

UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ingeniería



**Sistema interactivo de reconocimiento de comandos por voz
con análisis de emociones para robot animatrónico**

Trabajo de graduación presentado por Blanca María Belén Hernández
Batres para optar al grado académico de Licenciado en Ingeniería
Electrónica

Guatemala,

2018

UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ingeniería



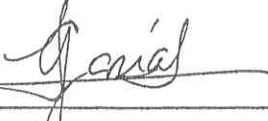
**Sistema interactivo de reconocimiento de comandos por voz
con análisis de emociones para robot animatrónico**

Trabajo de graduación presentado por Blanca María Belén Hernández
Batres para optar al grado académico de Licenciado en Ingeniería
Electrónica

Guatemala,

2018

Vo.Bo.:

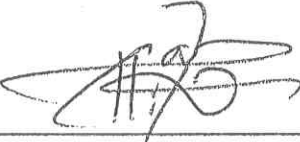
(f) 

Ing. Lynette García Pérez

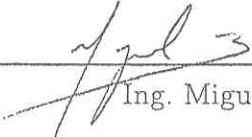
Tribunal Examinador:

(f) 

Ing. Lynette García Pérez

(f) 

MSc. Carlos Esquit

(f) 

Ing. Miguel Zea

Fecha de aprobación: Guatemala, 5. de diciembre de 2018.

Un animatrónico es la tecnología que resulta de animar personajes como marionetas u otras figuras mediante la electrónica con el fin de brindar entretenimiento o realizar tareas similares al comportamiento humano. Es por ello que las marionetas o muñecos deben guardar cierto parecido con los humanos o bien, con alguna otra especie. Puesto que construir un animatrónico fusiona varias disciplinas de la ingeniería como lo son la mecánica, el control de potencia, el procesamiento de señales, la programación e inteligencia artificial, entre otras, se considera un proyecto integral como trabajo de graduación. Este será realizado con un equipo de tres estudiantes de ingeniería mecatrónica y tres de ingeniería electrónica divididos de la siguiente manera: los estudiantes de mecatrónica se encargarán del diseño y ensamble mecánico (pueden encontrarse sus trabajos en [1], [2] y [3]), dos de los de electrónica se encargarán de la interacción con el usuario (El presente trabajo y [4]) y, el último, de la alimentación y movimiento de motores [5]. Debido a que originalmente el proyecto fue pensado para implementarse en uno de los parques de diversiones del Instituto de Recreación de los Trabajadores de la Empresa Privada de Guatemala (IRTRA), se eligió la figura de un pirata para ser animado, con el fin de colocarlo en la entrada de alguna atracción. Dicho pirata contará con varias funciones de comportamiento e interacción humana, más no será capaz de caminar sino se encontrará fijo en un punto. Por lo tanto, se espera que la movilidad del torso hacia arriba (incluido el rostro) sea lo más natural posible. El pirata deberá ser capaz de comunicarse efectivamente con los asistentes al parque, esto mediante control por voz, reconocimiento de gestos y su capacidad para conversar.

Específicamente, con respecto a este módulo, el de generación y reconocimiento de voz, se espera que el animatrónico sea capaz de reconocer diversos comandos por voz y responder a estos, ya sea mediante acciones o palabras. Así, este será capaz de mantener conversaciones breves con las personas y de realizar gestos y movimientos para complementar su comunicación. Además, con el fin de lograr una mejor interacción con el usuario se evaluará la posibilidad de reconocer emociones en la voz. De lograrse, el animatrónico será capaz de interpretar de una manera más exacta a los humanos.

Personalmente, considero que un animatrónico pone en práctica un sinnúmero de conceptos trabajados a lo largo de la carrera, por lo que se trata de una buena oportunidad para poner en práctica todo lo aprendido. Además, a pesar del nivel de complejidad del proyecto, las herramientas para trabajarlo no son sumamente sofisticadas y se encuentran al alcance. Sumado a esto, al menos en mi opinión y haciendo énfasis en el reconocimiento de voz con emociones, se trata de un proyecto con gran atractivo, por lo que realizarlo no será una tarea agobiante.

A lo largo de la elaboración de este trabajo muchas personas me apoyaron tanto en lo profesional como en lo personal. En este apartado quisiera mostrarles mi agradecimiento, no sin antes agradecer a Dios por haberme iluminado en todo momento y porque en Su plan estaba que yo estuviera en este escenario.

A mi familia por la motivación diaria y palabras de aliento. En especial a mis papás por todo el apoyo de diferentes tipos para permitirme trabajar en este proyecto y terminarlo. Sin ellos no estaría hoy aquí concluyendo este trabajo y a punto de finalizar la carrera de Ingeniería Electrónica.

A mi asesora, la Ingeniera Lynette García, por aceptar asesorarme en este trabajo a pesar de no contar con tanta disponibilidad de tiempo; por demostrarme su gran voluntad de ayudar y darme tranquilidad en cada reunión. Por aconsejarme al definir mis objetivos y proponer las formas adecuadas de alcanzarlos. Por orientarme en el mundo de la inteligencia artificial, en especial en el ámbito de aprendizaje de máquinas. Por mostrarme nuevas formas de abordar un problema para buscarle una solución. Por asistirme en la validación de los parámetros relevantes en la detección de emociones en la voz así como para encontrar relaciones entre estos. Por tomarse el tiempo de leer exhaustivamente este trabajo y proporcionarme una revisión integral del mismo: tanto en contenido como en edición. Por toda la paciencia y la orientación.

Al catedrático del curso, el Ingeniero Luis Pedro Montenegro, por ayudarme a estructurar este trabajo, asesorarme al plantear mis objetivos y guiarme para lograr cumplirlos. También por el tiempo dedicado a resolver mis dudas, así como su gran contribución con sugerencias y maneras de mejorar el trabajo. Por el apoyo semanal en la revisión de avances y críticas de ser necesario. Fue de gran apoyo tanto en la parte práctica como al momento de redactar y revisar este trabajo.

A mis compañeros por su disposición a ser sujetos de prueba, las risas, sugerencias, ocurrencias y sobretodo por la camaradería. A Arturo García, Luis Ruano, Betti Rodas, Manolo Benegas y Francisco Zelada, quienes también trabajaron en el área de Animatrónica, por vivir junto conmigo la experiencia, apoyarnos en cada entrega y por el buen trabajo en equipo. Específicamente a Arturo, por la cooperación con la sección de Codificación y Transmisión de generadores del Movimiento, así como por las oportunidades en que trabajamos juntos. A Juan Diego Benítez, Daniela Pocasangre y Gabriel Martínez por su apoyo en la implementación de la transformada de Fourier en Python. A Marcos Benedict, por acompañarme a lo largo de todo el proceso, compartir conmigo a diario en los laboratorios al trabajar, escuchar todas mis quejas y distraerme cuando era necesario. Todos forman parte importante de este proyecto.

Al Departamento de Tecnologías Interactivas por posibilitar la adquisición de muestras de audio de calidad. En especial a Elba Mazariegos, por estar pendiente en los momentos en que realizaba las pruebas y brindarme su ayuda con el equipo. También a Óscar Velasquez, por proporcionarme toda la información necesaria sobre el uso del Aula Virtual.

Al Departamento de Ingeniería Mecatrónica y Electrónica. Al director y secretaria del departamento, el Ingeniero Carlos Esquit y Lésly Gómez, por sugerir el uso del Aula Virtual para la elaboración de mis pruebas de audio y reservarla para mi uso. A los catedráticos José Penagos, Miguel Zea y Pablo Oliva, por asistirme en los inicios y encauzar este trabajo de graduación. A Estuardo y Willy por proporcionar siempre las herramientas y equipo necesarios y su apoyo los días que utilicé los laboratorios en horarios no establecidos o cuando necesitaba permanecer hasta tarde.

Al Departamento de Ciencias de la Computación. A Hector Hurtarte por su asesoría con programación en Python y manipulación de archivos L16. A Dogulas Barrios por su asesoría en la utilización del software R.

Al profesor Eugenio Aristondo, por su asesoría al momento de seleccionar las herramientas estadísticas adecuadas.

A mis amigos, a mis catedráticos, a cada una de las personas que se prestaron para las pruebas de audio y a todos aquellos que de una u otra forma fueron parte importante de este proceso. Mil gracias.

