Escuela
Politécnica
Superior

Detección de Mentiras mediante un clasificador automático

Grado en Ingeniería Robótica

Trabajo Fin de Grado

Autor:
Tamai Ramírez Gordillo
Tutor/es:
Francisco A. Pujol

Universitat d'Alacant
Universidad de Alicante

Junio 2022

Detección de Mentiras mediante un clasificador automático

Autor
Tamai Ramírez Gordillo

Tutor/es
Francisco A. Pujol
Departamento de Tecnología Informática y Computación

Grado en Ingeniería Robótica

Escuela Politécnica Superior

Universitat d'Alacant
Universidad de Alicante

ALICANTE, Junio 2022

Preámbulo

"La razón principal que ha llevado a realizar este proyecto es mi interés por el mundo de la investigación, centrado principalmente en el ámbito de la Inteligencia Artificial, así como brindar un nuevo enfoque en una de las áreas de la psicología, la detección de mentiras. El objetivo principal de este proyecto, por tanto, es crear un sistema que sea capaz de determinar si el discurso de una persona es veraz o no. "

Agradecimientos

Me gustaría agradecer a mi familia y amigos por todo el apoyo que me han brindado e interés para elaborar este proyecto, por confiar siempre en mí y estar en los mejores y peores momentos. A todos los participantes de la base de datos por prestarme su ayuda desinteresada y a mi tutor, Paco, por estar siempre en cada paso que daba desde el inicio y sobretodo, por nunca dejar de creer en mí.

"Si caminas solo llegarás más rápido,
si caminas acompañado llegarás más lejos".

Antiguo proverbio africano.

Índice general

Índice de figuras

Índice de tablas

1. Introducción.

Desde los inicios del ser humano se han ido desarrollando diversas capacidades para adaptarse al entorno o a las diferentes circunstancias de la vida. Conforme el habla fue desarrollándose junto a las habilidades sociales, también apareció la capacidad de ocultar la verdad. Los seres humanos aprendemos a mentir desde muy pequeños conforme nos relacionamos con el entorno y con el resto de seres humanos (Álava, 2016). Empleamos esta capacidad para multitud de fines diversos, para encajar en un grupo social, conseguir un objetivo, eludir un castigo, evitar hacer daño a un ser querido e incluso para preservar el control y la justicia en la política (Platón, 2009). Además, no solo se emplea la mentira contra otros seres humanos, también se emplean contra uno mismo, cuando por ejemplo, nos decimos que no somos lo suficientemente capaces para realizar una tarea y en realidad sí que lo somos.

La capacidad de detectar la mentira es una habilidad valiosa para cualquier investigador, ya sea policial o de otro tipo. No detectar una mentira durante una entrevista puede poner en riesgo toda la investigación y podría dar lugar a una demanda, hacer que una persona inocente pierda su trabajo o vaya a la cárcel, o incluso llevar a una empresa a la quiebra.

Cabe destacar que es muy compleja la detección fiable de una mentira, pues depende del tipo de persona, si es segura o no, si es una persona racional o emocional, etc. No obstante, hay varias claves que combinadas indican que lo más probable es que la persona esté mintiendo. Por ello, es necesario en muchas ocasiones el empleo de grabaciones en vídeos sobre el interrogatorio, de forma que, puedan analizarse todos los detalles y poder enfatizar en cada uno por separado, pudiendo así perfeccionar la detección de mentiras. Sin embargo, aún teniendo las grabaciones para analizar la posible mentira, se requiere de mucho tiempo de estudio para hacerlo completamente bien, debido a esto, desde que existen los métodos de machine learning, se plantea la búsqueda de herramientas que automaticen estos procedimientos, así poder minimizar el tiempo de estudio y maximizar los resultados que se puedan obtener. Partiendo de las posibilidades que ofrece el machine learning, se plantean los objetivos que se pretenden conseguir mediante la realización de este proyecto:

- Investigar en la búsqueda y creación de una base de datos útil para la detección de mentiras mediante el análisis de vídeos y audios.
- Mediante la aplicación de técnicas de visión computador y reconocimiento de audio, generar un clasificador para determinar si una persona está mintiendo.
- Estudiar las distintas metodologías existentes para este estudio y generar un sistema que integre dichas técnicas para la detección de mentiras fiable en el contexto que nos ocupa.

En este trabajo se busca analizar como se manifiesta la inteligencia humana en el campo de las mentiras, a fin de entender como funciona el cerebro humano durante los discursos no veraces y observar en qué aspectos tanto de la voz como de las emociones faciales se

exhiben. De esta forma, ampliar la frontera de conocimientos existente sobre este área de la psicología y ofrecer un sistema que permita reconocer y clasificar dichos aspectos para dirimir si una persona ha sido sincera o no. Además, habitualmente las personas no tienen mayor capacidad para determinar si otra persona está mintiendo que el azar (Ekman, s.f.). Por otro lado, el empleo del polígrafo para la detección de mentiras no suele ser eficiente, aunque estudios aseguran que tiene un 87% de acierto, los científicos de la Academia Nacional de Ciencias de Estados Unidos, determinaron que el polígrafo era bastante menos preciso de los que los examinadores del polígrafo habían determinado, otros científicos aseguraban que el porcentaje real del polígrafo era cercano al 75% (*Do Lie Detector Tests Really Work? | Psychology Today*, s.f.). Añadido a lo anterior, el polígrafo es un sistema que solo funciona si el investigado cree sinceramente que el polígrafo es capaz de detectar mentiras y por tanto tener preocupación o ansia, lo cual también puede ocurrirle a un inocente que tema que no le crean.

Una de las claves que ha motivado este proyecto, es poder generar un sistema que permita ayudar a los investigadores de crímenes a determinar si una persona puede estar mintiendo para encubrir un delito, por ejemplo de asesinato. Casos tan famosos como el de José Bretón (*Cronología del 'Caso Bretón'*, s.f.), quien asesinó a sus dos hijos y calcinó los cadáveres para ocultar pruebas que lo pudieran incriminar, simulando que sus hijos habían sido secuestrados y generando una trama bien hilada para generar una coartada perfecta o el caso de Mariluz Cortés (*Mari Luz Cortés, el llanto que se hizo multitud y reescribió el Código Penal | España*, s.f.), una niña de 5 años que fue violada y asesinada a manos de un vecino de su zona, el cual logró eludir la cárcel durante varios meses hasta que se descubrió que realmente había cometido dicho delito. Probablemente, si hubiera existido un sistema de detección de mentiras como el que se va a desarrollar en este proyecto, se podrían haber agilizado los procedimientos para incriminar a los asesinos analizando sus discursos.

Por último, se va a presentar la estructura que seguirá esta memoria. Comenzando por el marco teórico [2], se presentarán las claves psicológicas de la detección de mentiras así como los trabajos previos que han generado sistemas para dicha detección. Seguidamente, en el apartado de metodología [3], se explicarán las herramientas empleadas en este proyecto y cual es su cometido. A continuación, se presenta el desarrollo [4] de las funciones y algoritmos creados empleando las herramientas anteriores para generar el clasificador que determinará si una persona ha mentido o no. Se continuará con el apartado de resultados [5], en el cual se expondrán los modelos que se han creado para el sistema clasificador, junto a una comparativa de dichos modelos. Para concluir, el apartado de conclusiones [6] cuyo contenido será un resumen con la valoración del trabajo realizado.

2. Marco Teórico

2.1. Psicología

Contar una mentira es más agotador para el cerebro que decir la verdad. Basándose en la carga cognitiva que sufre el cerebro humano al contar una historia, se puede determinar cuál es mentira y cuál no, principalmente porque para elaborar una mentira o contarla se precisa de mayor carga cognitiva en nuestro cerebro que cuando se dice la verdad. Este enfoque de detección de mentiras mediante la carga cognitiva consta de tres técnicas fundamentales: (1) imponer una carga cognitiva, por ejemplo, pidiendo que se cuente la historia comenzando por el final de la misma, (2) alentar a proporcionar más información, un mentiroso tardaría mucho en inventar nueva información que sea coherente, y (3) hacer preguntas inesperadas, un mentiroso no sería capaz de responder rápidamente a estas preguntas, si dijese la verdad tardaría menos de un segundo en dar una respuesta (Vrij y cols., 2017).

Existen múltiples factores que pueden llegar a dar indicios de que un individuo está mintiendo o engañando, dentro de estos, uno de los más relevantes para la detección es la voz. Entiéndase por voz a todo lo que incluye el habla, no solo a las propias palabras. Dentro de los indicios vocales que denotan engaño, los más comunes son las pausas que cuanto más alargadas y en mayor frecuencia se generen, seguidos de una velocidad baja del habla, mayor es la probabilidad de encontrar una mentira en el discurso. Sobretodo esto sucede cuando se responde a una pregunta y en caso de que el sujeto no se haya preparado el discurso; en caso contrario, si el discurso ha sido preparado, se observará un aumento de la velocidad del habla y una reducción drástica de las pausas durante el discurso. Asimismo, el tono de la voz también es una característica que suscita que exista engaño en el discurso. Además, este es uno de los signos de la voz que está más documentado. Aproximadamente un 70% de los sujetos estudiados, presentan una elevación del tono de voz cuando se encuentran bajo el influjo de una perturbación emocional (Ekman, s.f.). De hecho, comparando la tonalidad entre un discurso sincero y uno en el que existe engaño, la tonalidad se vuelve más aguda aumentando la frecuencia del tono de la voz, en aquellos discursos donde el sujeto mantiene un discurso no veraz. Por tanto, si durante el discurso aumenta la tonalidad de la voz, se podría deducir que aumenta la probabilidad de que el individuo no esté siendo sincero en su discurso (Ekman y cols., 1976). Aunque, un aumento de la tonalidad durante el discurso no es signo de engaño por sí solo, es signo de temor, rabia o tristeza e incluso en algunos casos de excitación. Por otro lado, la ausencia de signos vocales de la emoción, tampoco implica que el discurso sea sincero. Es por ello que este signo debe evaluarse en conjunto con otros signos que evalúen la veracidad de un sujeto mediante la voz(Ekman, s.f.).

2.2. Técnicas Aplicadas de Aprendizaje Automático

Los métodos convencionales para detectar engaños como el polígrafo o mediante electroencefalograma no son suficiente para detectar una mentira, pues el individuo puede llegar a prepararse para estas pruebas y aunque es complicado puede llegar a engañar a la máquina. Sin embargo, hay algo que nunca se puede evitar y es nuestro comportamiento o lenguaje no verbal y nuestros gestos imprevistos, ya que el 70% de la comunicación del humano se basa en el lenguaje no verbal. Partiendo de esta premisa, se han buscado nuevos métodos que exploren el empleo de detección de engaño mediante métodos innovadores, en este caso, con detección de gestos en las manos de personas en videos RGB de juicios famosos. La metodología empleada para dicha tarea se basa en extrapolar un esqueleto de las manos de los individuos en los vídeos, a través del empleo de OpenPose, aplicarles ciertas técnicas de extracción de características y guardar estos datos en un vector de características (Fisher Vector). Una vez realizado esto, emplear estos datos para alimentar la red neuronal Long-Short Term Memory (LSTM). Entre estos dos métodos, se consigue discernir que ponencias en los juicios contienen engaños y cuáles no, debido a que el uso del Fisher Vector ayuda a la red LSTM a clasificar qué es un engaño y que no. El punto fuerte de esta metodología es que se puede llegar a implementar en tiempo real. De hecho, es tan precisa que permite detectar el engaño con hasta un 90.96% de acierto (Avola y cols., 2020).

La detección de mentiras mediante el lenguaje corporal no solo tiene que centrarse en el movimiento del torso del cuerpo o de sus extremidades, dado que gran parte de las mentiras pueden ser detectadas a raíz de microexpresiones en la cara o en el movimiento de los ojos. Además se debe tener en cuenta que esto no es válido para todo el mundo, depende en gran parte del tipo de género, etnia o cultura a la que pertenezca el individuo, es por ello que también es importante la variabilidad de personas para el estudio. Para ello, se emplea el sistema Silent Talker que se encarga de discernir qué expresiones indican mentira y cuales no en función de las características extraídas de los vídeos RGB. Por otro lado, se emplea OpenCV para el reconocimiento facial mediante el algoritmo Haar Cascade y así extraer las características que empleará el Silent Talker. Para el análisis de los datos se emplea Principal Component Analysis (PCA) junto a Self-Organizing Maps (SOM) y por último, para clasificar el engaño, se entrena una red neuronal con el algoritmo Random Forest y se pasa por el clasificador Support Vector Machines (SVM). Mediante el empleo de estos métodos, se logra alcanzar un 78% de acierto en la detección. (Khan y cols., 2021).

Según (Gallardo-Antolín y Montero, 2021) uno de los elementos de análisis de la detección de mentiras es la mirada y el movimiento que realizan los ojos cuando se miente. La importancia del movimiento de los ojos, reside en que es uno de los principales elementos expresivos de la cara y por ello, es conveniente estudiar la cantidad de parpadeos, los movimientos espontáneos no asociados a un estímulo visual específico, la dirección de la mirada e incluso la dilatación de las pupilas, ya que esta última característica, según varios estudios, está intrínsecamente relacionada con la mentira. Para lograr este cometido, se parte del dataset *Bag of Lies* (Gupta y cols., 2019), esta es una base de datos multimodal compuesta con información de vídeos, audio, mirada y datos de encefalogramas grabado a partir de 35 sujetos (25 masculinos y 10 femeninos). A partir de esta base de datos, se extraen las características referentes a la mirada, concretamente, los puntos de fijación de la mirada horizontales y verticales, y por otro lado, el diámetro de las pupilas del ojo izquierdo y derecho de los sujetos de la base

de datos, cabe destacar que los datos de las pupilas se fusionaron debido a la alta correlación entre estos dos parámetros, calculando la media de los datos de la pupila izquierda y derecha. A partir de estos datos, por un lado se entrena la red neuronal LSTM y por otro lado, se clasifican los datos con SVM de forma que se comparen los dos modelos y ver cuál es más eficiente en la detección. Los resultados obtenidos para estos datos son un 55.67% mediante el empleo de SVM y un 61.88% con la red LSTM.

