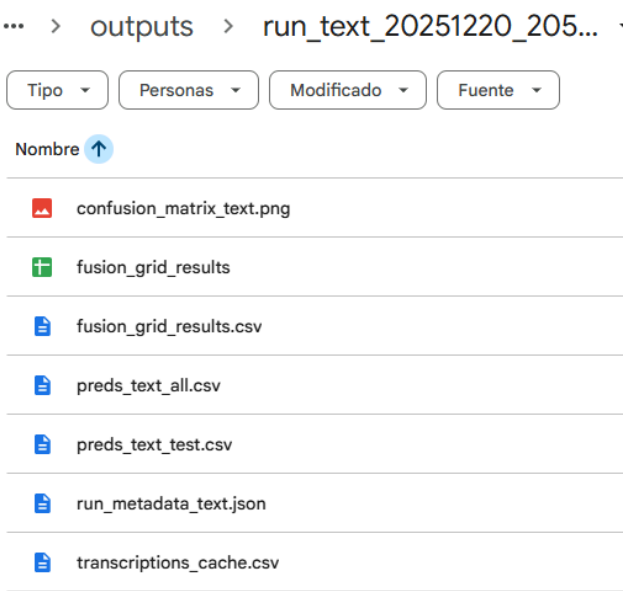


Imagen 27 Carpeta Outputs.

8.1.12 Consideraciones de Reproducibilidad

- Semilla fija (SEED = 42)
- Split estratificado
- Versionado por timestamp
- Registro automático de ejecuciones
- Separación modular acústico / texto

8.1.13 Disponibilidad del Código Fuente

Con el objetivo de garantizar la transparencia, trazabilidad y reproducibilidad del experimento, el código completo del MVP se encuentra disponible públicamente en el siguiente repositorio:

https://github.com/cdboronat/mvp-audio-scam-classifier

El repositorio contiene:

- Notebook 01_acustico_train_eval.ipynb
- Notebook 02_texto_transcribe_train_fusion.ipynb
- Estructura de carpetas sugerida
- Instrucciones de uso

La publicación del código permite:

- Validación independiente de los resultados.
- Reentrenamiento con nuevos datasets.
- Ajuste de hiperparámetros.

- Extensión del modelo hacia nuevos escenarios.
- Evaluación comparativa con otros enfoques.

8.1.16 Código

A partir de la siguiente línea se comparte el código en python utilizado.