Prefacio	VII
Lista de figuras	XIV
Lista de cuadros	XVI
Resumen	XVIII
Abstract	XX
1. Introducción	1
2. Antecedentes	3
2.1. Reconocimiento de voz y generación del habla	3
2.2. Emulación de conversación humana	3
2.3. Análisis de emociones en la voz	4
3. Justificación	9
4. Objetivos	11
4.1. Objetivo general	11
4.2. Objetivos específicos	11
5. Alcance	13
6. Marco teórico	15
6.1. Voz a texto (<i>Speech to text</i> - STT)	15
6.2. Texto a voz (<i>Text to speech</i> -TTS)	15
6.3. Cualidades del sonido	16
6.3.1. Altura o tono (Frecuencia)	16
6.3.2. Intensidad (Amplitud)	17
6.3.3. Duración (Tiempo)	17

6.3.4.	Timbre (Armónicos)	18
6.3.5.	Espacialidad	20
6.4.	Audio digital	20
6.4.1.	Parámetros fundamentales	20
6.4.2.	Archivos WAV	20
6.4.3.	Archivo .l16	21
6.5.	Dominio de frecuencia	21
6.5.1.	Transformada Rápida de Fourier - FFT	22
6.5.2.	Filtro Butterworth para la voz	23
6.6.	Aprendizaje de máquina <i>Machine Learning</i>	23
6.6.1.	Aprendizaje supervisado	24
6.6.2.	Aprendizaje no supervisado	26
6.7.	Estadística descriptiva	27
6.7.1.	Medidas de tendencia	28
6.7.2.	Medidas de dispersión	29
6.7.3.	Medidas de distribución	29
6.8.	Validación cruzada	30
6.8.1.	Tipos de validaciones cruzadas	31
7.	Lista de librerías	33
8.	Reconocimiento de voz	35
8.1.	Metodología	35
8.2.	Resultados	36
8.3.	Discusión	37
9.	Generación del habla	39
9.1.	Metodología	39
9.2.	Resultados	39
9.3.	Discusión	40
10.	Emulación de conversación humana	41
10.1.	Metodología	41
10.2.	Resultados	43
10.3.	Discusión	45
11.	Codificación y transmisión de generadores de movimiento	47
11.1.	Metodología	47
11.2.	Resultados	48
11.3.	Discusión	48
12.	Análisis de audio y patrones de emoción	49
12.1.	Metodología	49
12.1.1.	Análisis de intensidad	50
12.1.2.	Análisis de frecuencia	50
12.2.	Resultados	51
12.3.	Discusión	54

13.Codificación de la base de datos de emociones	57
13.1. Metodología	57
13.1.1. Parametrización de los archivos de audio	57
13.1.2. Construcción de la base de datos	59
13.2. Resultados	60
13.3. Discusión	60
14.Validación de la parametrización definida	61
14.1. Metodología	61
14.2. Resultados	62
14.3. Discusión	63
15.Evaluación de la probabilidad de reconocer emociones en la voz	65
15.1. Metodología	65
15.2. Resultados	66
15.3. Discusión	69
16.Pruebas de detección de emociones	73
16.1. Metodología	73
16.2. Resultados	74
16.3. Discusión	74
17.Conclusiones	77
18.Recomendaciones	79
19.Bibliografía	81
20.Anexos	85
20.1. Código en Python	85
20.2. Codificación de movimientos	94
20.3. Árbol de decisiones	94
20.4. Base de datos	96
21.Glosario	97

1.	Emociones y parámetros del discurso [11]	5
2.	Dominio del tiempo en comparación con el Dominio de frecuencia [22]	22
3.	Estructura de un árbol de decisiones [26]	25
4.	Ejemplo máquinas de vectores [26]	26
5.	Tipos de oblicuidad o simetría [30]	30
6.	Tipos de Curtosis [30]	30
7.	Validación cruzada de K iteraciones con K=4 [31]	31
8.	Validación cruzada aleatoria con K iteraciones [31]	32
9.	Validación cruzada dejando uno fuera (LOOCV)[31]	32
10.	Salida en consola para la prueba de STT	36
11.	Umbral de sensibilidad definido para la prueba de STT	36
12.	Excepción de audio lanzada	36
13.	Excepción de conexión lanzada	36
14.	Excepción de tiempo lanzada	36
15.	Información de las voces disponibles con pyttsx	39
16.	Introducción al programa de conversación	43
17.	Primera parte de una conversación con el chatbot	43
18.	Segunda parte de una conversación con el chatbot	44
19.	Chatbot iniciando la conversación con el usuario	44
20.	Comportamiento del programa cuando la entrada de audio no pudo ser descifrada	45
21.	Prueba de intensidad sujeto 1	51
22.	Prueba de intensidad sujeto 2	52
23.	Prueba de intensidad sujeto 3	52
24.	Prueba de intensidad sujeto 4	53
25.	Amplitud y frecuencia nota Si	53
26.	Amplitud y frecuencia nota Do	54
27.	Amplitud y frecuencia nota Sol	54

28.	Frecuencias de las notas musicales	56
29.	Fragmento de la base de datos codificada	60
30.	Fragmento del árbol de decisiones realizado con 24 parámetros.	62
31.	Código para el reconocimiento de voz	85
32.	Código para la generación del habla	86
33.	Código para la emulación de conversación humana	87
34.	Código para las pruebas de intensidad	88
35.	Código para las pruebas de frecuencia	89
36.	Código para la creación de la base de datos parametrizada	90
37.	Código para la creación del árbol de decisiones	91
38.	Código para la evaluación de la posibilidad de detectar emociones en la voz	92
39.	Código para las pruebas de detección de emociones	93
40.	Fragmento del código generado automáticamente para visualizar el árbol de decisiones en Webgraphviz	94
41.	Porción 1 del árbol generado por Webgraphviz	95
42.	Porción 2 del árbol generado por Webgraphviz	95
43.	Porción 3 del árbol generado por Webgraphviz	96
44.	Base de datos de emociones encontrada	96
45.	Formulario para solicitud de la base de datos	96

1.	Desencadenantes de un movimiento con su respectiva bandera	48
2.	Ocurrencias de los 24 parámetros evaluados con el árbol de decisiones.	62
3.	Comparación de algoritmos a seis emociones	66
4.	Probabilidad de acierto a seis emociones del algoritmo con el mejor resultado	66
5.	Matriz de confusión entre las seis emociones, utilizando el algoritmo LDA	66
6.	Reporte de clasificación para seis emociones, utilizando el algoritmo LDA	67
7.	Comparación de algoritmos a tres emociones: enojo, sorpresa y tristeza	67
8.	Probabilidad de acierto a tres emociones (enojo, sorpresa y tristeza) del algoritmo con el mejor resultado	67
9.	Matriz de confusión entre enojo, sorpresa y tristeza, utilizando el algoritmo LDA	67
10.	Reporte de clasificación para tres emociones: enojo, sorpresa y tristeza, utilizando el algoritmo LDA	67
11.	Comparación de algoritmos a tres emociones: alegría, miedo y tristeza	68
12.	Probabilidad de acierto a tres emociones (alegría, miedo y tristeza) del algoritmo con el mejor resultado	68
13.	Matriz de confusión entre alegría, miedo y tristeza, utilizando el algoritmo LDA	68
14.	Reporte de clasificación para tres emociones: alegría, miedo y tristeza, utilizando el algoritmo LDA	68
15.	Comparación de algoritmos a tres emociones: alegría, enojo y tristeza	68
16.	Probabilidad de acierto a tres emociones (alegría, enojo y tristeza) del algoritmo con el mejor resultado	69
17.	Matriz de confusión entre alegría, enojo y tristeza, utilizando el algoritmo LDA	69
18.	Reporte de clasificación para tres emociones: alegría, enojo y tristeza, utilizando el algoritmo LDA	69

19.	Porcentajes de acierto obtenidos por cada uno de los seis algoritmos supervisados, por cada set de tres emociones	74
20.	Interpretación por parte del módulo de <i>Control del movimiento de motores</i> de las banderas de movimiento codificadas	94

El objetivo del presente trabajo es implementar un sistema de reconocimiento de comandos por voz interactivo, que sea capaz de responder a comandos básicos y mantener conversaciones breves con el usuario, así como de accionar movimientos del animatrónico donde se utilice. Sumado a esto, se busca llevar a cabo un estudio de los cambios en la voz debido a los diferentes estados anímicos, para comprobar si es posible o no determinar emociones con base en la voz mediante software. Este trabajo comprende disciplinas de programación de alto nivel, aprendizaje de máquina y procesamiento de señales.

Para la creación del sistema de conversación, se implementaron las tecnologías de Speech to Text STT y Text to Speech TTS así como un poco de aprendizaje de máquina para la generación de respuestas en una conversación. También se utilizó comunicación serial para indicar al módulo de control de motores algunos movimientos acordes al diálogo actual. Se obtuvo un sistema capaz de recibir entradas por voz y responder también por voz. Este cuenta con varias bases de datos para conversaciones, una para movimientos y una para comandos específicos del lugar donde se implemente (por ejemplo, un parque de diversiones). Es también capaz de iniciar la interacción con el usuario por su cuenta, así como de aprender de las conversaciones que realice e indicar si no fue capaz de comprender una entrada de voz. Un aspecto en el que este programa podría mejorarse es en la velocidad de respuesta, lo que podría ser posible implementando un reconocimiento de audio en tiempo real.