Otro de los aspectos importantes en la detección de mentiras es la voz. Varios estudios psicológicos afirman que la acústica de la voz transmite información útil sobre el comportamiento veraz o no veraz de una persona. Dentro de la voz, las características prosódicas que cambian cuando una persona miente son la velocidad del habla o la tonalidad de la voz, la cual tiende a aumentar cuando una persona miente, o la energía de la voz que también sufre variaciones cuando se miente. Otro aspecto importante, son las pausas que se realizan durante el discurso. (Gallardo-Antolín y Montero, 2021) propone emplear redes neuronales LSTM con información de los espectrogramas empleando los *MFFCs*, a partir de los audios de la base de datos (Gupta y cols., 2019). Los *MFFCs* son coeficientes para la representación del habla basados en la percepción auditiva humana, este sistema ya se ha empleado en otros trabajos para crear sistemas de detección de la depresión o para la clasificación del sonido ambiental. Al igual que con la mirada, también propone el uso del clasificador SVM como comparativa con la red LSTM. En este caso, el sistema LSTM lograba un porcentaje de acierto en la detección de discursos no veraces del 63.89%, mientras que el clasificador SVM lograba un 60.31% de acierto.

3. Metodología

En este apartado se mostrarán todas las herramientas y recursos que se han empleado para llevar acabo el proyecto. Se explicará tanto la generación de la base de datos empleada, así como las herramientas, métodos y funciones para analizar los vídeos y audios de dicha base de datos. Todos los códigos creados con las herramientas que se van a emplear a continuación, se encuentran en `https://github.com/Tamai1306/TFG-Clasificador-Autom-tico-para-la-Detecci-n-de-Mentiras.git`.

3.1. Generación de la base de datos

En primer lugar, se ha investigado sobre posibles datasets que incluyan información de vídeos y audio que permitan dicha detección. Principalmente, es necesario que el dataset incluya vídeos sobre personas mostrando discursos veraces y discursos no veraces, en los cuales, se pueda extraer información de la señal de audio de la voz del individuo que está realizando el discurso, con la mayor nitidez posible, a fin de que durante su discurso no exista ruido u otras voces que interfieran en el posterior reconocimiento de audio y de extracción de características. Asimismo, en los vídeos a emplear, se debe mostrar con la mayor resolución y nitidez posible tanto la cara de la persona como su torso, para poder analizar tanto la emoción que esté trasluciendo su rostro a lo largo de la interacción como los movimientos corporales que realice su torso. Es por ello, que se ha decidido generar una base de datos que contenga estas características para poder analizarlas posteriormente y hacer el clasificador lo más robusto posible. Además, al generar una base de datos propia, se puede controlar de forma precisa cómo deben estar recogidos los datos, que posteriormente se emplearán en el clasificador.

Partiendo de las propiedades que debe contener la base de datos, fue necesario idear un sistema de pruebas en el que los individuos entrevistados tuvieran que mostrar discursos veraces y no veraces, que el investigador grabaría para su posterior análisis. Cabe destacar que este sistema de pruebas debía originar un aliciente en los individuos entrevistados, a modo de que los entrevistados mantuvieran una respuesta emocional fuerte, de forma que, sufrieran recelo a ser descubiertos y satisfacción en caso contrario.

De esta forma, los entrevistados participaron en un juego en el cual su motivación principal era lograr engañar al investigador. El juego consistía en que tanto el investigador como el entrevistado tenían una caja, teniendo en cuenta que el contenido de una de ellas era un billete de 50 euros, mientras que la otra contenía un folio en blanco. El objetivo del entrevistado era quedarse con la caja con el dinero, el del investigador por su lado, descubrir si existía engaño por parte de su contrincante y determinar qué caja contenía el dinero. Las reglas del juego son las siguientes:

1. El entrevistado al inicio de la prueba, visualizaba el contenido de su caja y por tanto averiguaba la posición del dinero.

2. El entrevistado trataba de convencer al investigador, que no conoce la caja que contiene el dinero, de mantener la caja que ha abierto o pedirle al investigador que moviera las cajas, debido a que el investigador es el único que puede cambiar la posición de las cajas. Así pues, el objetivo del entrevistado era convencer al investigador de mover las cajas o dejarlas en la posición actual con el fin de quedarse el dinero y el investigador tratar de descubrir la veracidad del discurso del entrevistado.

3. El juego acaba cuando el investigador abre el contenido de la caja que tiene en ese momento y así se descubre quien ha logrado quedarse con el dinero.

4. El único requisito del juego es que en al menos en una de las tres ocasiones en la que se jugase, aunque podían ser más, el entrevistado debía ser completamente sincero y decirle claramente al investigador donde se encontraba la caja con el dinero.

La realización de esta prueba no solo permitía recoger datos sobre si los entrevistados estaban siendo sinceros o trataban de engañar al entrevistador, también permitía vislumbrar las diferentes estrategias que puede emplear una persona para engañar a otra. Por ejemplo, es necesario destacar que una forma de mentir o engañar a otra persona es contar la verdad de una forma exagerada y así hacer creer a quien le está escuchando que su discurso es mentira y por ende engañarle. Asimismo, otras estrategias para engañar son el ocultamiento de la totalidad o parte de la información veraz y la más habitual, falsear el discurso (Ekman, s.f.).

Para concluir con este apartado, el investigador después de cada intervención con los entrevistados, debía apuntar si el entrevistado le había sido sincero o le había logrado engañar empleando alguna o varias de las estrategias anteriores. De este modo, cuando posteriormente se extraigan las características a analizar tanto del vídeo como del audio y estas se introduzcan en el clasificador, se podrá comprobar si el clasificador funciona correctamente o no y obtener su porcentaje de acierto, como se mostrará en la sección 5. Tras realizar varias pruebas, se ha generado una base de datos con un total de 25 vídeos y audios, de los cuales en 8 de ellos la persona ha sido sincera y en los otros 17 la persona ha mentido o engañado, además, 3 de las pruebas han sido realizadas a personas de género femenino y el resto a personas de género masculino. Para identificar qué prueba pertenece a cada género y si ha sido sincera o no, se ha generado un archivo *.txt* que lo especifica, como se explicará en el apartado 4. El dataset se encuentra ubicado en la plataforma *kaggle* y se puede acceder a él mediante el siguiente enlace:

`www.kaggle.com/dataset/`
`6bb95f89ef2bfd8df571ad3cc6e70f862d198e6748bd7ba807543a9d3589c7c5`

3.2. MATLAB

(*MATLAB*, s.f.) es un sistema de cómputo numérico que ofrece un entorno de desarrollo integrado con un lenguaje de programación propio. Este sistema tiene entre sus prestaciones básicas la manipulación eficiente y rápida de matrices, la representación de datos y funciones, la implementación de algoritmos, la creación de interfaces de usuario y la comunicación con programas en otros lenguajes y con otros dispositivos hardware. Además, este sistema tiene

ampliaciones llamadas *toolboxes* las cuales permiten aumentar las capacidades de MATLAB para la realización de diferentes tareas.

Dentro de las capacidades que ofrece MATLAB, para este proyecto el empleo de este sistema agilizaba la extracción de características de audio, gracias a las diversas *toolboxes* que incluye para el reconocimiento y procesamiento de audio, además de por la facilidad de carga y empleo de las mismas, tan solo es necesario buscar la *toolbox* que se necesite, descargarla y cargarla en el *path* de MATLAB. En concreto, se empleó inicialmente la *toolbox* (*Audio Toolbox*, s.f.), pues brinda herramientas para el procesamiento de audio, análisis de la voz y la medición acústica. Dentro de sus algoritmos, permite la ecualización, la extensión de tiempo, estimar métricas de las señales acústicas, tales como volumen y la nitidez, y extraer características del audio, como los MFCC y el tono de la voz. Añadida a esta *toolbox* se añadieron algunas más que cumplimentaban a esta y permiten la extracción de otras características. Así mismo, esta *toolbox* incluye herramientas para segmentar la señal de audio en trozos donde se detecta que la persona está hablando y descartando aquellas zonas donde no haya audio o exista ruido.

3.2.1. VOICEBOX: Speech Processing Toolbox

Dentro de las *toolboxes* existentes que analicen y procesen audio en MATLAB, se precisaba de una que tuviera implementados varios algoritmos que permitieran extraer características como la tonalidad de la voz *(pitch)* y la energía o intensidad de la misma de la forma más precisa posible a lo largo del espectro del audio de la voz. La *toolbox* (*VOICEBOX*, s.f.) incluye una gran variedad de funciones y algoritmos que permiten la extracción de las características anteriores, así como, funciones para analizar la señal de audio y hacer un procesamiento para el reconocimiento del habla.

3.2.2. The SpeechMark MATLAB Toolbox

Prosiguiendo con las *toolboxes*, era necesario encontrar alguna herramienta que permitiera la detección de la velocidad del habla de una persona. Existen diversas formas de hacerlo, en este caso se ha optado por calcular la velocidad del habla en sílabas/segundo. La *toolbox* (*The SpeechMark MATLAB Toolbox*, 2015) contiene algoritmos que detectan y miden los cambios de las señales acústicas del habla, concretamente, es capaz de realizar la detección automática de puntos de referencia acústicos abruptos y máximos en el habla, como por ejemplo la apertura y cierre de la glotis. Posteriormente, procesa las secuencias de puntos de referencia generadas en una estructura de ”sílabas” que revela algo sobre la complejidad de los enunciados en la fuente, incluidas las regiones con voz (dentro de las cuales las mediciones de tono son significativas), y ciertas regiones de ”pseudohabla” que pueden clasificarse como ruido no vocal.

3.3. DeepFace

Como ya se ha comentado anteriormente, también se iban a analizar las emociones que pudiera mostrar la persona que se encontrase ante el investigador. De esta forma, junto a los datos extraídos del audio, darle una mayor robustez al sistema. Para este análisis, era necesario una herramienta que tuviera la capacidad de hacer un reconocimiento facial de la

persona y extraer información de las microexpresiones que estuviese trasluciendo, a fin de asociarlas a una emoción u otra.

DeepFace (Serengil, s.f.) es un *framework* desarrollado en el lenguaje *Python* capaz de realizar un reconocimiento facial y de extraer de dicho reconocimiento información relevante como edad, género, emoción y etnia. Se trata de un sistema que combina las funcionalidades de otros sistemas de reconocimiento facial tales como VGG-Face, Google FaceNet, OpenFace, Facebook DeepFace, DeepID, ArcFace, Dlib y SFace, basados en redes neuronales y *deep learning*. En conjunto, este sistema logra obtener un porcentaje de acierto del 97.53% para el reconocimiento facial. En la Figura 3.1 se puede observar el resultado de aplicar este *framework* sobre varias imágenes en las que se muestran distintas emociones.

Figura 3.1: Ejemplo de reconocimiento de emociones con el *framework* Deep Face. (Serengil, s.f.)

Añadido a la funcionalidad que se requiere para este proyecto, también incluye funciones y algoritmos capaces de reconocer si en dos imágenes se encuentran la misma cara, para verificar imágenes. Incluyendo además, dentro de las funciones de este *framework* el poder elegir el *backend* del detector que se quiera emplear y así en función de las imágenes o la tarea donde se quiera aplicar el *framework* emplear el detector que mejor se ajuste a los requerimientos, debido a que, el empleo de un detector u otro en la alineación y detección de rostros puede hacer variar el porcentaje de acierto hasta en un 1%, *DeepFace* integra en su sistema los detectores de la Figura 3.2.

3.4. MediaPipe

Por otro lado, era conveniente encontrar un sistema que fuera capaz de reconocer gestos o movimientos corporales de forma precisa, para poder obtener más datos que pudieran aumentar la robustez del clasificador. Actualmente, uno de los *frameworks* más potentes para el análisis de video y audio es (*MediaPipe*, s.f.). Este *framework* desarrollado por Google

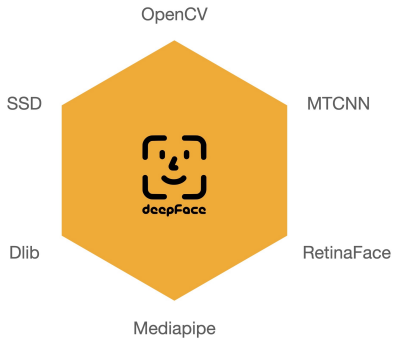


Figura 3.2: *Backends* de detectores para el reconocimiento y alineamiento facial. (Serengil, s.f.)

permite el desarrollo de *pipelines* con *machine learning* para procesar en tiempo real, datos sobre vídeo y audio. El potencial de este sistema, reside en su poder para emplear herramientas y *frameworks* de *machine learning* empleando la menor cantidad de recursos posibles, de hecho, es tan pequeño y eficiente que se puede emplear incluso en sistemas IoT (*Internet of Things*) y embebidos.