Notebook 01

```
# -*- coding: utf-8 -*-
"""01_acustico_train_eval.ipynb

MVP Clasificador Acústico (MFCC + CNN)

Objetivo de esta notebook:
 1) Normalizar audios (opus/ogg/mp4/m4a) a WAV mono 16kHz para que todo el pipeline sea consistente.
 2) Extraer features acústicas MFCC (representación estándar en tareas de voz/audio).
 3) Entrenar un modelo CNN simple que aprenda patrones acústicos para separar "legítimo" vs "sospechoso".
 4) Evaluar con métricas (accuracy, precision, recall, F1) y guardar outputs reproducibles.
 5) Exportar probabilidades por archivo para poder combinar luego con el modelo textual (Notebook 02).

Bibliotecas y dependencias
"""

## -------------------------
# A) Montar Google Drive
# -------------------------
from google.colab import drive
drive.mount('/content/drive')
```

```
import os, glob, subprocess, json
import numpy as np
import matplotlib.pyplot as plt

import librosa
import tensorflow as tf
from tensorflow.keras import layers, models

from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, confusion_matrix

from datetime import datetime

import pandas as pd



import random
SEED = 42
random.seed(SEED)
np.random.seed(SEED)
tf.random.set_seed(SEED)

"""Normalizar Audios

"""

# ==========================
# 2) CONVERSIÓN A WAV (OPUS/OGG/MP4/M4A)
# ==========================
# Necesario porque:
# - WhatsApp u otros medios de grabación suelen producir OPUS/OGG/MP4/M4A.
# - Si entrenamos mezclando formatos distintos, el modelo puede aprender de las propiedades de cada uno tomándolo como un diferencial.
```

```
# - Convertir TODO a WAV mono 16kHz nos deja una entrada estándar.

BASE_DATA = "/content/drive/MyDrive/UP/18-TESIS/Tesis/MVP/AudioDataset" # Donde se encuentran mis audios (Separar en 2 carpetas los legítimos y sospechosos)
BASE_OUT = "/content/drive/MyDrive/UP/18-TESIS/Tesis/MVP/" # Donde dejo los resultados

#
IN_FOLDERS = {
    "sospechoso": os.path.join(BASE_DATA, "sospechoso_sinconvertir"),
    "legitimo": os.path.join(BASE_DATA, "legitimo_sinconvertir"),
}

OUT_FOLDERS = {
    "sospechoso": os.path.join(BASE_DATA, "sospechoso_wav"),
    "legitimo": os.path.join(BASE_DATA, "legitimo_wav"),
}

for _, p in OUT_FOLDERS.items():
    os.makedirs(p, exist_ok=True)

# Verificamos que ffmpeg exista (Colab normalmente ya lo trae)
subprocess.run(["ffmpeg", "-version"], stdout=subprocess.DEVNULL, stderr=subprocess.DEVNULL)

def convert_folder(in_dir, out_dir):
    # Aceptamos múltiples extensiones porque los audios pueden venir de distintas apps / exportaciones
    exts = ["*.opus", "*.ogg", "*.mp4", "*.m4a"]
    files = []
    for e in exts:
        files.extend(glob.glob(os.path.join(in_dir, e)))

    print(f"Encontrados {len(files)} archivos en {in_dir}")
    converted, skipped = 0, 0
```

```
    for f in files:
        base = os.path.splitext(os.path.basename(f))[0]
        out_path = os.path.join(out_dir, base + ".wav")

         # Si ya existe el wav, no lo volvemos a convertir: así evitamos pisar y ahorramos
tiempo
        if os.path.exists(out_path):
            skipped += 1
            continue

        # Convertimos a WAV mono 16kHz, descartando video si existiera (MP4)
        cmd = [
            "ffmpeg", "-y",
            "-i", f,
            "-ac", "1",       # mono: una sola señal de audio
            "-ar", "16000",   # 16kHz: estándar común para voz
            "-vn",            # ignorar video
            out_path
        ]

                              res    =    subprocess.run(cmd,    stdout=subprocess.DEVNULL,
stderr=subprocess.DEVNULL)
        if res.returncode == 0 and os.path.exists(out_path):
            converted += 1

    print(f"Convertidos: {converted} | Omitidos (ya existían): {skipped}")
    print(f"Salida: {out_dir}\n")

for label in ["sospechoso", "legitimo"]:
    convert_folder(IN_FOLDERS[label], OUT_FOLDERS[label])

"""Configuración de directorios"""
```

```