Para la evaluación de detección de emociones en la voz, se implementaron seis algoritmos diferentes de aprendizaje de máquina supervisado y se analizó audio básicamente en intensidad, frecuencia y duración. Los algoritmos que presentaron mejores resultados fueron el Análisis Discriminante Lineal (LDA) y el Árbol de Decisiones (CART). Se determinó que de los 24 parámetros encontrados como relevantes en la detección de emociones utilizados, los más relevantes son dos: frecuencia con la mayor amplitud y la oblicuidad del espectro frecuencial. Otros dos parámetros casi tan relevantes como los anteriores fueron: la oblicuidad y la curtosis de la intensidad. Trabajando con las seis emociones y realizando pruebas con muestras conocidas por el

algoritmo, se obtuvo un porcentaje de acierto máximo del 69%. Se determinó tras varias pruebas que las emociones más fáciles de detectar son enojo, alegría y tristeza. Al disminuir el estudio a estas tres emociones, el porcentaje de acierto alcanzó hasta un 95%. Para la validación del algoritmo con muestras desconocidas por la base de datos, para las tres emociones identificadas como las más fáciles de detectar se consiguió un porcentaje de acierto máximo del 55%. Puede decirse que es posible detectar emociones en la voz si se mejoran ciertos aspectos que darían como resultado un mejor porcentaje de acierto. Entre estos: entrenar al algoritmo con una mayor cantidad de muestras, definir más parámetros relevantes en la detección de emociones y omitir las muestras de sujetos que presentaban emociones muy débiles y difíciles de identificar incluso por un ser humano.

The objective of this paper is to implement an interactive voice command recognition system, which is capable of responding to basic commands and having brief conversations with the user, as well as of activating animatronic movements where it is used. Added to this, it is sought to carry out a study of the changes in the voice due to the different moods, to verify if it is possible or not to determine emotions based on the voice through software. This work includes disciplines of high level programming, machine learning and signal processing.

For the creation of the conversation system, Speech to Text STT and Text to Speech TTS technologies were implemented as well as a little machine learning to generate responses in a conversation. Serial communication was also used to indicate to the engine control module some movements according to the current dialogue. A system capable of receiving voice inputs and answering by voice was obtained. This has several databases for conversations, one for movements and one for specific commands of the place where it is implemented (for example, an amusement park). It is also able to initiate the interaction with the user on his own, as well as to learn from the conversations he makes and indicate if he was not able to understand a voice input. One aspect in which this program could be improved is in the speed of response, which could be possible by implementing an audio recognition in real time.

For the evaluation of emotion detection in the voice, 6 different algorithms of supervised machine learning were implemented and audio was analyzed basically in intensity, frequency and duration. The algorithms that presented the best results were the Linear Discriminant Analysis (LDA) and the Decision Tree (CART). It was determined that of the 24 parameters found to be relevant in the detection of emotions used, the most relevant are two: frequency with the greatest amplitude and obliquity of the frequency spectrum. Two other parameters almost as relevant as the previous ones were: the obliquity and the kurtosis of the intensity. Working with the 6 emotions and performing tests with samples known by the algorithm, a maximum success rate of 69 % was obtained. It was determined after several tests that the easiest emotions to detect are anger, joy and sadness. By decreasing the study to these 3 emotions,

the success rate reached up to 95%. For validation of the algorithm with unknown samples from the database, for the 3 emotions identified as the easiest to detect, a maximum success rate of 55% was achieved. It can be said that it is possible to detect emotions in the voice if certain aspects are improved that would result in a better percentage of success. Between these; train the algorithm with a greater number of samples, define more relevant parameters in the detection of emotions and omit the samples of subjects who had very weak emotions and difficult to identify even by a human being.

Para llevar a cabo un programa que sea capaz de recibir comandos por voz y conversar así como un estudio de la detección de patrones emocionales en la voz, se siguieron los siguientes pasos:

1. Reconocimiento de voz
2. Generación del habla
3. Emulación de conversación humana
4. Codificación y transmisión de generadores de movimiento
5. Análisis de audio y patrones de emoción
6. Codificación de la base de datos de emociones
7. Validación de la parametrización definida
8. Evaluación de la probabilidad de reconocer emociones en la voz
9. Pruebas de detección de emociones

El reconocimiento de voz y la generación del habla son los pasos iniciales pues estos dan origen a las entradas y salidas del sistema. Posteriormente, para poder manipular dichas entradas y obtener resultados como la emulación de una conversación y el reconocimiento de patrones de emoción es necesario poseer ciertas bases en inteligencia artificial. Con el fin de complementar la experiencia de interacción por voz, el programa debe contar con una codificación de movimientos relacionados al contexto de su diálogo, la cual puede ser transmitida al módulo de control de motores mediante comunicación serial.

Por otro lado, para el reconocimiento de patrones de emoción es necesario profundizar en los temas de análisis de audio y aprendizaje de máquina. Surge así, la necesidad de codificar la base de datos de emociones con la que se cuenta para implementarla como entrenamiento de algoritmos. Así se hace posible la creación de un programa para la evaluación de la detección de patrones de emoción. Dicho programa necesita validación. Esta puede llevarse a cabo ingresando muestras ajenas a la base de datos y observando el comportamiento del algoritmo.

2.1. Reconocimiento de voz y generación del habla

Jasper es un proyecto creado por los estudiantes Shubhro Saha y Charlie Marsh. Se trata de una plataforma de procesamiento de la voz que puede estar todo el tiempo a disposición del usuario y es personalizable con los comandos que este prefiera. Este puede o no estar conectado a la internet, dependiendo de los motores STT y TTS que se elijan para configurarlo. Es capaz de controlar funciones simples de la computadora como iniciar un reproductor de música o manipular redes sociales. Funciona mediante una Raspberry Pi. [6]

2.2. Emulación de conversación humana

La librería Chatterbot [7] para python cuenta con una gran variedad de documentación con respecto a la implementación de plataformas de conversación por texto. Se tomaron varios de estos ejemplos para obtener un sistema al que pudieran incorporársele herramientas que permitieran hacer uso del mismo mediante la vía de comunicación por defecto del ser humano: la voz.

Existen también varias aplicaciones que implementan chatbots y es útil explorarlas para tener una idea de lo que va a desarrollarse. Eviebot [8], Cleverbot [9] y Mitsuku [10] son algunos ejemplos de aplicaciones conversacionales disponibles en la web. De estos, únicamente Eviebot implementa tecnologías TTS, pues este bot responde verbalmente al usuario haciéndolo parecer más real. Por otro lado, a pesar de no contar con funciones de reconocimiento de voz ni generación del habla, Mitsuku es capaz de llevar a cabo una conversación muy parecida a la humana por medio de entradas de texto. Su habilidad para mantener el contexto en una conversación re-

sulta impresionante. Este chatbot también es capaz de enviar imágenes para mejorar aún la experiencia de conversación vía chat. Cuenta también con un diseño de su aspecto personal, lo cual le otorga un mayor realismo. Ha sido cuatro veces ganador del premio "Loebner Prize Turing Test".

2.3. Análisis de emociones en la voz

En [11], [12] y [13] pueden encontrarse investigaciones previas sobre este tema, siendo el primer trabajo citado el más relevante. A continuación los aspectos más importantes encontrados.

La Real Academia Española de la lengua define emoción como la “alteración del ánimo intensa y pasajera, agradable o penosa, que va acompañada de cierta conmoción somática”. El carácter intenso y momentáneo de estas alteraciones hace difícil su detección por sistemas automáticos. Hemos de considerar además, cuáles son las emociones que estamos interesados en detectar. Un sistema diseñado para detectar alteraciones independientemente del carácter de las mismas será mucho más sencillo de implementar y aplicar que un sistema cuyo fin sea la determinación efectiva del signo de la emoción, positivo, negativo, neutro... etc. Es posible clasificar las emociones en unas pocas categorías primarias, concretamente las seis grandes emociones son: felicidad, sorpresa, miedo, disgusto, cólera, tristeza [11].

Un aspecto a considerar cuando se recogen emociones es la espontaneidad de las mismas. La mayor parte de las bases de datos disponibles para el estudio de las emociones están construidas ad hoc, esto es, los gestos o el discurso asociado a cada una de las emociones están representados por personas que, por indicación, gesticulan o hablan de acuerdo a la emoción que les ha sido indicada. Obviamente, no existe garantía de que el discurso o la gesticulación de esas personas sea la misma información que recogeríamos de modo espontáneo [11].