Este *framework* está pricipalmente contruido en *C++* debido a que es un lenguaje muy eficiente a la hora de trabajar con datos en memoria y emplea para las detecciones la librería para visión por computador (*OpenCV: OpenCV modules*, s.f.). No obstante aunque esté desarrollada en *C++*, incluye también adaptaciones de las funciones y herramientas para otros lenguajes, en el caso que ocupa el marco de este proyecto, se ha empleado este sistema con *Python*. Concretamente, de este *framework* se han empleado sus ”soluciones” que son ejemplos preconstruidos de código abierto con modelos preentrenados con *Tensorflow* (*TensorFlow*, s.f.) para la detección de la forma y movimiento de las manos, del torso e incluso de puntos de la cara mediante reconocimiento facial. El potencial de este tipo de ”soluciones” es que se pueden emplear en tiempo real y es capaz de extraer los datos de las posiciones de las manos, etc, de forma muy precisa y con un error bajo.

Para este proyecto, se ha optado por el empleo de la ”solución” que se encarga de la detección de la pose de una persona (*Pose*, s.f.). Esta herramienta permite entre otras cosas, calcular los puntos clave de las articulaciones del cuerpo, por ejemplo, para posteriormente emplearlos en un sistema que se encargue de verificar si se ha realizado un ejercicio físico en concreto, o incluso para verificar que se ha llevado acabo una postura de yoga. Básicamente, esta herramienta, se encarga de trackear con una alta fidelidad 33 puntos de referencia en 3D, estos puntos están reflejados en la Figura 3.3, y segmentar una máscara del fondo en todo el cuerpo a partir de fotogramas de vídeos RGB.

Utilizando un detector, el *pipeline* localiza primero la región de interés de la persona/postura dentro del fotograma. A continuación, el rastreador predice los puntos de referencia de la postura y la máscara de segmentación dentro de la región de interés, empleando el fotograma recortado de la región de interés como entrada. Hay que tener en cuenta que, en los casos de uso de vídeo, el detector sólo se invoca cuando es necesario, es decir, para el primer

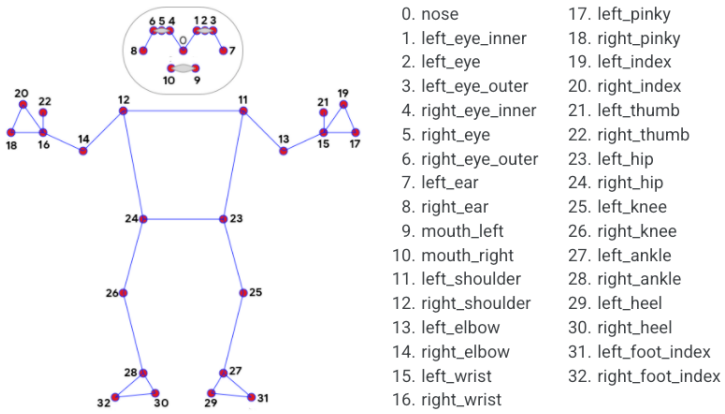


Figura 3.3: Puntos de referencia de detección de la pose del cuerpo humano. (*MediaPipe*, s.f.)

fotograma y cuando el rastreador ya no puede identificar la presencia de la pose del cuerpo en el fotograma anterior. Para el resto de fotogramas, el *pipeline* simplemente deriva el la región de interés a partir de los puntos de referencia de la pose del fotograma anterior.

Partiendo de estas herramientas, se seguirá el diagrama de flujo de la Figura 3.4. En los apartados posteriores se explicará detalladamente cada bloque del diagrama, a fin de mostrar el desarrollo seguido para confeccionar este proyecto.

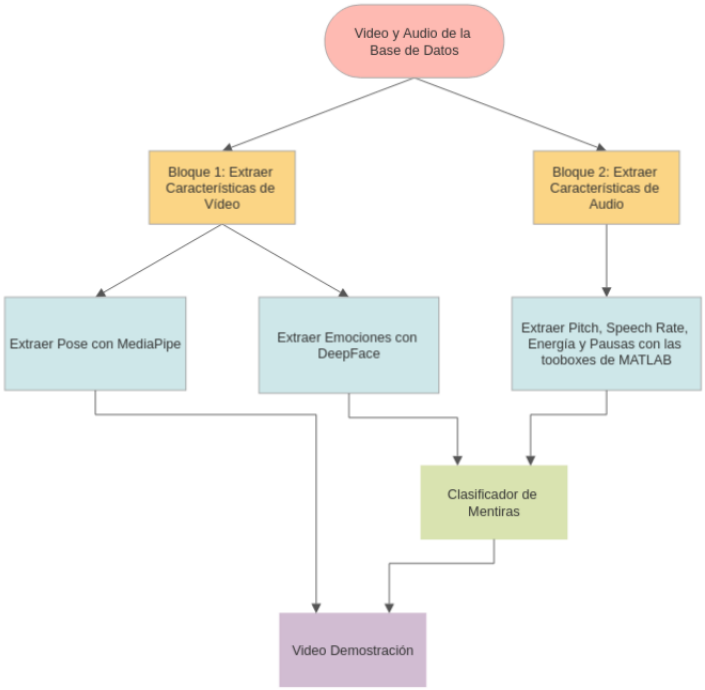


Figura 3.4: Diagrama de Flujo del Proyecto.

4. Desarrollo

Prosiguiendo con el desarrollo de este proyecto, se van a presentar como se han empleados los métodos y algoritmos anteriores para la extracción de características, tanto del audio empleando MATLAB como del vídeo, empleando tanto *DeepFace* como *MediaPipe*. Se ha optado por dividir la explicación en dos bloques principales, cuyo contenido será el siguiente:

- Bloque 1: Este primer bloque, se va a centrar el estudio en el análisis de técnicas y herramientas que permitan extraer características del audio de una persona durante una conversación, a fin de determinar si su discurso es veraz o contiene indicios que sugieran que la persona no esté siendo sincera.

- Bloque 2: El segundo bloque se encargará de emplear técnicas y herramientas de visión por computador de forma que se analicen las emociones ilustradas en la cara del individuo, así como los movimientos de su torso, a fin de añadirle robustez a las características extraídas en el bloque anterior.

Seguido de esta división, se explicará como se han combinado los dos bloques con el fin de generar el clasificador encargado de discernir si los elementos de la base de datos pertenecen a discursos veraces o no veraces.

4.1. Bloque 1: Análisis del engaño mediante el estudio de la voz

Previamente a explicar este bloque, hay que destacar que de la base de datos generada para este proyecto, se han extraídos los audios de cada entrevistado, dejando solo las partes en las que habla el entrevistado para que el sistema pueda centrarse en extraer las características solo de su voz y así evitar que se vea influido también por la voz del investigador. Estos archivos de audio se han codificado en formato *.wav*, pues en este formato la señal de audio se guarda como una onda y simplifica la forma en la que se va a trabajar con los datos, además de porque los algoritmos de extracción de características funcionan con este formato. Añadido a esto, se han guardado todos estos archivos en el mismo directorio, junto a un archivo que incluye el nombre de cada archivo, seguido del género de la persona a la que pertenece el audio y por último, un valor que indicará *true* si en el audio el entrevistado ha sido sincero o *false* si ha tratado de mentir o engañar al investigador. Es necesario tener en cuenta que si en alguno de los casos, el entrevistado dice la verdad pero de forma exagerada para tratar de engañar al entrevistador, esto también se ha considerado como mentira (Ekman, s.f.). Para ejemplificar el contenido de este archivo, se ha generado la siguiente Figura 4.1.

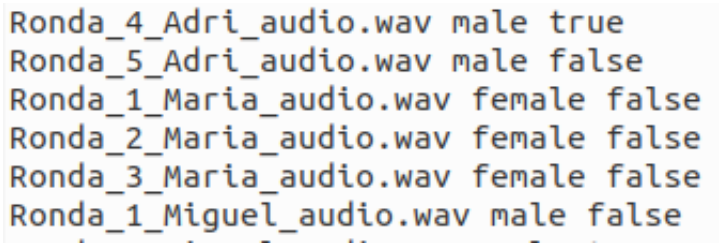


Figura 4.1: Archivo con los nombres de los audios, género de cada audio y valor de veracidad.

Seguidamente, es necesario introducir previamente los conceptos y las características a extraer de la señal de la voz. Dentro de las características a extraer, las principales son la tonalidad de la voz *(pitch)*, la velocidad del habla *(speech_rate)*, la energía o intensidad de la voz y las pausas durante el habla (Appelgren, s.f.). Estas son las características principales a extraer partiendo de la base de que contar la verdad y contar una mentira tienen formas distintas de expresión que se manifiestan principalmente en estas características. La clave reside en que el engaño o la mentira tienen una gran influencia sobre las emociones que presenta una persona durante su discurso, lo cual afecta directamente a la velocidad, el *pitch* y la energía del hablante, de forma habitual, si hay una carga emotiva grande se mostrará en un aumento del *pitch*. Asimismo, como ya se ha mencionado anteriormente, contar una mentira y mantener el engaño durante el discurso genera una mayor carga cognitiva al cerebro y esto afecta a la velocidad del discurso y, principalmente, a la duración y número de pausas que realiza el individuo, generalmente una mentira requiere de un mayor número de pausas para controlar la coherencia del discurso y lograr mantener la mentira junto a un descenso de la velocidad, aunque si por ejemplo, la mentira ha sido preparada, habrán menos pausas y de menor duración, junto a un aumento de la velocidad (Ekman, s.f.).

Partiendo de las premisas anteriores, se ha realizado un estudio de evaluación de herramientas que permitieran el análisis y extracción de características, tras su evaluación, se ha concluido que el empleo de MATLAB satisface las necesidades requeridas para lograr estos objetivos. Se ha optado por su utilización debido a la multitud de *toolboxes* existentes de acceso libre que contienen herramientas para trabajar con ondas de audio y poder extraer características, como ya se mencionó en el apartado de 3. La extracción de características seguirá el diagrama de flujo de la Figura 4.2:

Teniendo presente el diagrama, se procede a esbozar el núcleo principal del conjunto de los pasos, así como los inconvenientes encontrados y las soluciones propuestas.

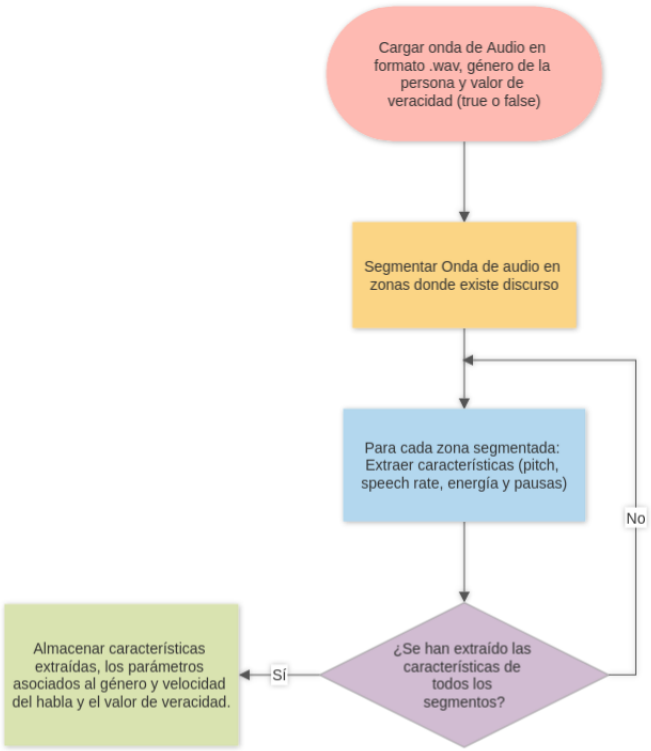


Figura 4.2: Diagrama de Flujo extracción de características de audio.

4.1.1. Segmentación de la onda de audio

Comenzando por la segmentación de la onda de audio, se ha empleado la *toolbox* de audio que proporciona de base *Matlab* (*Audio Toolbox*, s.f.). Dentro de dicha *toolbox*, se encuentra la función *detectSpeech* a la cual se le pasan como parámetros la onda de audio en crudo, la frecuencia de muestreo y otro conjunto de parámetros para delimitar las zonas donde existe voz y donde no, tales como la *mergeDistance* que se encargar de unir zonas de audio que se encuentren con una separación menor a dicha distancia o el parámetro *thresholds* que se encarga de delimitar si alguna de las zonas evaluadas como audio, pudiera ser ruido. Inicialmente, se indica que *mergeDistance* sea igual a 1 segundo, de esta forma todos los segmentos de audio cuya duración sea menor o igual a esta distancia se fusionaran, generando segmentos en los que existe discurso. Esta función devuelve los índices del vector de la onda de audio donde ha detectado voz, que posteriormente, se usarán para extraer de cada zona segmentada las características.

4.1.2. Extracción de características de las zonas segmentadas

Partiendo de la segmentación de audio anterior, a continuación, se va a evaluar cada zona segmentada para extraer características por separado. Inicialmente, se extrae directamente la energía de la onda de audio de dicho segmento mediante la función *v_teager* de la *toolbox* (*VOICEBOX*, s.f.), a continuación, se guardan los valores extraídos en un vector cuyo tamaño

es igual al del vector de la onda de audio, almacenandose en la sección de vector que coincide con la zona segmentada.