# ==========================
# 3) CONFIGURACIÓN DE RUTAS Y PARÁMETROS
# ==========================
# Por qué esta parte importa ?
# - Conviene fijar un setup y sostenerlo para que los resultados sean reproducibles.
# - Si ajustamos los parámetros todo el tiempo, después es difícil justificar qué versión es “la final”.

POS_DIR = os.path.join(BASE_DATA, "sospechoso_wav")
NEG_DIR = os.path.join(BASE_DATA, "legitimo_wav")
assert os.path.isdir(POS_DIR), f"No existe: {POS_DIR}"
assert os.path.isdir(NEG_DIR), f"No existe: {NEG_DIR}"

SR = 16000
DUR_SEC = 4.0              # ventana fija elegida para cubrir audios más largos; los más cortos se rellenan (padding)
N_SAMPLES = int(SR * DUR_SEC)

N_MFCC = 40              # cantidad típica de coeficientes MFCC para voz
HOP_LENGTH = 512
N_FFT = 2048
RANDOM_SEED = 42

print(f"Config: SR={SR}, DUR_SEC={DUR_SEC}, N_SAMPLES={N_SAMPLES}, N_MFCC={N_MFCC}")

"""Carga de Audios, Corte, MFCC"""

# ==========================
# 4) FUNCIONES: CARGA, PAD/TRIM, MFCC
# ==========================
# 1) cargar audio y normalizar volumen (evita que el modelo se guíe por amplitud)
# 2) pad/trim a duración fija (entrada consistente)
# 3) MFCC = representación compacta del espectro para tareas de voz
```

```
def load_audio(path: str, sr: int = SR) -> np.ndarray:
    y, _ = librosa.load(path, sr=sr, mono=True)
    maxv = np.max(np.abs(y)) if len(y) else 0.0
    if maxv > 0:
        y = y / maxv
    return y

def pad_or_trim(y: np.ndarray, n_samples: int = N_SAMPLES) -> np.ndarray:
    if len(y) < n_samples:
        return np.pad(y, (0, n_samples - len(y)), mode="constant")
    return y[:n_samples]

def extract_mfcc(y: np.ndarray) -> np.ndarray:
    mfcc = librosa.feature.mfcc(
        y=y, sr=SR, n_mfcc=N_MFCC, n_fft=N_FFT, hop_length=HOP_LENGTH
    )
    # normalización: ayuda a que el modelo vea “patrones” y no escalas arbitrarias
    mfcc = (mfcc - np.mean(mfcc)) / (np.std(mfcc) + 1e-8)
    return mfcc.astype(np.float32)

"""Carga del Dataset y armado de tensores"""

# ==========================
# 5) CARGAR DATASET Y ARMAR TENSORES
# ==========================
# - Además de X e y, guardamos "paths" para saber qué predicción corresponde a qué archivo.
# - Esto me permite despues combinar audio + texto en la Notebook 02 una correspondencia entre ellos

def list_wavs(folder: str):
    return sorted(glob.glob(os.path.join(folder, "*.wav")))

pos_files = list_wavs(POS_DIR)
```

```
neg_files = list_wavs(NEG_DIR)

print("Sospechosos:", len(pos_files))
print("Legítimos:", len(neg_files))

# Balance: si hubiera mínimas diferencias, truncamos al mínimo común para mantener dataset balanceado
n = min(len(pos_files), len(neg_files))
pos_files = pos_files[:n]
neg_files = neg_files[:n]
print("Usando balanceado:", n, "por clase (total:", 2*n, ")")

X, y, paths = [], [], []

def build_samples(file_list, label: int):
    for fp in file_list:
        audio = load_audio(fp)
        audio = pad_or_trim(audio)
        mfcc = extract_mfcc(audio)
        X.append(mfcc)
        y.append(label)
        paths.append(fp)  # para exportar predicciones por archivo

build_samples(pos_files, 1)  # 1 = sospechoso
build_samples(neg_files, 0)  # 0 = legitimo

X = np.stack(X, axis=0)               # (N, n_mfcc, time_frames)
y = np.array(y, dtype=np.int32)       # (N,)
X = X[..., np.newaxis]                # (N, n_mfcc, time_frames, 1) -> Conv2D

print("X shape:", X.shape, "y shape:", y.shape, "paths:", len(paths))
print("Time frames (T):", X.shape[2])

"""División Entrenamiento / Validación / Prueba"""
```

```
# ==========================
# 6) SPLIT TRAIN / VAL / TEST
# ==========================
X_trainval, X_test, y_trainval, y_test, paths_trainval, paths_test = train_test_split(
    X, y, paths, test_size=0.15, random_state=RANDOM_SEED, stratify=y
)

X_train, X_val, y_train, y_val, paths_train, paths_val = train_test_split(
    X_trainval, y_trainval, paths_trainval,
    test_size=0.1765,  # ~0.15 del total
    random_state=RANDOM_SEED, stratify=y_trainval
)

print("Train:", X_train.shape, "Val:", X_val.shape, "Test:", X_test.shape)