Existe evidencia de que un amplio rango de características lingüísticas tienen significado emocional. Estas características pueden ser tanto fonéticas (o acústicas) como sintácticas (o léxicas). Se necesita describir de modo adecuado los estados emocionales que aparecen en el discurso. Como indicamos anteriormente, uno de los problemas importantes en el reconocimiento de emociones es el modo de identificar de manera eficiente las características que definen las diferentes emociones y las relaciones entre ellas [11].

Existen ciertas variables que, por su propia definición, son específicas del discurso y no las encontraremos, o serán más difíciles de detectar en el gesto o en la representación facial del individuo. Por ejemplo, variaciones de la entonación pueden ser debidas a cambios en el estado emocional del individuo. La Figura 1 presenta un resumen de las relaciones entre las emociones y los parámetros del discurso [11].

	Ira	Felicidad	Tristeza	Miedo	Disgusto
Velocidad	Ligeramente acelerada	Acelerada o retardada	Pausada	Muy acelerada	Mucho más acelerada
Variación	Muy alta	Alta	Ligeramente baja	Muy alta	Muy baja
Rango	Amplio	Amplio	Estrecho	Amplio	Amplio
Respiración	Acompasada	Acompasada	Resonante	Irregular	Refunfunando
Intensidad	Alta	Alta	Baja	Normal	Baja
Articulación	Tensa	Normal	Pausada	Precisa	Normal
Calidad de la voz	Procedente del pecho	Estridente	Resonante	Irregular	Retumbante

Figura 1: Emociones y parámetros del discurso [11]

Existe, en general, una relación conocida entre el discurso y las emociones primarias. Las medidas del discurso que parecen ser buenas indicadores de estas emociones son medidas acústicas continuas, tales como las relacionadas con la variación del discurso, el rango, la intensidad y la duración del mismo. Sin embargo, esta relación no suele ser suficiente. Una de las líneas de investigación en el reconocimiento automático de emociones es la mejora de nuestra capacidad para identificar la correlación entre las señales acústicas en el discurso y el amplio rango de emociones producidos por el hablante. Los sistemas diseñados para llevar a cabo esta tarea, por lo general, son extremadamente sensibles a la variabilidad introducida por el hablante. Esta variabilidad se debe, especialmente a variaciones en la voz y en estilo causadas por ejemplo por diferentes estados de ánimo del hablante. [11]

Como se indicó, la percepción humana de las emociones asociadas al discurso proviene de múltiples variables. Una de las principales tareas en la detección automática de emociones es la recuperación automática de las variables más relevantes. El nivel de la voz es uno de los indicadores de la emoción. Un modo natural de medir este nivel de voz es mediante una función del voltaje recogido del micrófono. Sin embargo, no suele existir una relación directa entre el nivel de voz y dicha función del voltaje, debido fundamentalmente a las numerosas variables que influyen en los valores de esta última. Es posible medir la variación en la frecuencia de la vibración. Este factor es un buen indicador estadístico de algunos tipos de discurso. Otro factor importante, y fácil de conseguir, es la duración del discurso. Las variaciones en la intensidad del discurso sirven como variables para determinar el tipo de discurso: por ejemplo, la intensidad se desplaza hacia las vocales en los discursos sonoros. La calidad de la voz puede marcar las diferencias entre unas emociones y otras. Existen numerosas variables fonéticas relacionadas con la calidad de la voz: cociente de apertura de las cuerdas vocales, timbre de la voz, ruido, distribución de la energía, etc. Se ha intentado modelar el ritmo del discurso aunque esta tarea es más compleja. Es común el empleo de técnicas de reconocimiento de palabras en el discurso para la detección de aquellas palabras con claro significado emocional. [11]

Analizando audio afectado por diferentes emociones en la voz existe cierta relación entre parámetros como amplitud y frecuencia con la emoción del momento. Sin embargo, estos patrones son muy complicados para ser analizados manualmente. Por otro lado, una computadora mediante inteligencia artificial sí es capaz de encontrar dichos patrones. [11]

Podemos entender la clasificación como el proceso de asignar objetos a un conjunto

prefijado de categorías o clases. En el caso que nos atañe, estas categorías vienen determinadas por las emociones que queremos detectar a partir de una serie de características medidas sobre los individuos. Cuando conocemos *a priori* el conjunto de clases (emociones), el proceso de clasificación recibe el nombre de clasificación supervisada. Si el conjunto de clases nos es desconocido, el proceso recibe el nombre de clasificación no supervisada. Este método será considerado posteriormente. Si existe una función desconocida que va del espacio de características al espacio de emociones, nos referimos a ella como función objetivo. La solución del problema de clasificación será una estimación de la función objetivo, un clasificador. Por lo tanto, el problema de reconocimiento automático de emociones puede ser descrito como sigue: dada una muestra de objetos (caras, discursos, etc.), encontrar una función que asigne cada objeto a una de las emociones predefinidas, de modo que se minimice el error promedio de clasificación para futuras observaciones. El algoritmo de clasificación se conoce también con los nombres de proceso de aprendizaje, reconocimiento de patrones o discriminación. Existen dos etapas básicas en el diseño de un clasificador: la fase de entrenamiento y la fase de validación. En la fase de entrenamiento empleamos los datos muestras, llamados en este contexto, muestra de entrenamiento. Pueden imponerse restricciones sobre el clasificador, generalmente relativas a hipótesis sobre la distribución de las observaciones. Una vez construido el modelo, dichas hipótesis han de ser contrastadas. Empleamos la muestra de entrenamiento, considerando las restricciones si las hubiera, para construir el clasificador. Una vez disponemos de una regla de clasificación que asigna los objetos a las emociones, pasamos a la fase de validación. En esta fase, el clasificador obtenido en la fase de entrenamiento es empleado para clasificar las observaciones pertenecientes a la muestra de validación. [11]

Tanto en la fase de entrenamiento como en la fase de validación, el clasificador asigna cada observación a una emoción. La suma del número de objetos de la muestra de entrenamiento que no son correctamente clasificados, es decir, aquellas en las que la emoción observada no coincide con la emoción predicha por el clasificador, recibe el nombre de error de entrenamiento. Así mismo, la suma de las observaciones de la muestra de validación que no son correctamente clasificadas recibe el nombre de error de validación. La habilidad de un clasificador para clasificar correctamente observaciones que no pertenecen a la muestra de entrenamiento recibe el nombre de capacidad de generalización. Obviamente, se buscarán clasificadores tales que su capacidad de generalización sea máxima. [11]

El método de aprendizaje conocido como Redes Neuronales fue desarrollado simultáneamente en los ámbitos del análisis estadístico y de la inteligencia artificial. La idea central consiste en extraer combinaciones lineales de los atributos presentes en los objetos obteniendo una serie de características y modelizar las clases como funciones no lineales de dichas características. [14] y [15] presentan algunos ejemplos de la aplicación de esta técnica de clasificación al problema de reconocimiento de emociones. En el primero de estos artículos, las redes neuronales son adaptadas para identificar y agrupar los músculos de la cara que contribuyen a detectar las emociones. En el segundo artículo las emociones tratan de ser reconocidas a partir de señales obtenidas del discurso. Las redes neuronales obtuvieron mejores resultados de clasificación que

las técnicas alternativas con las que son comparadas: máquinas de vectores soporte, árboles de clasificación, k-vecinos más cercanos. [11]

Las redes bayesianas son clasificadores empleados para representar distribuciones conjuntas de modo que permitan calcular la probabilidad a posteriori de un conjunto de clases (emociones) dado un conjunto de características observadas en los objetos, y así clasificar los objetos en la clase más probable. Una red bayesiana se compone de un grafo dirigido en el cual cada nodo está asociado con una característica y con una distribución de probabilidad condicional. El grafo representa la estructura y las distribuciones de probabilidad y los parámetros de la red. La idea general consiste en usar una estrategia que pueda buscar de modo eficiente en el espacio de posibles estructuras y extraer aquella que de mejores resultados de clasificación. [11]

En Guatemala, el turismo representa una entrada significativa de ingresos para el país, pues se cuenta con una gran variedad de destinos sumamente llamativos, especialmente con los relacionados a la naturaleza. Sin embargo, existe otro gran grupo de personas que preferiría otro tipo de actividades o bien, complementar su experiencia en Guatemala de otra forma. Una de estas alternativas son los parques de diversiones; invirtiendo en ellos se conseguiría un ingreso aún mayor proveniente del turismo. Aun siendo de alta calidad, los parques de diversiones guatemaltecos no pueden competir con los de otros países debido al nivel tecnológico. Introducir animatrónicos lo más realistas posible podría ser un primer paso para mejorar las instalaciones y así atraer más clientela. Además, estos podrían cumplir con tareas simples que en la actualidad han sido adjudicadas a personas y así disminuir gastos. Los parques de diversiones son un ejemplo concreto, pero los animatrónicos podrían implementarse en cualquier ámbito orientado al entretenimiento. Además, abren las puertas a la utilización de mejores tecnologías debido a que son llamativos.