Prosiguiendo con la extracción, el empleo de la *toolbox* (*The SpeechMark MATLAB Toolbox*, 2015) permite el cálculo de la velocidad del segmento de audio en *sílabas/segundo* gracias a la función *lm_syl_count*. Habitualmente, la velocidad del habla se mide en *palabras/segundo*, sin embargo, las herramientas existentes para extraer características y poder transcribir el audio a texto de forma gratuita sin necesidad de realizar consultas a inteligencias artificiales de la red, se encuentran solo para audios en inglés. Debido a ello, se realizó un análisis para encontrar otra herramienta que pudiera medir velocidad, aunque la unidad de medida fuera distinta a la estándar. Asimismo, también esta medida es robusta al idioma en el que se hable y aunque en cada idioma la cuenta de sílabas funciona de formas diferentes, esa variación genera un error de ± 1 sílaba en la detección, lo cual al calcular la velocidad con el resto de sílabas a lo largo del tiempo, no supone un error significante para el estudio que se ha realizado. Al igual que con la energía, los valores de velocidad serán guardados en un vector del mismo tamaño que el vector de la onda de audio, almacenando solo los valores que pertenecen a la sección analizada.

Por otro lado, se ha determinado que si la duración del segmento de audio es superior a 3 segundos, el segmento se dividirá en subsegmentos empleando la función *detectSpeech*, con *mergeDistance*=0.5 segundos y *thresholds*=[0.2 0] y de esta forma eliminar ruidos. Este valor se determinó tras observar que si se empleaba un valor menor, el calculo de pausas no funcionaba correctamente, ya que se quiere analizar la pausas superiores a 0.5 segundos, pues si la pausa es superior a este valor, se detecta que es probable que la persona esté sufriendo una alta carga cognitiva para elaborar una mentira.

Para calcular el tiempo del segmento de audio, se calcula la diferencia entre los índices del segmento y se divide entre la frecuencia de muestreo de la onda de audio. Esta nueva segmentación sirve para extraer las características anteriores y además analizar las pausas que se producen durante el habla, tomando en cuenta el intervalo de separación que existe entre los elementos subsegmentados, dichas pausas serán guardadas en una tupla donde se especifique el intervalo de tiempo donde se ha producido la pausa y su duración. En el caso de la velocidad, solo se calculará si la duración del subsegmento es superior a 1 segundo, pues si el subsegmento tiene una duración menor, no hay suficiente información para calcular bien las sílabas. Al igual que en el caso anterior, los valores de velocidad se guardaran en el vector asociada a la misma y almacenando los valores de la zona subsegmentada.

Por último, se extrae el *pitch* del audio al completo, esto se debe a como extrae la tonalidad la función *v_fxpefac* de la *toolbox* (*VOICEBOX*, s.f.) la cual analiza la señal de audio por frames y en intervalos de tiempo, el valor del intervalo de incremento se puede especificar como parámetro de la función, en este caso se ha empleado un valor de 50 ms, para lograr extraer la mayor cantidad de información asociada al *pitch* posible y evitando suavizar demasiado la señal resultante. Esta función devuelve un vector los valores de *pitch* y otro vector con los tiempos en los que se ha obtenido cada valor.

4.1.3. Almacenamiento de las características extraídas

Para concluir con este bloque, se va a mostrar como han sido almacenadas las características anteriores, para poder emplearlas posteriormente en el clasificador. Lo primero a tener en cuenta es que los vectores de energía y velocidad, tienen tamaños distintos al vector de *pitch*,

esto derivado de como extrae el *pitch* la función *v_fxpefac*. Por ello, hay que correlacionar los valores de tiempos entre los vectores, esto se realiza obteniendo los tiempos entre los que se han calculados los segmentos donde se han extraído las características, como se ha explicado anteriormente, el calculo se realiza dividiendo el valor del los índices entre los que se encuentra la zona segmentada entre la frecuencia de muestreo. Una vez obtenido el tiempo de inicio de la zona segmentada y el tiempo en el que finaliza, se interpolan dichos valores con el vector de tiempos del *pitch*, de forma que se obtengan los tiempos de *pitch* más cercanos a los tiempos de la zona segmentadas. Una vez obtenidos dichos tiempos, se busca en que índices del vector se encuentran y así poder calcular la media de *pitch* que existe entre esos índices del vector que contiene los valores del *pitch*. Asimismo, se obtienen también los valores medios de energía y de velocidad de la zona segmentada. Seguidamente, se guardan en una tupla el intervalo de tiempo de la zona segmentada, con los valores medios de *pitch*, energía y velocidad que se acaban de calcular. Esto también se realiza para las zonas subsegmentadas de los intervalos de audio que tengan una duración de más de 3 segundos.

Por último, se genera un archivo *.txt* que resume los valores de las características extraídas. Añadiendo en dicho archivo, tanto el género de la voz analizada, el rango de *pitch* asociado a dicho género, la energía media de la señal de audio entera, el número de pausas y se indica si la persona del audio ha sido sincera o ha mentido. Las líneas posteriores del archivo muestran los valores almacenados en las tuplas tanto de las características extraídas como de las pausas calculadas. Por otro lado, se almacenan las tuplas en formato *.mat* para posteriormente extraer dicha información en el clasificador.

4.2. Bloque 2: Análisis de las emociones y movimientos corporales

Continuando con el segundo bloque encargado del análisis del vídeo, se toman los vídeos creados en la base de datos para analizar *frame* a *frame* tanto las emociones como los movimientos del torso del individuo, concretamente, se va a centrar el estudio en el cálculo y reconocimiento del movimiento de hombros. Para lograr estos objetivos, se van a emplear la librería de (*OpenCV: OpenCV modules*, s.f.) y los frameworks *DeepFace* (Serengil, s.f.) y *MediaPipe* (*MediaPipe*, s.f.). La extracción de características de vídeo seguirá el diagrama de flujo de la Figura 4.3.

4.2.1. Detección del rostro y extracción de emociones

Anteriormente, se ha mencionado la utilidad que proporciona el *framework DeepFace* en este proyecto. En este caso concreto, se va a emplear la herramienta que permite el reconocimiento de emociones del rostro de una persona. Para ello, el *framework* se apoya en el uso del detector y clasificador *Haar Cascade* (*OpenCV: Face Detection using Haar Cascades*, s.f.) junto a *Opencv*.

Haar Cascade es un algoritmo basado en características para la detección de objetos que fue propuesto en 2001 por Paul Viola y Michael Jones. La implementación original se utiliza para detectar la cara frontal y sus características como los ojos, la nariz y la boca. Sin embargo, existe una implementación pre-entrenada disponible en su GitHub para otros objetos también como para el cuerpo completo, la parte superior del cuerpo, la parte inferior del cuerpo, la sonrisa, y muchos más. Básicamente, se trata de un método rápido de procesamiento de

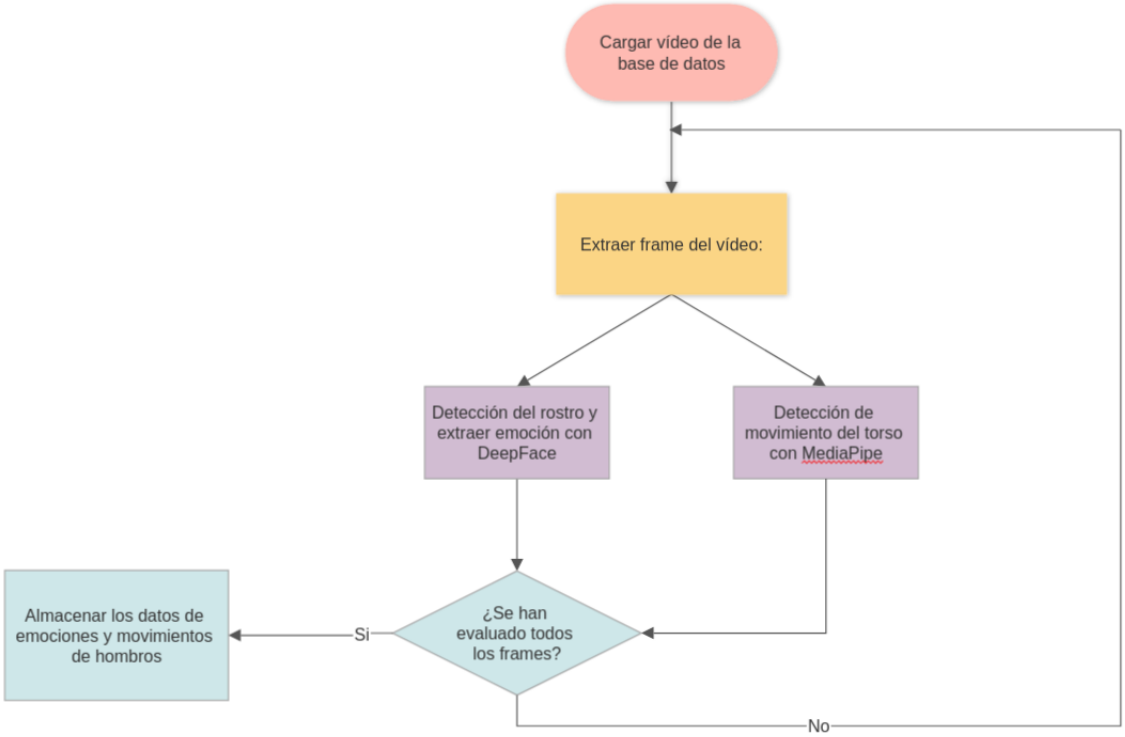


Figura 4.3: Diagrama de Flujo extracción de características de audio.

imágenes capaz de detectar las caras empleando para ello un conjunto de elementos como los que se muestran en la Figura 4.4. Estos elementos rectangulares se asemejan a los *kernels* empleados en la detección o segmentación de objetos en imágenes, en este caso, se emplean para detectar características de la caras como los ojos.

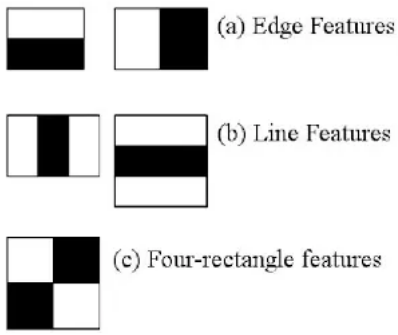


Figura 4.4: Elementos rectangulares para la extracción de características del algoritmo *Haar Cascade*. (*OpenCV: Face Detection using Haar Cascades*, s.f.)

Estos elementos, se deslizan por la imagen calculando la suma del píxel que se encuentra en la zona blanca y substrayéndola de la suma de los píxeles que caen en la zona negra del

elemento. Realizando esto por toda la imagen, el resultado es la acumulación de puntos clave con características dentro de la misma. En conjunto con una gran cantidad de imágenes en la que se hayan extraído estas características y guardadas en un mapa de características con un posterior entrenamiento mediante *machine learning*, generan un detector y clasificador de caras.

Una vez cargado el clasificador y detector *Haar Cascade* mediante *OpenCV*, se abrirá el vídeo en el cual se vaya a realizar la detección y se irá analizando el video *frame* a *frame*. Para cada frame, se convertirá la imagen a escala de grises, ya que simplifica los cálculos con el detector *Haar Cascade*, y se pasará el resultado por el detector, como resultado se obtendrá un vector con las posiciones x e y en píxeles del rectángulo que envuelve el rostro de la persona. A continuación, se analiza la imagen con la detección con el *framework* de *DeepFace* que, en caso de haberse detectado correctamente el rostro, extraerá las probabilidades de las emociones que se trasluzca del mismo, empleando para ello el *backend* de detección propio de *OpenCV*. Estas probabilidades serán almacenadas en vectores para su posterior muestreo. Añadido a esto, también se guarda la emoción dominante presente, es decir, aquella emoción con mayor probabilidad en la detección y también se guardará el tiempo en el que se ha detectado dicha emoción.

Tras realizar estos pasos por todos los *frames* del video, se guardará tanto el vector de tiempos en los que se haya encontrado emociones, como el vector de emociones dominantes, que posteriormente se empleará en el clasificador.

Por último, cabe mencionar que tanto los datos extraídos de las emociones y del movimiento de hombros que han sido almacenados en vectores, estos se guardarán en memoria empleando la función *numpy.savez* que incluye la librería *Numpy* (*NumPy*, s.f.) de *Python*, la cual permite guardar en un archivo comprimido los datos que se requiera.

4.2.2. Detección y reconocimiento de movimientos corporales.

Otro de los *frameworks* que se comentó en el apartado 3 es el *framework* de *MediaPipe* (*MediaPipe*, s.f.). También, se mencionó la utilidad que se le iba a dar en este proyecto, empleando su ”solución” de pose para detectar la pose del cuerpo humano y los puntos clave de la misma. Partiendo de los códigos de ejemplo que proporciona el *framework*, al igual que con el *framework* de *DeepFace*, se carga el vídeo donde se va a calcular la pose y para cada *frame* del vídeo, se recorta la imagen para dejar solo la zona que se quiere calcular, en este caso, los hombros. Una vez tomada la imagen recortada, se convierte de BGR a RGB, se procesa dicha imagen con el *framework* y se vuelve a convertir la imagen a BGR, que es el formato en el que opera *OpenCV*. Una vez calculada la pose, se estiman los 33 puntos de referencia y una vez realizado esto, se dibujan en la imagen los puntos con sus respectivas conexiones entre ellos. Tras calcular la pose, se obtienen las coordenadas de los puntos de los hombros, que son los puntos 11 y 12 respectivamente de la pose, concretamente, se extrae la *coordenada y* de cada punto y se almacena en un vector. Este proceso se hace de forma simultánea a la detección de emociones anterior.