# ==========================
# Estos son algunos de los parametros que se pueden ajustar para realizar diferentes ejecuciones y comparar resultados
# ==========================
# Modificación de 4 valores y pruebas de resultados

RUN_NAME = "Modelo_A"   # ej: Modelo_A, Modelo_B, etc
LR = 1e-3               # learning rate
BATCH_SIZE = 16
CW1 = 2.0               # peso para clase sospechoso (1)

# Si queremos tambien probar un DUR_SEC distinto por ejecución, cambiar DUR_SEC en la parte del codigo correspondiente
# (y volver a ejecutar desde la parte de armado de dataset).

"""Creación del modelo"""

# ==========================
# 7) MODELO CNN
```

```
# ==========================
# - “Patrones locales” en MFCC (conv2d) que pueden correlacionar con estilos de habla, ritmos, etc.
# Nota:
# - Esto NO captura semántica. Por eso luego se complementa con transcripción (Notebook 02).

input_shape = X_train.shape[1:]

model = models.Sequential([
    layers.Input(shape=input_shape),

    layers.Conv2D(32, (3,3), padding="same"),
    layers.BatchNormalization(),
    layers.Activation("relu"),
    layers.MaxPooling2D((2,2)),
    layers.Dropout(0.2),

    layers.Conv2D(64, (3,3), padding="same"),
    layers.BatchNormalization(),
    layers.Activation("relu"),
    layers.MaxPooling2D((2,2)),
    layers.Dropout(0.3),

    layers.Conv2D(128, (3,3), padding="same"),
    layers.BatchNormalization(),
    layers.Activation("relu"),
    layers.MaxPooling2D((2,2)),
    layers.Dropout(0.3),

    layers.Flatten(),
    layers.Dense(128, activation="relu"),
    layers.Dropout(0.4),

    layers.Dense(1, activation="sigmoid")
```

```
])


model.compile(
    optimizer=tf.keras.optimizers.Adam(learning_rate=LR),
    loss="binary_crossentropy",
    metrics=[
        "accuracy",
        tf.keras.metrics.Precision(name="precision"),
        tf.keras.metrics.Recall(name="recall")
    ]
)

model.summary()

"""Entrenamiento"""

# ==========================
# 8) ENTRENAMIENTO
# ==========================
# Por qué class_weight:
# - En detección de estafas suele ser más grave “no detectar” (FN) que alertar de más (FP).
# - class_weight hace que equivocarse con sospechoso “cueste más” durante el entrenamiento.

callbacks = [
    tf.keras.callbacks.EarlyStopping(
        monitor="val_loss", patience=6, restore_best_weights=True
    ),
    tf.keras.callbacks.ReduceLROnPlateau(
        monitor="val_loss", factor=0.5, patience=3, min_lr=1e-6
    )
]
```

```
class_weight = {0: 1.0, 1: 2.0}

history = model.fit(
    X_train, y_train,
    validation_data=(X_val, y_val),
    epochs=40,
    batch_size=BATCH_SIZE,
    callbacks=callbacks,
    class_weight = {0: 1.0, 1: CW1},
    verbose=1
)

"""Salida  : Curvas"""

# ========================
# 9) CURVAS + OUTPUTS (carpeta por corrida)
# ========================
run_id = datetime.now().strftime("%Y%m%d_%H%M%S")
OUT_DIR = os.path.join(BASE_OUT, "outputs", f"run_{run_id}")
os.makedirs(OUT_DIR, exist_ok=True)

def plot_history(hist, key, filename):
    plt.figure()
    plt.plot(hist.history[key], label=f"train_{key}")
    plt.plot(hist.history[f"val_{key}"], label=f"val_{key}")
    plt.xlabel("epoch")
    plt.ylabel(key)
    plt.legend()
    plt.grid(True)
    plt.savefig(os.path.join(OUT_DIR, filename), dpi=200)

plot_history(history, "loss", "curve_loss.png")
plot_history(history, "accuracy", "curve_accuracy.png")
```

```
print("Guardado en:", OUT_DIR)

"""Evualuación resultados"""

# ==========================
# 10) EVALUACIÓN: REPORTE + F1 + MATRIZ
# ==========================
y_prob = model.predict(X_test).ravel()
y_pred = (y_prob >= 0.5).astype(int)

print(classification_report(y_test, y_pred, target_names=["legitimo","sospechoso"]))

cm = confusion_matrix(y_test, y_pred)
print("Confusion matrix:\n", cm)

# F1 manual para la clase sospechoso
tp = np.sum((y_test==1) & (y_pred==1))
fp = np.sum((y_test==0) & (y_pred==1))
fn = np.sum((y_test==1) & (y_pred==0))