Con respecto al módulo de reconocimiento de voz, este le otorga mayor realismo al animatrónico. Es decir, no es un módulo imprescindible para construir un animatrónico, más el valor que este agrega es bastante significativo. Sin reconocimiento de voz y/o la habilidad para hablar, el animatrónico se convierte sencillamente en un personaje con movimiento, incapaz de realmente interactuar con el usuario. La habilidad de conversar con el usuario es un ingrediente clave para que el animatrónico cobre vida. En cuanto al reconocimiento de emociones, puede decirse que optimiza el reconocimiento de voz. Identificar la emoción de una persona permite interactuar con ella más acertadamente y así brindarle una mejor atención. Profundizando en el ejemplo de los parques de diversiones, un animatrónico podría amenizar de mejor manera el ambiente reconociendo que las personas muestran tonos de alegría o sorpresa, o proporcionar rutas de evacuación o instrucciones útiles en un desastre si detecta tonos de miedo o pánico, así como incluso referir a los clientes con encargados calificados si identifica tonos de enojo o sobresalto. El módulo de reconocimiento empático de

emociones será implementado en el animatrónico; sin embargo, es un módulo que podría aplicarse en diversos dispositivos o plataformas y convertirse en una herramienta muy útil en cualquier ámbito.

4.1. Objetivo general

Conseguir que el **animatrónico** pueda interpretar el idioma español y reaccionar según sea conveniente, respondiendo verbalmente similar a un ser humano y llevar a cabo un estudio sobre la detección de emociones en la voz.

4.2. Objetivos específicos

- Implementar un sistema que reconozca efectivamente patrones de voz y sea capaz de hablar con un tono de voz similar al humano.
- Dotar al sistema con la capacidad de conversar.
- Codificar movimientos respuesta accionados por voz.
- Determinar la factibilidad y viabilidad de detectar estados emocionales por medio de la voz.

Con este trabajo se espera contar con una **plataforma** de conversación que además responda a comandos relacionados con su entorno (en este caso un parque de diversiones). El medio para comunicarse será exclusivamente el habla: la plataforma responde a comandos por voz y responde hablando. Se estructurará de tal forma que resulte simple cambiar de idioma si se desea hacerlo en el futuro. Además, se proporcionará una manera fácil de agregar y suprimir comandos.

Esta plataforma será capaz de implementarse en cualquier dispositivo capaz de correr programas en Python. Es decir, mientras se cuente con una computadora en el medio donde quiera utilizarse, el programa correrá sin problemas.

La precisión de las respuestas habladas del sistema dependerán de la calidad del sonido, por lo que este aspecto está sujeto al micrófono con el cual se implemente el sistema. Todas las pruebas se realizarán con el micrófono incluido dentro de una computadora para garantizar un buen funcionamiento aún sin un micrófono especializado.

En cuanto al vocabulario y capacidad de conversación del programa, este dependerá del uso que se le de. Mientras más se utilice mejores resultados se obtendrán.

Con respecto al reconocimiento de emociones, se realizará el análisis y las pruebas necesarias para determinar si es posible discernir emociones con ayuda de la inteligencia artificial. De ser posible, se implementará en el programa. De no serlo, se presentarán las razones por las cuales no fue posible.

6.1. Voz a texto (*Speech to text*- STT)

Es un tipo de **software** que efectivamente toma contenido de audio y lo transcribe a palabras escritas para un procesador de palabras o alguna otra opción capaz de desplegar texto. Este tipo de tecnología es de gran utilidad para usuarios que necesitan generar gran cantidad de contenido escrito sin verse obligados a escribirlo manualmente. También es de gran utilidad para personas con discapacidades que les dificultan utilizar un teclado. Este tipo de software también tiende a conocerse como software de reconocimiento de voz. [16]

Este tipo de software funciona separando el audio en pequeñas muestras que pueden ser asociadas a fonemas o unidades de pronunciación. Después, algoritmos complejos clasifican los resultados para intentar predecir la frase dicha. Los STT han mejorado sobremanera en exactitud y evolucionado para tomar un rol más importante en las comunicaciones modernas mediante plataformas digitales. Algunos ejemplos de motores STT incluyen: PocketSphinx, Google y Julius. [16]

6.2. Texto a voz (*Text to speech* -TTS)

Es un tipo de aplicación de síntesis del habla que se utiliza para crear una versión de sonido hablado a partir del texto en un documento. Este tipo de software puede ser utilizado para habilitar la lectura de pantalla de algún dispositivo para una persona con discapacidades de la vista, o utilizarse simplemente para facilitarle tareas al usuario. TTS se utiliza comúnmente en conjunto con software de reconocimiento de voz. Algunos ejemplos de TTS incluyen: Lucent, Elan, AT&T y Google [17].

6.3. Cualidades del sonido

Son aquellas características fundamentales que definen y determinan un sonido y por ende, permiten diferenciarlos. Estos son altura, intensidad, duración, timbre y espacialidad [18].

6.3.1. Altura o tono (Frecuencia)

Es la cualidad que define si un sonido es grave o agudo respecto a otro. Grave y agudo son conceptos siempre relativos, un mismo sonido puede ser grave o agudo según con qué otro sonido se lo compare [18].

La altura es la propiedad más característica de los sonidos, tanto simples (sinusoidales) como complejos. Los sistemas de alturas se encuentran entre los más elaborados e intrincados jamás desarrollados tanto en la cultura occidental como no occidental. La altura tiene relación con la frecuencia de un sonido simple y con la frecuencia fundamental de un sonido complejo. La frecuencia de un sonido es una propiedad cuya producción puede a menudo controlarse, y se mantiene durante su propagación hacia los oídos del oyente. La frecuencia (cantidad de oscilaciones por segundos) se mide en Hertz. A mayor frecuencia más agudo es el sonido [18].

La altura puede describirse como un atributo unidimensional, es decir que todos los sonidos pueden ser ordenados a lo largo de una sola escala con respecto a la altura. Los extremos de esta escala son grave (sonidos con frecuencia baja) y agudo (sonidos con frecuencia alta). A veces, puede dificultarse la tarea de comparar la altura de dos sonidos distintos por factores tales como la diferencia tímbrica entre ellos o el componente de ruido en cada uno [18].

La altura en su sentido musical tiene un rango de alrededor de 20 a 5000 Hz, más o menos el rango de las fundamentales de las cuerdas de un piano o los tubos de un órgano. Sonidos con frecuencias más altas son audibles, pero sin una sensación definida de altura. El oído humano es capaz de percibir frecuencias de entre 20 y 20000 Hertz. Las frecuencias inferiores a 20 Hz se denominan infrasonidos y las frecuencias superiores a 20000 Hz son ultrasonidos [18].

En el caso de los sonidos complejos, el oído se comporta como un analizador de Fourier. Si el sonido es armónico las frecuencias de los parciales son proporcionales a los números enteros cada armónico, según su frecuencia, excita la zona correspondiente de la membrana basilar y envía información al cerebro. Este se encarga de realizar la fusión de los armónicos, percibiéndose un sólo sonido con la frecuencia fundamental, incluso oímos ésta, aunque no esté presente en el espectro (lo que se conoce como altura residual). Este análisis armónico se combina también con la percepción de periodicidad de la forma de onda. En el caso de los sonidos inarmónicos la percepción de altura es mucho más compleja y misteriosa; al oído le llegan una serie de frecuencias

de las que no es capaz de extraer la fundamental y tampoco hay periodicidad; no obstante, hay sonidos inarmónicos como los de la marimba, xilófono, etc. donde sí se oye una altura definida, que suele corresponder al parcial de más amplitud o al más grave; en fin, queda mucho por investigar todavía en éste terreno [18].

6.3.2. Intensidad (Amplitud)

Es la cualidad que define si un sonido es fuerte o suave respecto a otro; al igual que la altura es un concepto relativo, se puede decir que un sonido es fuerte o suave siempre que se lo relacione con otro. La unidad de medida de la intensidad es el decibel (dB) [18].

A la fuerza con la que oscila la fuente sonora se le denomina energía de oscilación. La energía de oscilación de la onda sonora determina la intensidad del sonido (mayor energía de oscilación = más fuerte) [18].

La amplitud es el valor máximo absoluto de la variación de presión, positiva o negativa, de la onda sonora. Está relacionada con nuestra sensación dinámica del sonido y el oído es extremadamente sensible a esta magnitud; así, si la presión atmosférica es 100,000 unidades, el oído puede detectar una variación de presión de 0,00002 unidades que correspondería al sonido más débil o umbral de audición [18].