Tras procesar todas las coordenadas de los dos hombros a lo largo del vídeo, se van a analizar los cambios que se producen en las coordenadas a fin de reconocer si en algún instante, la persona ha elevado alguno o ambos hombros durante su intervención. Este movimiento, habitualmente se traduce en un movimiento corporal subsconciente asociado a la duda, si se realiza en algún momento en el que en su discurso no hayan dudas, sino afirmaciones o

negaciones, se tomará este movimiento como un indicio de engaño (Ekman y cols., 1976). Para la detección de movimiento, hay que destacar como están dispuestos los ejes x e y de la imagen, estos se muestran en la Figura 4.5.

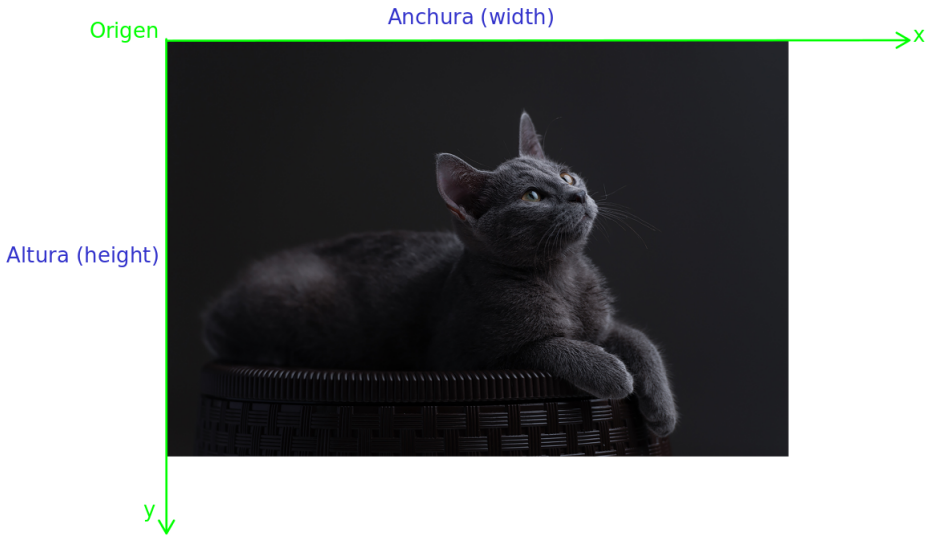


Figura 4.5: Disposición de los ejes de coordenadas en las imágenes de OpenCV.

En este caso, las coordenadas de los puntos tienen un rango entre 0 y 1, siendo para la *coordenada y*, el 0 el borde superior de la imagen y 1 el borde inferior. Por tanto, lo que se requiere detectar son cambios en los que el valor de la *coordenada y* descienda hasta un valor y, posteriormente, vuelva a ascender.

Una forma de poder detectar estas variaciones en la posición, es emplear una ”ventana dinámica” que se deslice por el vector de coordenadas en busca de estas variaciones. La idea se basa en tomar subvectores de una longitud concreta, en este caso 50 casillas, e ir deslizando esa ventana por todo el vector. Para cada ventana, se obtiene el valor mínimo de esa ventana y se compara con el valor de la casilla inicial y final de la ventana, si la diferencia entre el valor mínimo y el valor de estas dos casillas es mayor a un valor de error de 10^{-2}, se analizará si existe una secuencia descendente hasta el valor mínimo seguida de una secuencia ascendente desde ese valor mínimo hasta el final del vector. En el caso de que la suma del número de valores descendentes y ascendentes sea menor a la longitud de la ventana menos 1, se habrá detectado el movimiento de hombros que se requiere y se guardará en una matriz los valores de tiempo en los que se ha detectado dicho movimiento, posteriormente, se desplazará la ventana a los 50 siguientes elementos del vector de coordenadas, en el que el valor de inicio de esa nueva ventana sea el valor siguiente al valor final de la ventana actual en la que se haya detectado el movimiento.

4.3. Generación del clasificador automático

El objetivo de este apartado, es explicar cómo se han combinado los datos extraídos tanto en el bloque del audio, como en el del vídeo para detectar si existen indicios de mentira en el discurso analizado. Debido a que los datos del audio y del vídeo, están en formatos distintos, principalmente por haber empleado MATLAB para el audio y los *frameworks* en *Python* para el vídeo, era primordial encontrar alguna herramienta que permitiese cargar todos los datos juntos para poder analizarlos.

En este caso, una de las ventajas que tiene *Python* es que es un lenguaje que contiene multitud de librerías y *frameworks* para trabajar de forma rápida y eficiente con datos en diferentes formatos. Concretamente la librería *SciPy* (*SciPy*, s.f.), permite cargar los archivos *.mat* generados con MATLAB. Por tanto, el primer paso para generar el clasificador automático es cargar todos los datos extraídos tanto en el audio como en el vídeo. Trabajar con los datos de forma directa puede ser muy engorroso y complicado, para simplificar el acceso a los datos y realizarlo de forma rápida y eficiente, se ha decidido almacenar los datos en vectores de diccionarios de *Python*, de forma que para acceder a los datos, simplemente hay que indicar la etiqueta del diccionario que contiene la información a la que se quiere acceder.

Partiendo de la información anterior, se necesitaba de un algoritmo que pudiera clasificar en base a los datos anteriores si una persona está siendo sincera o, por el contrario, está mintiendo. Tras valorar distintas estrategias y algoritmos, se observó que lo más eficiente era generar un árbol de decisión que fuera ponderando los datos obtenidos anteriormente. Teniendo en cuenta que donde hay más volumen de datos e información es en los datos de audio, el árbol de decisión se iba a centrar en ponderar principalmente estos datos. Su generación se basa en estudiar cómo afectan las mentiras en cada uno de los datos, como ya se ha mencionado con anterioridad. Siguiendo de esta premisa, se generó el siguiente árbol de decisión, presente en la Figura A.1.

Este árbol se encargará de ir leyendo el conjunto de datos de audio línea a línea e ir ponderando hasta obtener un valor final. Las ponderaciones se han realizado teniendo en cuenta el peso que tiene cada característica en el reconocimiento de mentiras y adaptando la ponderación en función de los datos. Cabe mencionar que este árbol se ha ido adaptando y mejorando conforme se han ido realizando pruebas como se comentará en el apartado 5.

Por otro lado, para añadirle robustez a la ponderación anterior se ha tomado en cuenta que cuando aumenta el *pitch*, se indica que la persona está bajo un influjo emocional fuerte, habitualmente el aumento del *pitch* se asocia al temor y un descenso por debajo de la media del rango de *pitch* se asocia a la tristeza. Por ello, en una segunda parte del clasificador, se han vuelto a analizar los datos de *pitch* y si el *pitch* supera el valor máximo del rango asociado al género de la persona y en esos instantes se detecta la emoción asociada al miedo, se aumenta el valor de α de la ponderación del árbol anterior. De forma similar, si el valor del *pitch* se encuentra por debajo de la media del rango del *pitch* asociado al género y se detecta que en esos instantes hay tristeza, también se aumenta la ponderación del valor de α.

En el caso de los movimientos corporales, no se han tomado en cuenta para estas ponderaciones, pues depende de haber detectado si hay incongruencias entre esos movimientos y las afirmaciones o negaciones que realice la persona investigada. Por ello, lo que se ha realizado es un aviso cuando se detecte este tipo de movimientos en el vídeo demostración que se presenta en el apartado 5, de forma que se notifique mediante un mensaje cuando se ha detectado una

mentira gracias al clasificador o mediante la detección de estos movimientos. En trabajos futuros, se buscará asociar este tipo de movimientos a la ponderación del clasificador para añadirle mayor robustez al sistema.

Por último, se ha valorado el resultado de α obtenido y su relación con la información existente de si el individuo ha sido sincero o no. A partir de esta información, se obtienen los valores de verdaderos positivos (TP), falsos positivos (FP), verdaderos negativos (TN) y falsos positivos (FP). En este proyecto, los verdaderos positivos serán cuando la persona haya mentido y el clasificador haya detectado que ha mentido; los falsos positivos cuando haya mentido pero el sistema haya detectado que ha sido sincera; los verdaderos negativos cuando haya sincera y se obtenga de resultado que ha sido sincera y por último; los falsos negativos, que se producirán cuando la persona haya sido sincera pero el sistema detecte que ha mentido. A partir de estos datos se valoró el sistema en términos de la precisión media del engaño engaño (DACC), la precisión media de la honestidad (HACC) y la tasa media de identificación o precisión (ACC), que se definieron mediante las siguientes ecuaciones:

$$DACC = \frac{TP}{TP + FP} \tag{4.1}$$

$$HACC = \frac{TN}{TN + FN} \tag{4.2}$$

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.3}$$

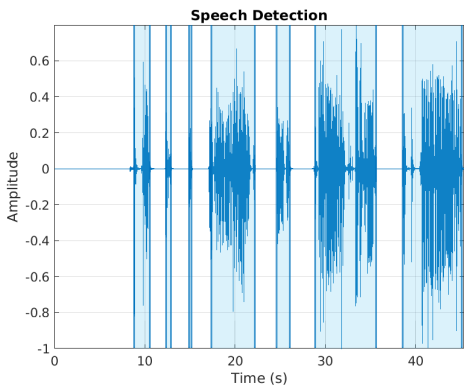
Estos valores indicarán cómo de eficiente es el clasificador que se ha generado y su utilidad, los valores resultantes se explicarán en el apartado 5.

5. Resultados

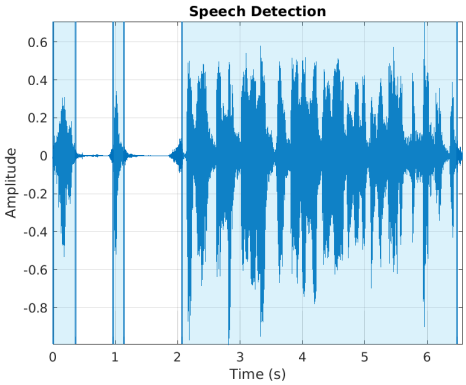
Siguiendo con la estructura de este proyecto, se van a presentar los resultados obtenidos tras la extracción de características de audio y vídeo, comparando los datos obtenidos entre una persona con discurso veraz y una con discurso no veraz, principalmente en los resultados obtenidos a través del audio que es donde mejor se observa dicha diferencia, para posteriormente, evidenciar los resultados obtenidos tras hacer varias pruebas con el clasificador y viendo los porcentajes de acierto que se obtienen con dichas pruebas. De forma similar a la estructura del apartado 4, se realizará una división de la exposición, a fin de detallar por partes los resultados obtenidos y llegando a una conclusión final.

5.1. Bloque 1: Análisis del engaño mediante el estudio de la voz

Una vez extraídas las características pertenecientes al audio y explicadas anteriormente, se procede al análisis de las mismas. Inicialmente, se va a presentar cómo la función *detectSpeech* mencionada anteriormente ha segmentado la onda de audio tanto para el audio al completo, Figura 5.1a, como en zonas cuya segmentación supera los 3 segundos, Figura 5.1b, y se ha realizado una subsegmentación para obtener también las pausas generadas, tal y como se muestra en la Figura 5.1.



(a) Segmentación de la onda de Audio



(b) Subsegmentación de audio para extraer las pausas.

Figura 5.1: Detección de habla en una señal de audio.

A continuación, se van a exponer las Figuras resultantes asociadas a cada características extraída durante el procesamiento de la señal de audio. Comenzando con la energía del habla, se genera la Figura 5.2a la cual muestra el valor de intensidad o energía del audio a lo largo del tiempo.

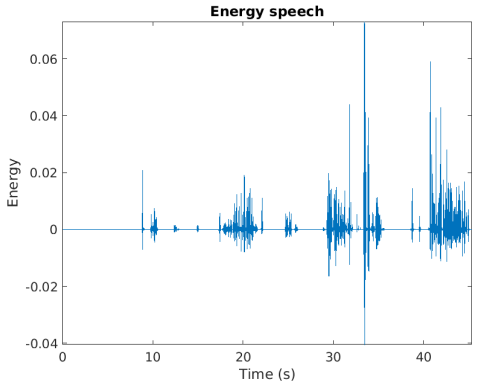
Añadido a lo anterior, se genera también una figura cuyo contenido muestra el *pitch* del habla a lo largo del tiempo. Para lograr evaluar el *pitch* es necesario tener en cuenta los rangos de frecuencia de las personas, cuya diferencia subyace principalmente por razones de género y edad (April 06 y 2020, 2015). Por ejemplo, no tiene el mismo de rango de frecuencias un niño cuya voz aún no está desarrollada, que un adulto que dependiendo de su sexo tendrá una tendencia más grave si es hombre o una tendencia más aguda si es una mujer. En el caso de estudio que ocupa este trabajo, se no se ha tomado en cuenta el análisis de la voz de un niño, debido esencialmente a que tienen la capacidad de modular la frecuencia de su voz en un rango más amplio que un adulto. Dentro de los rangos de *pitch* que tienen los adultos, los hombres mantienen un rango de tonalidad grave entre los 85 y 180 *Hz*, sin embargo, la tonalidad de las mujeres tiene una tendencia más aguda, entre los 165 y los 255 *Hz*. Cabe destacar que estos rangos pertenecen al habla habitual de las personas, puesto que, una persona cuando canta, por ejemplo, modula su voz de forma distinta, variando los rangos anteriores. Conociendo estos rangos, se podrá determinar cuando una persona se encuentra bajo un influyo emocional fuerte, por ejemplo, cuando su *pitch* supera el valor máximo del rango asociado a su género, como se ha comentado en ocasiones anteriores. La Figura resultante con esta característica es la Figura 5.2b

La velocidad media del habla de una persona durante una conversación oscila sobre las 4 *sílabas/segundo* (*Speech tempo*, 2022). A raíz de ello, conviene determinar un rango de velocidades que dirima cuando una persona está hablando a una velocidad rápida y cuando a una velocidad lenta. Debido a que cada persona mantiene unos rangos de velocidad variables y que este factor es también dependiente en muchos casos del idioma y de la cultura, se ha optado por generar un rango en el que la velocidad habitual del individuo se encuentre oscilando entre ±12% de la velocidad natural, es decir, que la velocidad normal del habla se encuentra entre las 3.5 y 4.5 *sílabas/segundo*. Al igual que en el caso anterior, tener en cuenta este rango permitirá dirimir y ponderar cuando una persona tiene un indicio de engaño o mentira. En la Figura 5.2c se muestra tanto la velocidad extraída para cada zona segmentada y los rangos de la velocidad explicados.