precision = tp / (tp + fp + 1e-9)
recall = tp / (tp + fn + 1e-9)
f1 = 2 * precision * recall / (precision + recall + 1e-9)
print(f"Precision={precision:.4f} Recall={recall:.4f} F1={f1:.4f}")

# Guardar matriz como figura
plt.figure()
plt.imshow(cm)
plt.title("Matriz de confusión (acústico)")
plt.xticks([0,1], ["legitimo","sospechoso"])
plt.yticks([0,1], ["legitimo","sospechoso"])
for (i,j), v in np.ndenumerate(cm):
    plt.text(j, i, str(v), ha="center", va="center")
plt.xlabel("Predicción")
plt.ylabel("Real")
```

```
plt.grid(False)
plt.savefig(os.path.join(OUT_DIR, "confusion_matrix.png"), dpi=200)

"""Guardar Modelo y Metadata"""

# ==========================
# 11) GUARDAR MODELO + METADATA
# ==========================
MODEL_PATH = os.path.join(OUT_DIR, "model_mvp.keras")
model.save(MODEL_PATH)

meta = {
    "sr": SR,
    "dur_sec": DUR_SEC,
    "n_samples": N_SAMPLES,
    "n_mfcc": N_MFCC,
    "hop_length": HOP_LENGTH,
    "n_fft": N_FFT,
    "random_seed": RANDOM_SEED,
    "class_weight": class_weight,
    "files_per_class_used": n,
    "pos_dir": POS_DIR,
    "neg_dir": NEG_DIR
}

with open(os.path.join(OUT_DIR, "run_metadata.json"), "w") as f:
    json.dump(meta, f, indent=2)

print("Modelo guardado en:", MODEL_PATH)
print("Modelo y metadata guardados en:", OUT_DIR)



# 1) CSV del TEST (sirve para documentar resultados reportados)
test_df = pd.DataFrame({
    "path": paths_test,
```

```
    "y_true": y_test,
    "y_prob_acoustic": y_prob,
    "y_pred_acoustic": y_pred
})
test_df.to_csv(os.path.join(OUT_DIR, "preds_acoustic_test.csv"), index=False)

# 2) CSV de TODO el dataset (sirve para fusión con texto en Notebook 02)
y_prob_all = model.predict(X).ravel()
y_pred_all = (y_prob_all >= 0.5).astype(int)

all_df = pd.DataFrame({
    "path": paths,
    "label_true": y,
    "y_prob_acoustic": y_prob_all,
    "y_pred_acoustic": y_pred_all
})
all_df.to_csv(os.path.join(OUT_DIR, "preds_acoustic_all.csv"), index=False)

print("OK: preds_acoustic_test.csv y preds_acoustic_all.csv guardados en:", OUT_DIR)

# ==========================
# BITÁCORA DEL RUN (append a un CSV global)
# ==========================
log_row = {
    "run_id": run_id,
    "run_name": RUN_NAME,
    "out_dir": OUT_DIR,

    # dataset / features
    "sr": SR,
    "dur_sec": DUR_SEC,
    "n_mfcc": N_MFCC,
    "hop_length": HOP_LENGTH,
```

```
    "n_fft": N_FFT,
    "files_per_class_used": n,

    # training params
    "learning_rate": LR,
    "batch_size": BATCH_SIZE,
    "cw_sospechoso": CW1,
    "random_seed": RANDOM_SEED,

    # results (test)
    "tp": int(tp),
    "fp": int(fp),
    "fn": int(fn),
    "precision_sospechoso": float(precision),
    "recall_sospechoso": float(recall),
    "f1_sospechoso": float(f1),
    "cm_00": int(cm[0,0]),
    "cm_01": int(cm[0,1]),
    "cm_10": int(cm[1,0]),
    "cm_11": int(cm[1,1]),
}

LOG_PATH = os.path.join(BASE_OUT, "outputs", "runs_log.csv")
os.makedirs(os.path.dirname(LOG_PATH), exist_ok=True)

log_df = pd.DataFrame([log_row])

if os.path.exists(LOG_PATH):
    log_df.to_csv(LOG_PATH, mode="a", header=False, index=False)
else:
    log_df.to_csv(LOG_PATH, index=False)

print("Bitácora actualizada:", LOG_PATH)

"""Inferencia de Audio Nuevo"""
```

```
# ==========================
# 12) INFERENCIA SOBRE UN AUDIO NUEVO
# ==========================
def predict_file(wav_path: str):
    audio = load_audio(wav_path)
    audio = pad_or_trim(audio)
    mfcc = extract_mfcc(audio)
    x = mfcc[np.newaxis, ..., np.newaxis]
    prob = float(model.predict(x).ravel()[0])
    label = "sospechoso" if prob >= 0.5 else "legitimo"
    return label, prob

example_wav = pos_files[0]
label, prob = predict_file(example_wav)
print("Archivo:", example_wav)
print("Predicción:", label, "Prob:", prob)

#Para analizar cual fue la mejor ejecución
import pandas as pd, os
LOG_PATH = os.path.join(BASE_OUT, "outputs", "runs_log.csv")
df = pd.read_csv(LOG_PATH)
df.sort_values(["recall_sospechoso", "f1_sospechoso"], ascending=False).head(10)
```