Hay varias maneras de medir los niveles de intensidad. La más habitual es la unidad llamada decibel (dB), que es el escalón mínimo de percepción. El decibel es definido como función logarítmica a fin de adecuarse aproximadamente a nuestro comportamiento perceptivo. El oído humano puede percibir intensidades de entre 20 y 120 dB, los sonidos que están por encima de 120 dB dañan el oído [18].

El nivel de audibilidad máxima es menos fácil de establecer: hay una intensidad mínima por debajo de la cual el oído humano no percibe sonido, pero no hay una intensidad máxima por encima de la cual el oído deje de percibir sonido. En ese extremo, las cosas son diferentes. Establecido en el límite inferior de audibilidad el cero convencional de la escala comparativa de decibels, se tendrá un umbral de dolor en un entorno que se sitúa entre 120 y 140 dB, umbral que preanuncia la posibilidad de daños irreparables en el órgano auditivo (que pueden llevar a la sordera total). El susurro de una persona cercana o un murmullo a un metro de distancia estará en torno de los 20 dB. Una conversación normal podrá variar entre los 60 y 70 dB. Una calle con tránsito se situará entre los 80 y 90 dB. Un avión de tipo jet a 15 metros de distancia estará ya en los 120 dB, y un martillo neumático a un metro de distancia sobrepasará este valor en unos 7 u 8 dB [18].

6.3.3. Duración (Tiempo)

Se refiere al lapso de tiempo durante el cual percibimos un sonido. Está determinada fundamentalmente por las características de la fuente sonora. En este sentido

tenemos varias posibilidades:

- En algunos casos la fuente sonora permite que se le pueda aplicar energía en forma continua luego del estímulo inicial (como un violín o una gaita). Aquí la duración del sonido depende del tiempo durante el cual le apliquemos energía.
- En otros casos el sonido se extingue progresivamente (ataque y caída). En estas fuentes sonoras, el sonido puede apagarse rápidamente (por ejemplo un tambor), o puede prolongarse unos cuantos segundos (por ejemplo en la guitarra). En este tipo de fuente sonora también incide en la duración del sonido la fuerza que se le aplica a dicha fuente sonora.

Hay un **umbral** inferior situado alrededor del 1/15 de segundo, por debajo del cual se hace difícil discernir sonidos sucesivos. El umbral superior es, en principio, indeterminado. En materia de duración, los límites son más psicológicos que físicos [18].

El papel de lo psicológico en nuestra percepción de los fenómenos acústicos es muy importante. Nuestro sistema perceptivo puede dejar de oír un sonido que no le interesa. O bien puede establecer una audición selectiva [18].

Existe una duración objetiva, que es la duración de los sonidos posible de ser medida físicamente. La unidad usada suele ser el segundo. Pero existe también una duración subjetiva que es la duración que nosotros percibimos en los sonidos. Suele usarse la unidad “dura” y se ha definido a 1 dura como la duración subjetiva de un sonido sinusoidal de 1 kHz, con 60 dB de SPL y 1 s de duración objetiva. Duplicando y reduciendo a la mitad podemos determinar la relación existente entre las duraciones objetivas y subjetivas [18].

6.3.4. Timbre (Armónicos)

En general se puede decir que el timbre permite diferenciar dos sonidos de igual altura, intensidad y duración, pero de procedencia diversa. Depende del grado de complejidad del movimiento oscilatorio que lo origina, que se refleja en la forma de la onda. El timbre nos permite diferenciar las fuentes sonoras de donde proviene un sonido [18].

Se puede hablar de timbre en tres niveles:

- General. Cuando distingue elementos de distintas clases, como diferenciar una guitarra de una flauta
- Parcial. Cuando distingue elementos de una misma clase, como diferenciar tipos de guitarras

- Particular. Cuando distingue las posibilidades de un único elemento dentro de una clase dada, como diferenciar los distintos modos de tocar una guitarra: pulsando las cuerdas, con o sin púa, golpeando la caja, etc.

Los principales factores que influyen en la determinación del timbre son:

- La envolvente espectral, es decir, la intensidad relativa de los parciales.
- La envolvente dinámica, en particular la conjunción de las envolventes dinámicas de cada uno de los parciales; También se le llama envolvente temporal (no confundirla con la espectral) es la línea imaginaria que une los puntos de amplitud máxima de la onda en el tiempo. En el sonido de una trompeta, por ejemplo, se pueden apreciar tres fases: el ataque, que es cuando el sonido crece, el estado estable, donde la amplitud se mantiene y la caída, cuando el sonido se apaga. Cada instrumento tiene su propio tipo de envolvente, lo cual ayuda a identificar su timbre; por ejemplo, la envolvente de la marimba y muchos instrumentos de percusión tiene un ataque muy rápido seguido de la caída sin estado estable; en el piano el ataque es más lento que en los instrumentos de percusión y también se extingue rápidamente; en las cuerdas y vientos el ataque es más lento y hay un estado estable.
- Los transitorios, que son parciales de muy corta duración que se generan en el ataque, pero también en la caída de un sonido. Ello hace que todos los sonidos tengan siempre una componente de ruido.

El timbre es un fenómeno dinámico, quiere decir que varía en el tiempo. Esto se debe a la evolución de las envolventes dinámicas de cada uno de los parciales que hace que la envolvente espectral (es decir, la intensidad relativa de los parciales) sea distinta en cada momento [18].

La envolvente tímbrica es la superficie que generan las envolventes dinámicas de todos los parciales que componen ese sonido. Helmholtz descubrió que el timbre está también determinado por la forma de onda y su espectro, es decir, por el número de armónicos presentes y sus intensidades respectivas. Por ejemplo, el violín suena más brillante que la flauta porque sus armónicos agudos tienen más intensidad; el clarinete suena hueco porque sus armónicos pares son débiles, etc. Especialmente importantes son los formantes, que son picos o zonas más prominentes del espectro; la voz, por ejemplo, tiene de 3 a 5 formantes, y las distintas vocales corresponden a distintas posiciones de los formantes en el rango de frecuencias [18].

En cuanto a la percepción del timbre, se lo suele caracterizar con términos visuales: brillante, opaco, etc.

6.3.5. Espacialidad

La espacialidad no se considera un parámetro sonoro en la medida de los anteriores, es percibida en función de varios factores: la intensidad relativa entre el sonido original y la resonancia del mismo sonido en el espacio, nuestra experiencia de la variación tímbrica de ese tipo de sonido con el aumento o disminución de la distancia, la proporción de las intensidades con que llega a cada uno de nuestros oídos, el **defasaje** de los instantes de incidencia en uno y otro. La percepción espacial se modifica, entre otras cosas, con la altura del sonido, que resulta menos localizable en frecuencias graves, y más nítido en su ubicación en frecuencias agudas. A la espacialidad se agrega el eventual movimiento, es decir, la variabilidad de ubicación de aquello que suena en el espacio. La eclosión de las técnicas electroacústicas de composición en la segunda mitad del siglo XX ha permitido trabajar la posibilidad de desplazamiento espacial de los sonidos en el transcurso del tiempo como una posibilidad real de comportamiento musical de éstos. [18]

6.4. Audio digital

6.4.1. Parámetros fundamentales

Los parámetros básicos para describir la secuencia de muestras que representa el sonido son:

- El número de canales: 1 para mono, 2 para estéreo, 4 para el sonido cuadrafónico, etc.
- Tasa de muestreo: El número de muestras tomadas por segundo en cada canal.
- Número de bits por muestra: Habitualmente 8 o 16 bits.

Como regla general, las muestras de audio multicanal suelen organizarse en tramas. Una trama es una secuencia de tantas muestras como canales, correspondiendo cada una a un canal. En este sentido el número de muestras por segundo coincide con el número de tramas por segundo. En estéreo, el canal izquierdo suele ser el primero[19].

Como ejemplo concreto, el sonido telefónico pasa por un proceso de **compansión** (p.ej. compresión logarítmica) y se codifica en secuencias de 8 bits. Sin embargo, los datos almacenados se corresponden a un rango dinámico lineal de 14 bits, por lo que hay cierta ambigüedad en el número de bits por muestra. [19]

6.4.2. Archivos WAV

La forma de onda del archivo de audio en formato o WAVE es un formato de archivo de audio que se desarrolló en conjunto con Microsoft e IBM. Fue lanzado

por primera vez en 1991. El nombre del archivo de los archivos WAVE termina con la extensión de .WAV o .WAVE. Es por eso que los archivos en este formato son universalmente también aceptados como archivos WAV. Es el formato más utilizado en Windows PC para audio sin editar y sin comprimir. El archivo WAV puede contener el audio comprimido como el audio sin comprimir. Los archivos WAV más comunes son sin comprimir; por lo que, WAV es considerado como un formato de audio sin pérdida, conservando la alta calidad del sonido. Este tipo de archivo es utilizado en las estaciones de radio también [20].