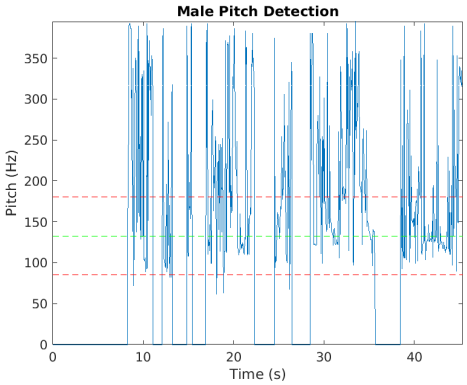
Por último, se analizan las pausas generadas durante la intervención del individuo cuando la zona segmentada del audio tiene una duración superior a los 3 segundos. Como ya se ha mencionado anteriormente, este análisis sobre la duración y número de pausas permite aproximar cuanta carga cognitiva le está requiriendo al individuo para poder mantener la coherencia de su discurso, que luego se ponderará junto a la velocidad, pues estos dos parámetros están fuertemente relacionados en la detección de mentiras o engaños. Asimismo, para mostrar la detección de pausas se ha generado la Figura 5.2d

Prosiguiendo con los resultados obtenidos en la extracción de características de audio, se van a exponer los resultados expuestos en los archivos *.txt* generados como resumen de la extracción de características, la generación de estos archivos fue explicada en el apartado 4.1.3. Añadido a esto, se van a mostrar tanto el archivo generado para una persona que está siendo sincera, Figura 5.3b, como para una persona que ha engañado o mentido al investigador, Figura 5.3a, así realizar una comparación que muestre los puntos clave en los que se vea las diferencias entre un discurso y el otro. Dichos archivos se encuentran expuestos en la Figura 5.3.

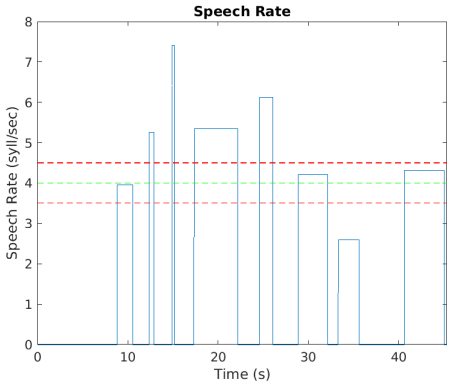
Como se puede observar, en el archivo que resumen las características extraídas de una persona que ha mentido 5.3a, se supera el máximo de *pitch* en varias ocasiones, acompañado



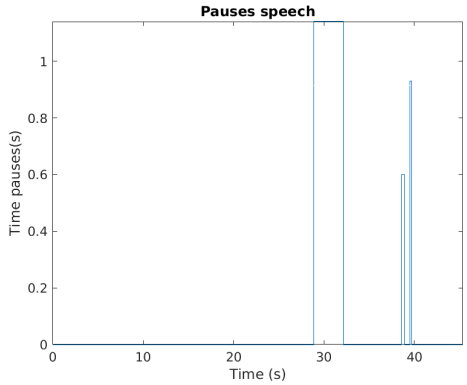
(a) Extracción de la energía de una señal de audio.



(b) Extracción del *pitch* de una señal de audio.



(c) Extracción la velocidad de una señal de audio.



(d) Extracción de las pausas generadas durante el discurso.

Figura 5.2: Extracción de características de audio.

en ocasiones de un aumento de la velocidad o una disminución, teniendo un valor fuera del rango de velocidades habitual. Además, se observa que realizan varias pausas y de una duración superior a 0.5 segundos, indicando pausas en la que ya se requiere de una carga cognitiva para elaborar un discurso coherente y poder mantener el engaño o mentira. Por otro lado, también se observa cuando ocurren los cambios anteriores, que la energía de ese instante supera a la energía media calculada para toda la onda de audio, lo cual aumenta los indicios de que la persona puede estar mintiendo. En contraposición, en la Figura 5.3b, la persona que ha sido sincera, en este caso no ha realizado pausas y apenas supera el rango máximo de *pitch*. Por otro lado, la velocidad medida para cada instante tiende a mantenerse dentro del rango de velocidades en más ocasiones que las del caso anterior. Es cierto que la energía del audio supera la media, sin embargo, esto por si solo no puede generar que haya indicios de engaño, de hecho, es la característica con menor peso a tener en cuenta y, como ya se ha mencionado con anterioridad, es la combinación de todas las características lo que dirime que se haya detectado una mentira, por eso se construyó el árbol de ponderaciones visto en el apartado 4.3 en el cual se ponderan las características en función de su relevancia para la detección de mentiras o engaños.

```
El análisis de este audio muestra lo siguiente:
Se trata de la voz de un hombre, por tanto, su frecuencia de tono natural debe estar entre los 85 y 180 Hz
Su voz tiene una energía media de:  1.133188e-04
Numero de Pausas realizadas:  3.00
La persona de este audio está mintiendo

Analisis Completo del espectro de audio:
Time    8.79    10.56   Pitch   222.3   Energy  6.82567771204334e-05   Speech_Rate     3.95
Time    12.33   12.9    Pitch   135.33  Energy  1.74308273999867e-05   Speech_Rate     5.26
Time    14.91   15.18   Pitch   152     Energy  1.05663648074683e-05   Speech_Rate     7.41
Time    17.34   22.2    Pitch   201.18  Energy  0.000137005972489076    Speech_Rate     5.35
Time    24.6    26.07   Pitch   162.55  Energy  7.54895583239549e-05    Speech_Rate     6.12
Time    28.86   32.19   Pitch   180.07  Energy  0.000233669854669435    Speech_Rate     4.2
Time    33.33   35.64   Pitch   200.74  Energy  0.000673002114447583    Speech_Rate     2.6
Time    38.58   38.94   Pitch   163.16  Energy  8.84001816611867e-05    Speech_Rate     0
Time    39.54   39.72   Pitch   162.62  Energy  5.62879913786895e-05    Speech_Rate     0
Time    40.65   45.06   Pitch   161.51  Energy  0.000416562887424685    Speech_Rate     4.31
Time    32.19   33.33   Pause_Duration  1.14
Time    38.94   39.54   Pause_Duration  0.6
Time    39.72   40.65   Pause_Duration  0.93
```

(a) Archivo resumen de características de una persona que ha engañado o mentido.

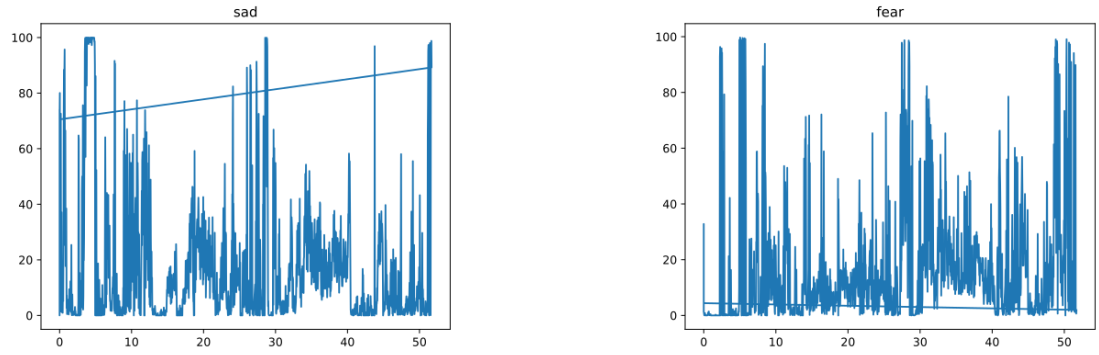
```
El análisis de este audio muestra lo siguiente:
Se trata de la voz de un hombre, por tanto, su frecuencia de tono natural debe estar entre los 85 y 180 Hz
Su voz tiene una energía media de:  3.929629e-05
Numero de Pausas realizadas:  0.00
La persona de este audio está siendo sincera

Analisis Completo del espectro de audio:
Time    10.95   11.76   Pitch   172.34  Energy  0.000192679825249842    Speech_Rate     2.47
Time    18.6    18.9    Pitch   231.85  Energy  0.000149671263265986    Speech_Rate     3.33
Time    25.35   30.42   Pitch   173.51  Energy  0.000120937405613503    Speech_Rate     3.55
Time    41.91   43.98   Pitch   179.89  Energy  0.000159785860951913    Speech_Rate     3.86
Time    45.09   49.11   Pitch   153.42  Energy  0.000223606828327708    Speech_Rate     3.98
Time    0       Pause_Duration  0
```

(b) Archivo resumen de características de una persona que ha sido sincera.

Figura 5.3: Archivos con los resúmenes de la extracción de características.

5.2. Bloque 2: Análisis de las emociones y movimientos corporales

Prosiguiendo con la muestra de resultados, se van a mostrar las gráficas generadas durante la detección de emociones de uno de los vídeos de la base de datos, con el fin de mostrar el funcionamiento del *framework* de *Deep Face*, véase la Figura 5.4. Como se comentó, en el apartado 4, el uso de las emociones servirá para añadirle robustez al sistema de clasificación de mentiras o engaño. Para ello lo que se hará es tomar los datos del *pitch*. En el caso de que en alguno de los intervalos de tiempo la media del *pitch* extraído supere el rango máximo de *pitch* asociado al género de la persona, se analizará la cantidad de *frames* del vídeo en los cuales la persona haya sentido miedo, aumentando el valor de ponderación α del clasificador. Por otro lado, si dicho valor de *pitch* es menor a la media del rango de *pitch*, se analizarán la cantidad de *frames* en los cuales la persona haya sentido tristeza. Se han analizado estas dos emociones principalmente porque se sabe que están intrínsecamente asociadas al *pitch* y dejan traslucir que la persona siente miedo y/o tristeza por ser descubierto si está mintiendo. En el caso del resto de emociones, no existe una relación tan directa con el *pitch* por ello no se han valorado en el clasificador automático.



(a) Detección de tristeza a lo largo del vídeo.

(b) Detección de miedo a lo largo del vídeo.

Figura 5.4: Detección de emociones durante el vídeo.

Como ya se ha comentado anteriormente, el uso de los movimientos corporales no se ha añadido en las ponderaciones del clasificador, debido a que para poder emplear estos datos, se precisan de otros que no se han extraído para este proyecto principalmente por su complejidad. Es por ello, que se ha determinado que se almacenen estos datos referentes al movimiento de hombros, a fin de generar un sistema visual en el cual se va a tomar el vídeo y audio del la persona investigada y se van a procesar todos los datos en conjunto para generar una demostración visual, es decir, se tomará el resultado del clasificador junto a estos movimientos para mostrar su eficacia en tiempo real. De esta forma, ejemplificar el funcionamiento del sistema generado.

5.3. Pruebas realizadas con el clasificador

Tras generar la implementación del clasificador explicado en el apartado 4.3. Se ha implementado dicho árbol de ponderaciones en *Python*, cargando tanto las características extraídas del audio como las emociones que trasluce la persona investigada. En este apartado, se van a mostrar los resultados obtenidos tras cargar dichos datos y pasarlos por el clasificador. Como se ha comentado, el clasificador se encargará de analizar cada una de las zonas en las que se ha segmentado el audio y ponderando las características de dicha zona siguiendo el árbol de decisión. Tras esto, se obtendrá un valor de α que evaluará si el audio de la persona indica que la persona ha sido sincera o ha mentido. Por otro lado, a este valor de alfa se le añadirá una ponderación extra, un valor de 0.1 extra por cada detección, en caso de que se detecten las emociones de miedo o tristeza teniendo en cuenta los detalles explicados anteriormente para esta ponderación. Tras tener el valor de α final se ha determinado que si dicho valor se encuentra por debajo de un valor de umbral de 2 se determina que la persona ha sido sincera, en caso contrario, si el valor de α supera este umbral se determinará que la persona ha mentido. Este umbral se ha determinado tras analizar las ponderaciones del árbol de decisión, por tanto cuanto menor sea el valor de las ponderaciones, más cerca de la verdad estará el individuo, y cuanto mayor sea, mostrará que la persona genera una mayor cantidad de indicios de engaño o mentira.