Notebook 02

```
# -*- coding: utf-8 -*-
"""02_texto_transcribe_train_fusion.ipynb.ipynb
```

MVP Clasificador Textual (Transcripción + Modelo de texto) + Fusión con el modelo acústico (Notebook 01)

Objetivo de esta notebook:

```
1) Leer los WAV del dataset (mismos paths que Notebook 01).
2) Generar transcripciones automáticamente (ASR) y armar un dataset textual.
3) Entrenar un clasificador de texto (baseline fuerte y simple: TF-IDF + Logistic Regression).
4) Evaluar con métricas (accuracy, precision, recall, F1).
5) Combinar probabilidades con el modelo acústico exportado en preds_acoustic_all.csv.
6) Guardar outputs reproducibles: csvs, reportes y bitácoras.
"""

# -------------------------
# A) Montar Drive + imports
# -------------------------
from google.colab import drive
drive.mount('/content/drive')

import os, glob, json
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from datetime import datetime

from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, confusion_matrix

# Modelo textual (Este modole es simple, rápido y apto para datasets chicos)
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression

# Para ASR (Whisper open source)
!pip -q install -U openai-whisper
import whisper

# -------------------------
# B) Rutas
```

```
# -------------------------
# Importante , esta notebook solo puede ejecutarse posterior a haber trabajado con los audios y haber generados resultados con la Notebook 01
BASE_DATA = "/content/drive/MyDrive/UP/18-TESIS/Tesis/MVP/AudioDataset"
BASE_OUT  = "/content/drive/MyDrive/UP/18-TESIS/Tesis/MVP"

POS_DIR = os.path.join(BASE_DATA, "sospechoso_wav")
NEG_DIR = os.path.join(BASE_DATA, "legitimo_wav")

assert os.path.isdir(POS_DIR), f"No existe: {POS_DIR}"
assert os.path.isdir(NEG_DIR), f"No existe: {NEG_DIR}"



# Archivo de Notebook 01 (predicciones acústicas para fusión)  ejemplo : content/drive/MyDrive/UP/18-TESIS/Tesis/MVP/outputs/run_20251220_204449/preds_acoustic_all.csv
ACOUSTIC_PREDS_CSV = "/content/drive/MyDrive/UP/18-TESIS/Tesis/MVP/outputs/run_20251220_204449/preds_acoustic_all.csv"    # <-- completar con el path a preds_acoustic_all.csv del run elegido

# Output por run
run_id = datetime.now().strftime("%Y%m%d_%H%M%S")
OUT_DIR = os.path.join(BASE_OUT, "outputs", f"run_text_{run_id}")
os.makedirs(OUT_DIR, exist_ok=True)

print("OUT_DIR:", OUT_DIR)

# ============================================================
# C) Cargar lista de WAVs y labels
# ============================================================
def list_wavs(folder: str):
    return sorted(glob.glob(os.path.join(folder, "*.wav")))
```

```
pos_files = list_wavs(POS_DIR)
neg_files = list_wavs(NEG_DIR)

print("Sospechosos:", len(pos_files))
print("Legítimos:", len(neg_files))

n = min(len(pos_files), len(neg_files))
pos_files = pos_files[:n]
neg_files = neg_files[:n]
print("Usando balanceado:", n, "por clase (total:", 2*n, ")")

paths = pos_files + neg_files
y = np.array([1]*len(pos_files) + [0]*len(neg_files), dtype=np.int32)

print("Total samples:", len(paths))

# ============================================================
# D) Transcripción automática (ASR)
# ============================================================
#
# - Usaremos Whisper que es Open Source y  puede transcribir español bastante bien.
# - Guardamos un cache CSV para no transcribir todo cada vez.
# - Si el audio es corto (3-4s) va rápido.

WHISPER_MODEL = "small"  # opciones: tiny / base / small / medium
LANG = "es"

CACHE_PATH = os.path.join(OUT_DIR, "transcriptions_cache.csv")

model_asr = whisper.load_model(WHISPER_MODEL)

def transcribe_file(wav_path: str) -> str:
    # fp16=False por compatibilidad; si tenés GPU puede ir fp16=True
    result = model_asr.transcribe(wav_path, language=LANG, fp16=False)
    text = (result.get("text") or "").strip()
```

```
    return text

texts = []
for i, p in enumerate(paths, 1):
    t = transcribe_file(p)
    texts.append(t)
    if i % 10 == 0:
        print(f"Transcritos {i}/{len(paths)}")