Aunque WAV es comúnmente sin comprimir, todavía es compatible con compresión con pérdida y puede utilizar el administrador de compresión de audio para lograrlo. Un archivo WAV no solo contiene la parte de audio, sino también la información sobre el archivo que incluye el mono o estéreo de la propiedad, la profundidad de bits, la frecuencia de muestreo y el número de pistas [20].

WAV es un derivado de RIFF (Resource Interchange File Format), un formato contenedor genérico para almacenar datos de sonido y vídeo en etiquetados. El tamaño máximo de archivo de los archivos WAV es de hasta 4 GB. Los archivos más grandes que pasen este límite no serán compatibles para codificarse en este formato [20].

Los archivos WAV sin comprimir tienen la calidad de audio sin pérdidas. Es compatible con Windows y Macintosh OS, así que no tendrá ninguna dificultad para reproducirlos en estas plataformas. La edición y manipulación de este tipo de archivo también es muy fácil, ya que no requiere de ninguna codificación o decodificación. El tamaño grande será un problema si no tiene suficiente espacio en sus dispositivos. Esto también hace que sea limitado compartir archivos WAV a través de Internet.[20]

6.4.3. Archivo .l16

Linear 16-bit Pulse-Code Modulation (PCM), conocido simplemente como L16, es un formato de audio de 16 bits, lineal, sin comprimir y modulado por impulsos codificados. Se utiliza para enviar un archivo PCM en bruto. Comúnmente, estos archivos están contenidos en un archivo en formato WAV. Al utilizar directamente un archivo L16 es necesario indicar la frecuencia de muestreo. Adicionalmente, puede indicarse el número de canales y si el archivo se ordenó según el bit más significativo (big endian) o el menos significativo (little endian) [21].

6.5. Dominio de frecuencia

La representación en el dominio del tiempo brinda las amplitudes de la señal en los instantes del tiempo durante los cuales fue muestreada. Sin embargo, en muchos casos, es necesario saber el contenido de la frecuencia de una señal más que las amplitudes de las muestras individuales.[22]

El teorema de Fourier afirma que cualquier forma de onda en el dominio puede ser representada por la suma acumulada de senos y cosenos. Entonces la misma forma de onda puede ser representada en el dominio de frecuencia como un par de valores de amplitud y fase en la frecuencia de cada componente.[22]

En el dominio de frecuencia, pueden separarse conceptualmente las ondas sinusoidales que añaden para formar la señal compleja en el dominio del tiempo. La Figura 2 muestra los componentes de la frecuencia, los cuales se separan en el dominio del tiempo como impulsos distintos en el dominio de frecuencia. La amplitud de cada línea de frecuencia es la amplitud de la forma de onda del tiempo para este componente de frecuencia. La representación de una señal en términos de sus componentes de frecuencia individuales es la representación de la señal en el dominio de frecuencia. La representación del dominio de frecuencia podría proporcionar una mejor comprensión sobre la señal y el sistema en el que fue generada.[22]

Las muestras de una señal obtenida desde el dispositivo de adquisición de datos constituyen la representación en el dominio del tiempo de la señal. Algunas medidas, como ruido o distorsión armónica, son difíciles de cuantificar al inspeccionar la forma de onda del tiempo. Cuando la misma señal es mostrada en el dominio de frecuencia pueden medirse fácilmente las frecuencias armónicas y las amplitudes. Si se tiene ruido en una señal medida, esta puede transformarse del dominio del tiempo al dominio de frecuencia para aislar la alteración en ella.[22]

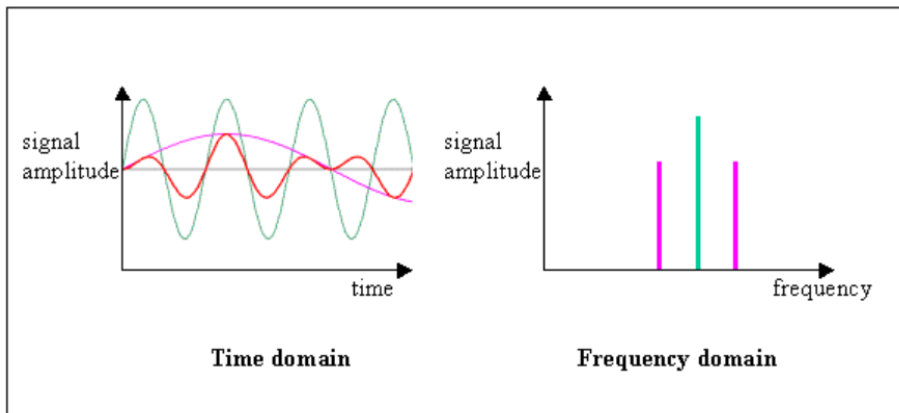


Figura 2: Dominio del tiempo en comparación con el Dominio de frecuencia [22]

6.5.1. Transformada Rápida de Fourier - FFT

La Transformada Rápida de Fourier (FFT) proporciona un método para examinar una relación en términos del dominio de frecuencia [22]. Es un algoritmo que reduce el tiempo de cálculo de n^2 pasos a $n \cdot \log_2(n)$. El único requisito es que el número de puntos en la serie tiene que ser una potencia de 2 (2n puntos), por ejemplo 32, 1024, 4096, etc. [23]

6.5.2. Filtro Butterworth para la voz

El filtro de Butterworth es uno de los filtros electrónicos más básicos, diseñado para producir la respuesta más plana que sea posible hasta la frecuencia de corte. En otras palabras, la salida se mantiene constante casi hasta la frecuencia de corte, luego disminuye a razón de $20n$ dB por década (ó $6n$ dB por octava), donde n es el número de polos del filtro.

La respuesta en frecuencia de un filtro Butterworth es muy plana (no posee ondulaciones) en la banda pasante, y se aproxima a cero en la banda rechazada. Visto en un gráfico logarítmico, esta respuesta desciende linealmente hasta el infinito negativo. Para un filtro de primer orden, la respuesta varía en -6 dB por octava (-20 dB por década). (Todos los filtros de primera orden, independientemente de sus nombres, son idénticos y poseen la misma respuesta en frecuencia.) Para un filtro Butterworth de segundo orden, la respuesta en frecuencia varía en -12 dB por octava, en un filtro de tercer orden la variación es de -18 dB, y así por delante. Los filtros Butterworth poseen una caída en su magnitud como una función lineal con ω . El Butterworth es el único filtro que mantiene el mismo formato para órdenes más elevados (sin embargo con una inclinación más íngreme en la banda atenuada) mientras otras variedades de filtros (Bessel, Chevyshev, elíptico) poseen formatos diferentes para órdenes más elevados.

Comparado con un filtro chevyshev del Tipo I/Tipo II o con un filtro elíptico, el filtro Butterworth posee una caída relativamente más lenta, y por lo tanto irá a requerir un orden mayor para implementar una especificación de banda rechazada particular. Sin embargo, el filtro Butterworth presentará una respuesta en fase más lineal en la banda pasante que los filtros Chebyshev del Tipo I/Tipo II o elípticos.

En muchos receptores o transmisores de voz por radio, a menudo se necesita un filtro pasabanda de voz para mantener la señal de voz dentro de la banda vocal de 300-3000Hz. El filtro se usa en el lado del transmisor para mantener el espectro transmitido lo más estrecho posible o en el lado del receptor para suprimir cualquier ruido no deseado. Un filtro de voz ideal tendría una respuesta plana ideal dentro de la banda y atenuación infinita para las frecuencias fuera de la banda. [24]

Un filtro de voz ideal no existe. Sin embargo, un filtro Butterworth de paso de banda de 5to orden para la banda 300-3000Hz se considera una buena solución para aplicaciones de radiotelefonía. Un filtro de orden mayor tendría mejores resultados aunque resulta más complicado de implementar. [24]

6.6. Aprendizaje de máquina *Machine Learning*

Así como la inteligencia artificial es el intento de conseguir que los ordenadores tengan un comportamiento inteligente similar al de los humanos, el aprendizaje de máquina fue el primer paso en esta dirección y tenía como objetivo que los ordenadores

podrían aprender por sí solos cosas que los humanos no son capaces de describir o reflejar en ningún lenguaje de programación. Así es como empezó el aprendizaje de máquina, una disciplina (al igual que la inteligencia artificial) muy ligada a las matemáticas y a la estadística y que desde hace casi medio siglo ha ido haciendo progresos hasta nuestros días. En este momento aprendizaje de máquina es uno de los campos más activos y potentes de la informática con aplicaciones en todos los sectores imaginables. El aprendizaje de máquina se puede dividir en dos tipos: supervisado y no supervisado [25].