Para las pruebas, se han calculado tanto el porcentaje de acierto (ACC), la precisión media de engaño (DACC) y la precisión media de la honestidad (HACC), empleando las ecuaciones vistas en el apartado 4.3. Con estos datos además, se generarán varias matrices de confusión resultantes de las pruebas realizadas, acabando, con un gráfico que mostrará el espacio ROC (Receiver operating characteristic curve). Este gráfico ilustra la capacidad de diagnóstico de un sistema clasificador binario al variar su umbral de discriminación, en este caso, al variar las ponderaciones del árbol. Este gráfico se crea mostrando el ratio de verdaderos positivos (TPR) frente al ratio de falsos positivos (FPR), con varias combinaciones de ponderaciones. Las ecuaciones para calcular estos ratios son las siguientes:

$$TPR = \frac{TP}{TP + FN} \tag{5.1}$$

$$FPR = \frac{FP}{FP + TN} \tag{5.2}$$

5.3.1. Modelo 1: Empleo del árbol de decisión original con solo audio.

Tras evaluar el árbol de decisión original y empleando solo las ponderaciones realizadas para el audio, se obtienen los resultados de la Tabla 5.1 y la Matriz de confusión 5.2

ACC	DACC	HACC	TPR	FPR
52.00%	47.06%	62.5%	0.62	0.64

Tabla 5.1: Resultados del Modelo 1.

		Predicción	
		Mentira	Verdad
Real	Mentira	8	3
	Verdad	9	5

Tabla 5.2: Matriz de confusión Modelo 1.

5.3.2. Modelo 2: Modificación del árbol de decisión y empleando solo audio

Tras tomar en cuenta el árbol de decisión generado, hay un detalle que no se ha tenido en cuenta. Una de las claves para saber si existe indicios de engaño o mentira, es la duración de las pausas, pues estas reflejan la carga cognitiva que le está requiriendo al individuo a mantener la mentira. Es por ello que en la ponderación debería tomarse en cuenta la duración de las pausas, de forma que a mayor duración de la pausa, mayor será la ponderación a sumar al valor de α. Añadido a esto, para que la ponderación extra no empeore la detección y se mantenga dentro de unos varemos razonables, se ha decidido que el factor a añadir al valor de α sea proporcional a la pausa que se esté valorando en ese instante entre la pausa de

mayor duración. Así, el valor máximo a añadir será de 1 en el peor de los casos, además, este valor solo se sumará en los caso en que la pausa a evaluar sea superior a 0.5 segundos y que exista una velocidad menor al mínimo del rango de velocidad, que es cuando se presenta un aumento de la carga cognitiva. La modificación del árbol de decisiones se encuentra en la Figura A.2.

ACC	DACC	HACC	TPR	FPR
68.00%	64.71%	62.5%	0.69	0.55

Tabla 5.3: Resultados del Modelo 2.

		Predicción	
		Mentira	Verdad
Real	Mentira	9	0
	Verdad	8	8

Tabla 5.4: Matriz de confusión Modelo 2.

Como se refleja en el porcentaje de precisión (ACC) y en el valor de precisión media del engaño (DACC), se observa que este cambio aumenta considerablemente la detección de los casos en los que existe mentira.

5.3.3. Modelo 3: Empleo del árbol de decisión modificado con audio y emociones

En la prueba anterior, se mostró como el añadir a la ponderación un factor en función de la duración de las pausas, aumentaba considerablemente el porcentaje de precisión. Este porcentaje ya es bastante bueno e indica que el sistema tiene una buena clasificación. En la prueba actual se va a añadir la detección de emociones que se ha comentado anteriormente, en la cual se ha tenido en cuenta los casos en los que exista la emoción del miedo o tristeza. En este caso, los resultados se presentan en la Figura 5.5 y la matriz de confusión asociada en la Figura 5.6

ACC	DACC	HACC	TPR	FPR
80.00%	94.12%	50%	0.8	0.2

Tabla 5.5: Resultados del Modelo 3.

Como se puede observar, añadir estas ponderaciones asociadas a las emociones aumentan el porcentaje de precisión del sistema.

		Predicción	
		Mentira	Verdad
Real	Mentira	13	3
	Verdad	4	5

Tabla 5.6: Matriz de confusión del Modelo 3.

5.3.4. Modelo 4: Empleo del árbol de decisión modificado con audio y emociones cambiando la ponderación del árbol

Tras realizar las pruebas anteriores, se observó que las ponderaciones del árbol de decisión se podían ajustar aumentando la ponderación de los casos en los que no hay pausas y en la que la velocidad de ese intervalo de tiempo se encuentra fuera de los límites del rango. De esta manera, se generó otro árbol de decisión partiendo de esta consideración. Este último árbol se encuentra en la Figura A.3, asimismo los resultados de este modelo se encuentran en la Figura 5.7 y la matriz de confusión resultante en la Figura 5.8.

ACC	DACC	HACC	TPR	FPR
84.00%	100.0%	50%	0.81	0

Tabla 5.7: Resultados del Modelo 4.

		Predicción	
		Mentira	Verdad
Real	Mentira	17	4
	Verdad	0	4

Tabla 5.8: Matriz de confusión del Modelo 4.

En este último caso, el sistema ha sido capaz de determinar correctamente todos los casos en los que existe mentira o engaño, obteniéndose así el mejor porcentaje de acierto para los datos empleados.

5.4. Receiver Operating Characteristic (ROC) space

Prosiguiendo con los resultados, se va a generar espacio ROC (Receiver Operating Characteristic o Característica Operativa del Receptor) para evaluar los modelos propuestos de las pruebas anteriores, con el fin de mostrar cuál es el mejor modelo. El espacio ROC es una representación gráfica del ratio de verdaderos positivos (TPR) frente al ratio de falsos positivos (FPR) para evaluar los sistemas clasificadores binarios, como es el caso de este proyecto en el que el clasificador discrimina entre casos de mentira y de sinceridad. Partiendo de su definición, el espacio ROC para las pruebas realizadas se muestra el la Figura 5.5. Cabe mencionar, que en este caso se representa en la gráfica los modelos mediante un punto en el espacio en el espacio ROC y no la curva variación de cada modelo al cambiar el umbral de decisión.

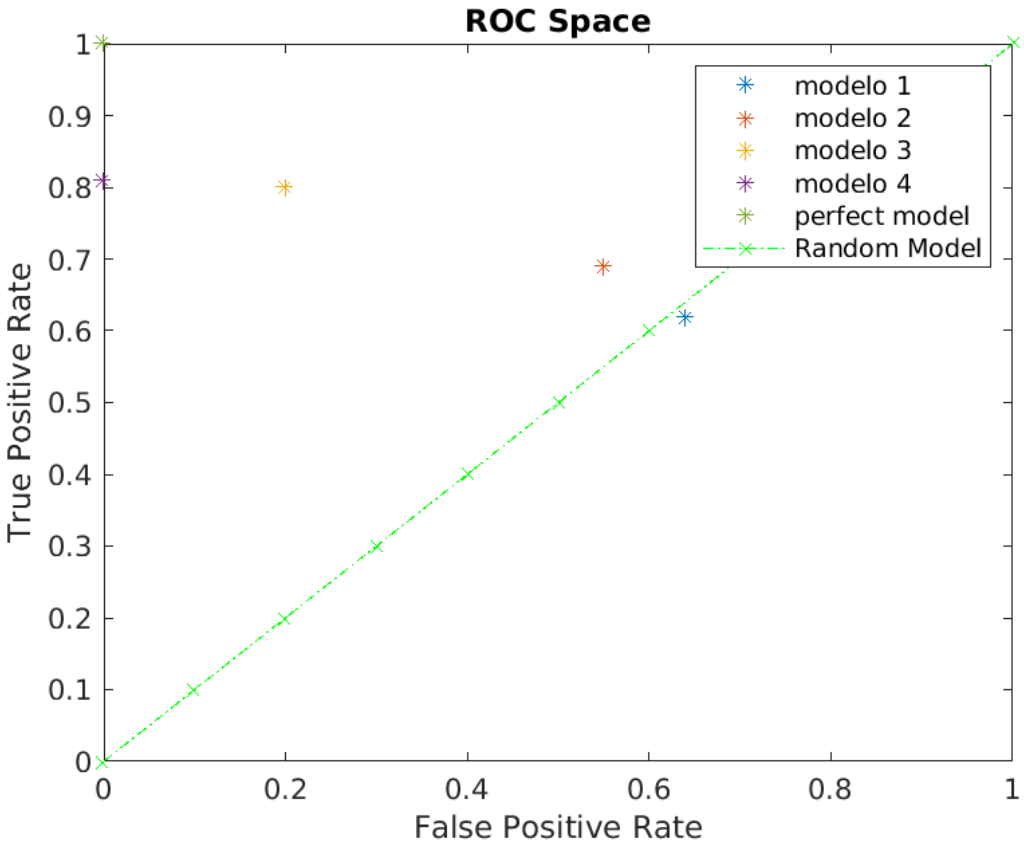


Figura 5.5: Modificación del árbol de decisión añadiendo el factor de pausas.

5.5. Aplicación del sistema de detección.

Para concluir con este apartado, se va a presentar un vídeo que recoge la ejecución del sistema de detección de mentiras. Para realización de esta aplicación, se ha empleado el mejor modelo obtenido durante las pruebas junto a varios de los vídeos de la base de datos. Para esta demostración, se ha empleado uno de los vídeos en los que claramente hay indicios de mentira y, por otro lado, un vídeo en el que haya pocos indicios de mentira o sean casi imperceptibles por los humanos, de esta forma, observar como responde el sistema ante estas dos situaciones.

El funcionamiento se basa en que en caso de que el modelo detecte mentira, se almacenarán los tiempos en los que se haya hecho dicha detección, incluyendo para ello, los tiempos en los que se detecta los movimientos de hombros descritos en apartados anteriores debido a que cuando una persona está mintiendo, aumenta el número de movimientos de hombros, sobretodo, en los momentos en los que su lenguaje corporal contradice lo que su voz expresa. Una vez se han obtenido los tiempos en los que se producen las mentiras, se emplea (*OpenCV: OpenCV modules*, s.f.) junto a *Pygame* (*Pygame*, s.f.)[1] para reproducir el vídeo junto al audio del mismo y durante la ejecución, si el tiempo del *frame* que se está mostrando coincide con el tiempo de alguna de las detecciones, se mostrará en el vídeo un mensaje de alarma, así cualquier persona que esté empleando este sistema pueda analizar visualmente los momentos en los que se ha detectado una mentira o engaño. El vídeo demostración con la ejecución del sistema se encuentra en el siguiente enlace:

`https://youtu.be/NqE1uMfK7sQ`

[1] *Pygame* es una librería de *Python* para el desarrollo de videojuegos, en esta demostración se han empleado las funciones que permiten reproducir archivos de audio

6. Conclusiones

En el estado en el que se encuentra actualmente el proyecto se han logrado afrontar los objetivos propuestos al inicio, desarrollando un sistema que se iba adaptando y mejorando para hacerlo más robusto con el conjunto de datos del que se partía. Datos que se generaron con el fin de dar un enfoque de partida al proyecto y poder analizar los puntos claves en los que se manifiestan las mentiras. Además, este sistema se ha ido confeccionando a medida que iban surgiendo problemas asociados a la complejidad del desarrollo de un sistema que sea capaz de detectar mentiras, en gran parte por la dificultad que supone entender la psicología y el funcionamiento de la inteligencia humana, y poder desgranar los puntos clave de manifestación de las mentiras, que dependen y derivan de muchas variables, como pueden ser la cultura o la educación. Sin embargo, se logró encontrar un punto medio en el que confluyen características que se manifiestan a modo general en cada ser humano y que permitían realizar un análisis para obtener unos resultados. Cabe destacar que también este proyecto ha llevado a muchos puntos de inflexión, pues se preveían inicialmente unos resultados distintos a los obtenidos.

En cuanto a su uso, pienso que podría ser útil en casos en los que se necesite de alguna herramienta que ayude a analizar si el discurso de una persona es sincero o no, por ejemplo en los casos en los que no se tengan pruebas suficientes para incriminar a una persona por un delito pero se tenga sospechas de que miente. No obstante, teniendo en mente los resultados obtenidos, se observa que el sistema cataloga correctamente todos los casos en los que existe mentira o engaño, sin embargo, solo en el 50% de los casos es capaz de catalogar correctamente si una persona ha sido completamente sincera, por tanto, el sistema puede incurrir en catalogar a una persona de culpable de mentir siendo inocente, pues el inocente puede llegar a manifestar indicios de que miente y, sin embargo, no estar haciéndolo, lo cual aumenta la dificultad del desarrollo de un sistema completamente robusto. Por ello, en trabajos futuros, se buscaría mejorar el sistema de forma que sea capaz de valorar correctamente estos casos sin perder la eficiencia del sistema actual.

Personalmente, la realización de este trabajo ha sido satisfactoria, pues aún teniendo presente la dificultad que suponía, se han logrado cumplir los objetivos propuestos. Además, porque siempre he intentado descubrir como funciona la psicología y la inteligencia humana en distintos campos, en este caso concreto, siempre me había preguntado si se podría crear un sistema que detectase mentiras mediante audio y emociones, dando un enfoque distinto al que se tiene sobre la detección de mentiras, pues siempre que se trata este tema se suele recurrir al polígrafo. Añadido a esto, yo tenía nulos conocimientos sobre tratamiento de señales de audio y reconocimiento de emociones, así que este trabajo me ha permitido adquirir nuevos conocimientos que probablemente me sirvan en un futuro. Este proyecto, también me ha dado la oportunidad de introducirme en el mundo de la investigación en el cual me gustaría formar parte en un futuro desarrollando proyectos de índole similar a este.