df_text = pd.DataFrame({"path": paths, "label": y, "text": texts})
df_text.to_csv(CACHE_PATH, index=False)
print("Transcripciones guardadas:", CACHE_PATH)

# (opcional) ver ejemplos
df_text.sample(5)

# ============================================================
# E) Split train/val/test (igual filosofía que notebook 01)
# ============================================================
X_trainval, X_test, y_trainval, y_test, paths_trainval, paths_test = train_test_split(
    df_text["text"].values, df_text["label"].values, df_text["path"].values,
    test_size=0.15, random_state=42, stratify=df_text["label"].values
)

X_train, X_val, y_train, y_val = train_test_split(
    X_trainval, y_trainval,
    test_size=0.1765, random_state=42, stratify=y_trainval
)

print("Train:", len(X_train), "Val:", len(X_val), "Test:", len(X_test))

# ============================================================
# F) Modelo textual (TF-IDF + Logistic Regression)
# ============================================================
```

```
# Por qué este baseline:
# - Para datasets chicos suele ser muy competitivo.
# - Entrena rápido.
# - Da probabilidades para fusionar con acústico.
# - No requiere GPU.

RUN_NAME = "T_tfidf_logreg"
MAX_FEATS = 5000
NGRAM_RANGE = (1,2)

vectorizer = TfidfVectorizer(
    lowercase=True,
    max_features=MAX_FEATS,
    ngram_range=NGRAM_RANGE
)

Xtr = vectorizer.fit_transform(X_train)
Xva = vectorizer.transform(X_val)
Xte = vectorizer.transform(X_test)

clf = LogisticRegression(max_iter=200, class_weight="balanced")
clf.fit(Xtr, y_train)

# probas de clase 1 (sospechoso)
y_prob = clf.predict_proba(Xte)[:, 1]
y_pred = (y_prob >= 0.5).astype(int)

print(classification_report(y_test, y_pred, target_names=["legitimo","sospechoso"]))

cm = confusion_matrix(y_test, y_pred)
print("Confusion matrix:\n", cm)

# F1 manual para sospechoso
tp = np.sum((y_test==1) & (y_pred==1))
fp = np.sum((y_test==0) & (y_pred==1))
```

```
fn = np.sum((y_test==1) & (y_pred==0))
precision = tp / (tp + fp + 1e-9)
recall = tp / (tp + fn + 1e-9)
f1 = 2 * precision * recall / (precision + recall + 1e-9)
print(f"Precision={precision:.4f} Recall={recall:.4f} F1={f1:.4f}")

# Guardar matriz
plt.figure()
plt.imshow(cm)
plt.title("Matriz de confusión (texto)")
plt.xticks([0,1], ["legitimo","sospechoso"])
plt.yticks([0,1], ["legitimo","sospechoso"])
for (i,j), v in np.ndenumerate(cm):
    plt.text(j, i, str(v), ha="center", va="center")
plt.xlabel("Predicción")
plt.ylabel("Real")
plt.grid(False)
plt.savefig(os.path.join(OUT_DIR, "confusion_matrix_text.png"), dpi=200)

# Guardar preds test
test_df = pd.DataFrame({
    "path": paths_test,
    "y_true": y_test,
    "y_prob_text": y_prob,
    "y_pred_text": y_pred
})
test_df.to_csv(os.path.join(OUT_DIR, "preds_text_test.csv"), index=False)

# Guardar preds ALL
Xall = vectorizer.transform(df_text["text"].values)
y_prob_all = clf.predict_proba(Xall)[:, 1]
y_pred_all = (y_prob_all >= 0.5).astype(int)

all_df = pd.DataFrame({
```

```
    "path": df_text["path"].values,
    "label_true": df_text["label"].values,
    "text": df_text["text"].values,
    "y_prob_text": y_prob_all,
    "y_pred_text": y_pred_all
})
all_df.to_csv(os.path.join(OUT_DIR, "preds_text_all.csv"), index=False)

# Metadata textual
meta = {
    "run_id": run_id,
    "run_name": RUN_NAME,
    "asr_model": WHISPER_MODEL,
    "lang": LANG,
    "vectorizer": "TfidfVectorizer",
    "max_features": MAX_FEATS,
    "ngram_range": NGRAM_RANGE,
    "classifier": "LogisticRegression(class_weight=balanced)",
    "threshold": 0.5
}
with open(os.path.join(OUT_DIR, "run_metadata_text.json"), "w") as f:
    json.dump(meta, f, indent=2)

print("Guardado texto en:", OUT_DIR)

# ============================================================
# G) Bitácora textual (append)
# ============================================================
log_row = {
    "run_id": run_id,
    "run_name": RUN_NAME,
    "out_dir": OUT_DIR,
    "asr_model": WHISPER_MODEL,
    "max_features": MAX_FEATS,
    "ngram_range": str(NGRAM_RANGE),
```