Por último, mencionar que se está trabajando en la redacción de un artículo para posteriormente publicarlo en la revista científica *Expert Systems with Applications* con los resultados

del trabajo, a fin de que cualquier persona que quiera replicar o mejorar el proyecto tenga una guía a seguir.

Bibliografía

Appelgren, M. (s.f.). Detecting Deception in Conversational Speech. , 43.

April 06, S. W. U., y 2020. (2015, agosto). *What Is the Frequency Range of Human Speech?* Descargado 2022-04-28, de https://www.reference.com/science/frequency-range-human-speech-3edae27f8c397c65

Audio Toolbox. (s.f.). Descargado 2022-05-21, de https://es.mathworks.com/products/audio.html

Avola, D., Cinque, L., De Marsico, M., Fagioli, A., y Foresti, G. L. (2020, octubre). LieToMe: Preliminary study on hand gestures for deception detection via Fisher-LSTM. *Pattern Recognition Letters*, *138*, 455–461. Descargado 2021-05-06, de https://www.sciencedirect.com/science/article/pii/S0167865520303123 doi: 10.1016/j.patrec.2020.08.014

Cronología del 'Caso Bretón'. (s.f.). Descargado 2022-05-28, de https://www.antena3.com/noticias/sociedad/cronologia-caso-breton_2013071257521f6b6584a8ec2159c0d7.html

Do Lie Detector Tests Really Work? | Psychology Today. (s.f.). Descargado 2022-05-28, de https://www.psychologytoday.com/us/blog/the-nature-deception/202001/do-lie-detector-tests-really-work

Ekman, P. (s.f.). Como detectar mentiras. , 188.

Ekman, P., Friesen, W. V., y Scherer, K. R. (1976). Body movement and voice pitch in deceptive interaction. *Semiotica*, *16*(1), 23–27. (Place: Netherlands Publisher: Mouton and Company) doi: 10.1515/semi.1976.16.1.23

Gallardo-Antolín, A., y Montero, J. M. (2021, julio). Detecting Deception from Gaze and Speech Using a Multimodal Attention LSTM-Based Framework. *Applied Sciences*, *11*(14), 6393. Descargado 2022-05-28, de https://www.mdpi.com/2076-3417/11/14/6393 doi: 10.3390/app11146393

Gupta, V., Agarwal, M., Arora, M., Chakraborty, T., Singh, R., y Vatsa, M. (2019). Bag-Of-Lies: A Multimodal Dataset for Deception Detection. En (pp. 0–0). Descargado 2021-05-06, de https://openaccess.thecvf.com/content_CVPRW_2019/html/CV-COPS/Gupta_Bag-Of-Lies_A_Multimodal_Dataset_for_Deception_Detection_CVPRW_2019_paper.html

Khan, W., Crockett, K., O'Shea, J., Hussain, A., y Khan, B. M. (2021, mayo). Deception in the eyes of deceiver: A computer vision and machine learning based automated deception detection. *Expert Systems with Applications*, *169*, 114341. Descargado 2021-05-06, de

https://www.sciencedirect.com/science/article/pii/S0957417420310289 doi: 10.1016/j.eswa.2020.114341

Mari Luz Cortés, el llanto que se hizo multitud y reescribió el Código Penal | España. (s.f.). Descargado 2022-05-28, de https://www.elmundo.es/espana/2019/08/11/5d4ebcbcfdddff46b68b4694.html

MATLAB. (s.f.). Descargado 2022-05-21, de https://es.mathworks.com/products/matlab.html

MediaPipe. (s.f.). Descargado 2022-05-22, de https://mediapipe.dev/

NumPy. (s.f.). Descargado 2022-05-23, de https://numpy.org/

OpenCV: Face Detection using Haar Cascades. (s.f.). Descargado 2022-05-22, de https://docs.opencv.org/3.4/d2/d99/tutorial_js_face_detection.html

OpenCV: OpenCV modules. (s.f.). Descargado 2022-05-22, de https://docs.opencv.org/4.x/

Platón. (2009). *La República.* Ediciones AKAL. (Google-Books-ID: aAk3O462g1QC)

Pose. (s.f.). Descargado 2022-05-22, de https://google.github.io/mediapipe/solutions/pose.html

Pygame. (s.f.). Descargado 2022-05-31, de https://www.pygame.org/wiki/about

SciPy. (s.f.). Descargado 2022-05-23, de https://scipy.org/

Serengil, S. I. (s.f.). *deepface: A Lightweight Face Recognition and Facial Attribute Analysis Framework (Age, Gender, Emotion, Race) for Python.* Descargado 2022-05-22, de https://github.com/serengil/deepface

The SpeechMark MATLAB Toolbox. (2015, octubre). Descargado 2022-04-19, de https://speechmrk.com/speechmark-products-downloads/the-speechmark-matlab-toolbox/

Speech tempo. (2022, marzo). Descargado 2022-04-19, de https://en.wikipedia.org/w/index.php?title=Speech_tempo&oldid=1076963169 (Page Version ID: 1076963169)

TensorFlow. (s.f.). Descargado 2022-06-01, de https://www.tensorflow.org/?hl=es-419

VOICEBOX. (s.f.). Descargado 2022-04-19, de http://www.ee.ic.ac.uk/hp/staff/dmb/voicebox/voicebox.html#analysis

Vrij, A., Fisher, R. P., y Blank, H. (2017, febrero). A cognitive approach to lie detection: A meta-analysis. *Leg Crim Psychol*, *22*(1), 1–21. Descargado 2021-04-23, de http://doi.wiley.com/10.1111/lcrp.12088 doi: 10.1111/lcrp.12088

Álava, M. J. (2016). *La verdad de la mentira: Claves para descubrir el daño emocional y los secretos de las mentiras propias y ajenas.* La Esfera de los Libros. (Google-Books-ID: ZqYqDQAAQBAJ)

A. Anexo I

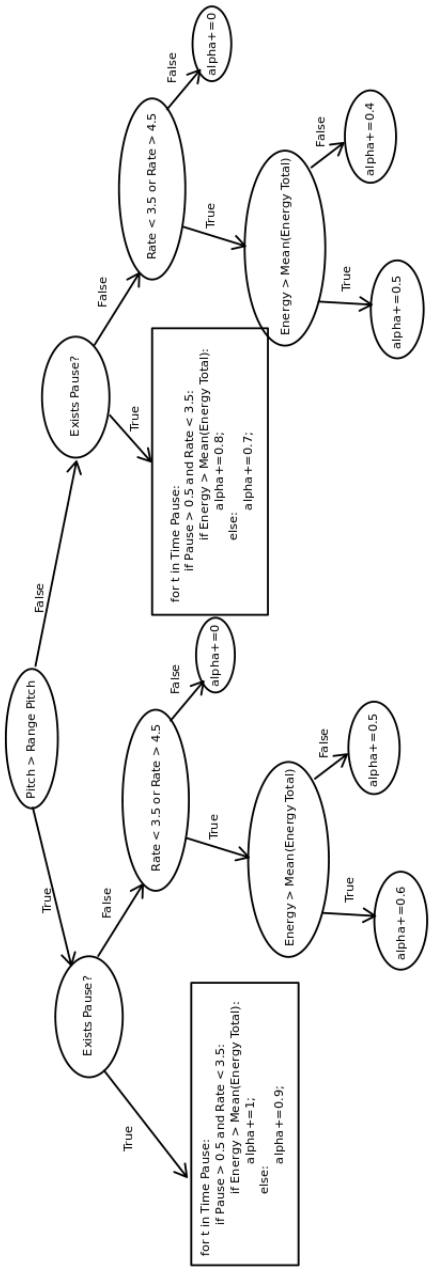


Figura A.1: Árbol de decisión para ponderar los datos de audio.

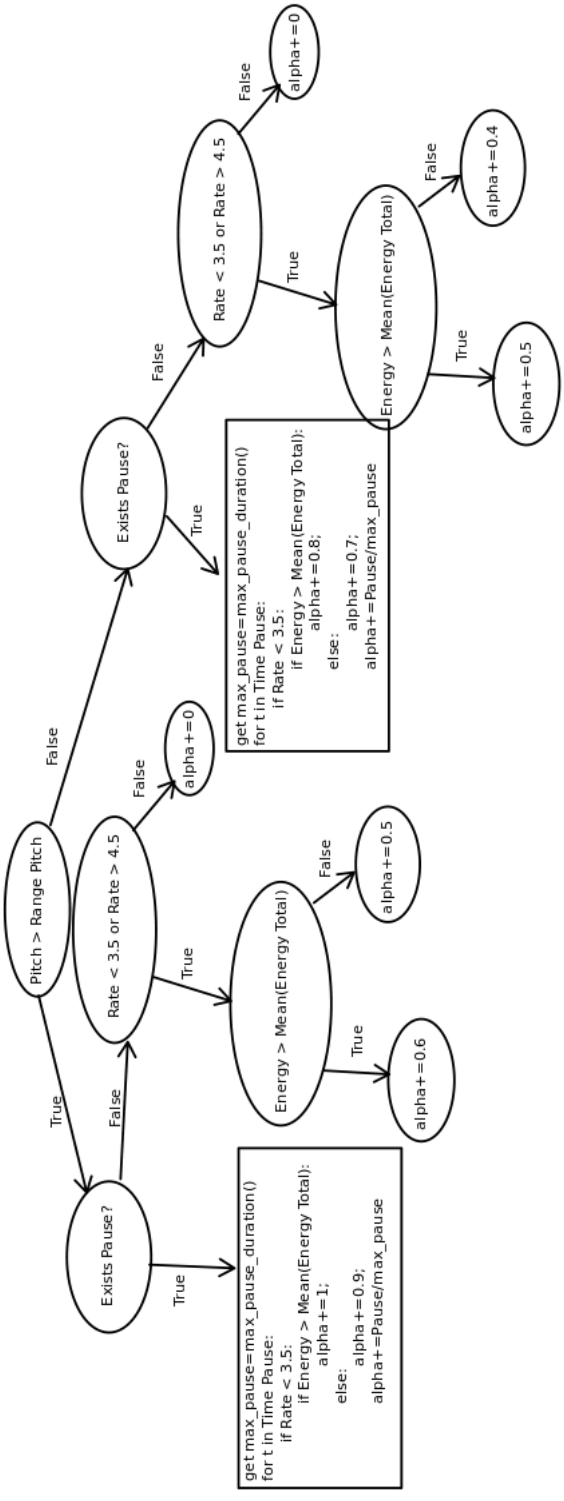


Figura A.2: Modificación del árbol de decisión añadiendo el factor de pausas.

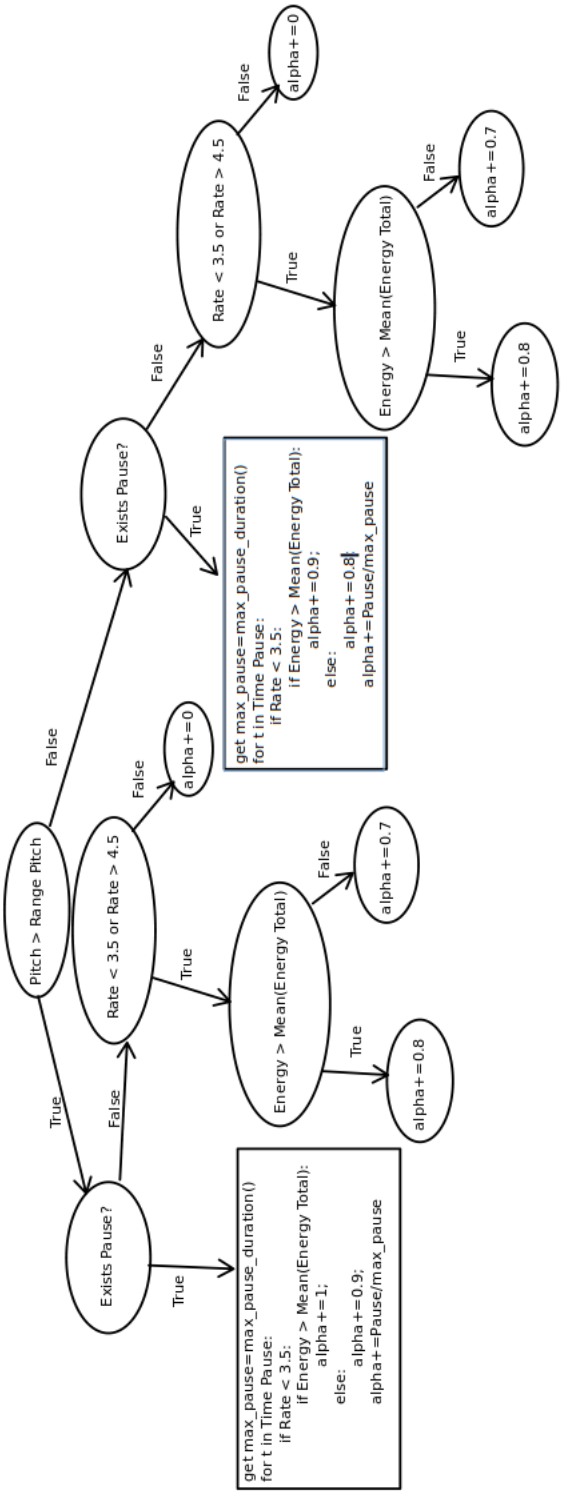


Figura A.3: Modificación del árbol de decisión añadiendo el factor de pausas.