```
    "threshold": 0.5,
    "tp": int(tp),
    "fp": int(fp),
    "fn": int(fn),
    "precision_sospechoso": float(precision),
    "recall_sospechoso": float(recall),
    "f1_sospechoso": float(f1),
    "cm_00": int(cm[0,0]),
    "cm_01": int(cm[0,1]),
    "cm_10": int(cm[1,0]),
    "cm_11": int(cm[1,1]),
}

LOG_PATH = os.path.join(BASE_OUT, "outputs", "runs_log_text.csv")
os.makedirs(os.path.dirname(LOG_PATH), exist_ok=True)
pd.DataFrame([log_row]).to_csv(LOG_PATH, mode="a", header=not
os.path.exists(LOG_PATH), index=False)
print("Bitácora texto actualizada:", LOG_PATH)

# ============================================================
# H) Fusión acústico + texto
# ============================================================
# La idea:
# - El acústico tiende a recall alto (no dejar pasar estafas) pero precision baja.
# - El texto debería bajar falsos positivos.
# - Fusionamos probas y elegimos un umbral final.
#
# score = alpha * p_acoustic + (1-alpha) * p_text

assert ACOUSTIC_PREDS_CSV and os.path.exists(ACOUSTIC_PREDS_CSV), \
    "Completar ACOUSTIC_PREDS_CSV con la ruta al preds_acoustic_all.csv del run
elegido (Notebook 01)."

df_a = pd.read_csv(ACOUSTIC_PREDS_CSV)
```

```
df_t = pd.read_csv(os.path.join(OUT_DIR, "preds_text_all.csv"))

# merge por path
df = df_t.merge(df_a[["path", "y_prob_acoustic"]], on="path", how="inner")

print("Merge rows:", len(df), "de", len(df_t))

# Evaluar fusión en el mismo split de test que el texto (paths_test)
test_paths_set = set(paths_test)
df_test = df[df["path"].isin(test_paths_set)].copy()

def eval_probs(y_true, probs, thr=0.5):
    y_pred = (probs >= thr).astype(int)
    cm = confusion_matrix(y_true, y_pred)
    tp = np.sum((y_true==1) & (y_pred==1))
    fp = np.sum((y_true==0) & (y_pred==1))
    fn = np.sum((y_true==1) & (y_pred==0))
    prec = tp / (tp + fp + 1e-9)
    rec  = tp / (tp + fn + 1e-9)
    f1   = 2 * prec * rec / (prec + rec + 1e-9)
    return {"tp":tp,"fp":fp,"fn":fn,"precision":prec,"recall":rec,"f1":f1,"cm":cm}

# Probamos algunos alpha y umbrales
alphas = [0.2, 0.4, 0.6, 0.8]
thresholds = [0.5, 0.45, 0.4]

fusion_rows = []

y_true = df_test["label_true"].values.astype(int)

for a in alphas:
    score = a*df_test["y_prob_acoustic"].values + (1-a)*df_test["y_prob_text"].values
    for thr in thresholds:
        m = eval_probs(y_true, score, thr=thr)
        fusion_rows.append({
```

```
            "alpha_acoustic": a,
            "threshold": thr,
            "tp": int(m["tp"]),
            "fp": int(m["fp"]),
            "fn": int(m["fn"]),
            "precision_sospechoso": float(m["precision"]),
            "recall_sospechoso": float(m["recall"]),
            "f1_sospechoso": float(m["f1"]),
            "cm_00": int(m["cm"][0,0]),
            "cm_01": int(m["cm"][0,1]),
            "cm_10": int(m["cm"][1,0]),
            "cm_11": int(m["cm"][1,1]),
        })

fusion_df =
pd.DataFrame(fusion_rows).sort_values(["recall_sospechoso","f1_sospechoso"],
ascending=False)
fusion_df.to_csv(os.path.join(OUT_DIR, "fusion_grid_results.csv"), index=False)
print("fusion_grid_results.csv guardado en:", OUT_DIR)

fusion_df.head(10)

# ============================================================
# I) Elegir mejor fusión según tu criterio A (recall primero)
# ============================================================
best = fusion_df.iloc[0].to_dict()
print("\n=== MEJOR FUSIÓN (recall primero, luego F1) ===")
print(best)

# Guardar bitácora fusión global (append)
best_row = {
    "run_id": run_id,
    "out_dir": OUT_DIR,
    "acoustic_preds_csv": ACOUSTIC_PREDS_CSV,
```

```
    **best
}

LOG_FUSION = os.path.join(BASE_OUT, "outputs", "runs_log_fusion.csv")
pd.DataFrame([best_row]).to_csv(LOG_FUSION, mode="a", header=not
os.path.exists(LOG_FUSION), index=False)
print("Bitácora fusión actualizada:", LOG_FUSION)
```

8.1.15 Cierre

El presente anexo documentó el proceso técnico que permite reconstruir el MVP ProxIA desde cero, asegurando replicabilidad, validación externa y extensibilidad futura.