6.6.1. Aprendizaje supervisado

Es uno de los aprendizajes más interesantes y empleados en ciencia porque permite el aprendizaje de patrones o características que no son aparentes para el ser humano. Este tipo de aprendizaje de máquina es ideal para tareas de clasificación como puede ser saber qué tipo de partículas han aparecido en una colisión del LHC [25]. La forma en la que se ejecuta este tipo de procedimientos es bastante sencilla y consta de un paso de entrenamiento en el que al algoritmo se le presentan datos ya analizados para que “aprenda” las características de cada clase de partícula, por ejemplo. Durante esta fase el propio ordenador es capaz de establecer conexiones entre los datos de la colisión y el tipo de partículas que han aparecido, sin necesidad de saber física cuántica [25]. Una vez entrenado suele haber una fase de prueba en el que se le vuelven a dar datos conocidos pero se presentan sin solución de forma que el ordenador tiene que aplicar todo lo aprendido mediante aprendizaje de máquina para poder descubrir que tipo de partícula se produjo. Si supera esta fase con un alto porcentaje de aciertos el algoritmo está ya preparado para enfrentarse a nuevos datos y analizarlos mejor y más rápido que un humano [25].

Algoritmos de Aprendizaje Supervisado

Regresión Logística - LR

La regresión logística es una poderosa manera estadística de modelar un resultado binomial con una o más variables explicativas. Mide la relación entre la variable dependiente categórica y una o más variables independientes estimando las probabilidades utilizando una función logística, que es la distribución logística acumulativa. [26]

Análisis Discriminante Lineal - LDA

Suponiendo que un conjunto de objetos se clasifica en una serie de grupos; el Análisis Discriminante equivale a un análisis de regresión donde la variable dependiente es categórica y tiene como categorías la etiqueta de cada uno de los grupos, y las variables independientes son continuas y determinan a qué grupos pertenecen los objetos. Se trata de encontrar relaciones lineales entre las variables continuas que mejor discriminan en los grupos dados a los objetos. Además, se trata de definir una regla de decisión que asigne un objeto nuevo, que no sabemos clasificar previamente, a uno de

los grupos prefijados [27].

K Vecinos más cercanos - KNN

El algoritmo clasifica cada dato nuevo en el grupo que corresponda, según tenga k vecinos más cerca de un grupo o de otro. Es decir, calcula la distancia del elemento nuevo a cada uno de los existentes, y ordena dichas distancias de menor a mayor para ir seleccionando el grupo al que pertenecer. Este grupo será, por tanto, el de mayor frecuencia con menores distancias.[28]

Árbol de decisiones - CART

Un árbol de decisiones es una herramienta de apoyo a la decisión que utiliza un gráfico o un modelo similar a un árbol de decisiones y sus posibles consecuencias, incluidos los resultados de eventos fortuitos, los costos de recursos y la utilidad. Presentan una apariencia como la que se observa en la Figura 3. Desde el punto de vista de la toma de decisiones empresariales, un árbol de decisiones es el número mínimo de preguntas sí / no que uno tiene que hacer, para evaluar la probabilidad de tomar una decisión correcta, la mayoría del tiempo. Este método le permite abordar el problema de una manera estructurada y sistemática para llegar a una conclusión lógica. [26]

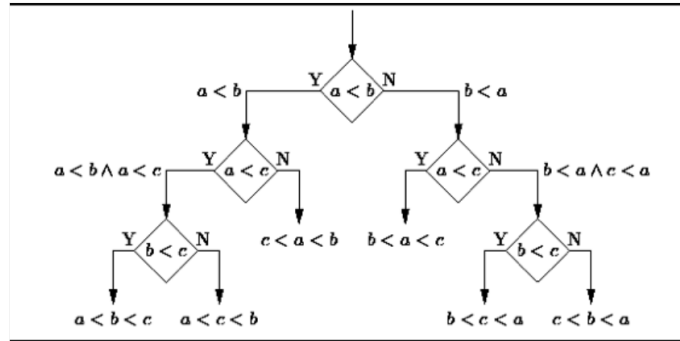


Figura 3: Estructura de un árbol de decisiones [26]

Clasificador de Naïve Bayes - NB

Los clasificadores Naïve Bayes son una familia de simples clasificadores probabilísticos basado en la aplicación de Bayes "teorema con fuertes (Naïve) supuestos de independencia entre las características". Esta basado en la ecuación 1 donde $P(\frac{A}{B})$ es probabilidad posterior, $P(\frac{B}{A})$ es probabilidad, $P(A)$ es probabilidad previa de clase, y $P(B)$ predictor probabilidad previa. [26]

$$P(\frac{A}{B}) = \frac{P(\frac{B}{A})P(A)}{P(B)} \quad (1)$$

Máquina de vectores - SVM

SVM es un algoritmo de clasificación binario. Dado un conjunto de puntos de dos tipos en el lugar N dimensional, SVM genera un hiperplano (N - 1) dimensional para

separar esos puntos en dos grupos. Por ejemplo, si se tienen algunos puntos de dos tipos en un papel que son linealmente separables, SVM encontrará una línea recta que separa esos puntos en dos tipos y situados lo más lejos posible de todos esos puntos, como muestra la Figura 4. En términos de escala, algunos de los mayores problemas que se han resuelto utilizando SVMs (con implementaciones adecuadamente modificadas) son publicidad en pantalla, reconocimiento de sitios de empalme humanos, detección de género basada en imágenes, clasificación de imágenes a gran escala. [26]

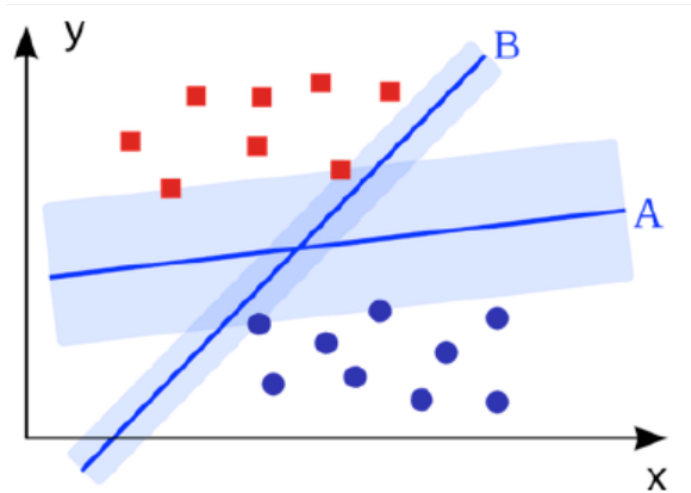


Figura 4: Ejemplo máquinas de vectores [26]

6.6.2. Aprendizaje no supervisado

Por otro lado tenemos el aprendizaje no supervisado en el que no existe una fase de entrenamiento. En este caso el algoritmo suele poseer criterios de calidad de la respuesta y “aprende” cuál es la mejor manera de llegar al resultado óptimo lo más rápido posible. Este tipo de aprendizaje de máquina suele ser más limitado en aplicabilidad pero es también muy potente y utilizado [25].

A continuación un ejemplo, suponiendo que quiera hacerse un ajuste lineal a una serie de puntos. La convención matemática es que la mejor recta es aquella en la que la suma de distancias al cuadrado de los puntos a la recta es menor. Éste será el criterio que le daremos a nuestro algoritmo, pero no le diremos cómo conseguir este resultado, eso lo aprenderá él solo. Este tipo de aprendizaje de máquina suele tener un comienzo totalmente aleatorio, por ejemplo poniendo rectas en cualquier posición y dirección y viendo cómo varía la suma de distancias al cuadrado [25].

Con el paso del tiempo y de repetir y repetir el proceso el ordenador se irá dando cuenta de cuáles son las características que tienen todos los ajustes lineales óptimos. De esta forma, si se le presentan nuevos datos tardará mucho menos en conseguir la recta porque ya “sabe lo que busca” la idea de esta técnica de aprendizaje de máquina es que el ordenador será capaz de encontrar la fórmula de calcular dicha

recta inmediatamente, dados los puntos [25].

6.7. Estadística descriptiva

La estadística descriptiva es la rama de las Matemáticas que recolecta, presenta y caracteriza un conjunto de datos (por ejemplo, edad de una población, altura de los estudiantes de una escuela, temperatura en los meses de verano, etc.) con el fin de describir apropiadamente las diversas características de ese conjunto. Al conjunto de los distintos valores numéricos que adopta un carácter cuantitativo se llama variable estadística. [29]

Las variables pueden ser de dos tipos:

- Variables cualitativas o categóricas: no se pueden medir numéricamente (por ejemplo: nacionalidad, color de la piel, sexo).
- Variables cuantitativas: tienen valor numérico (edad, precio de un producto, ingresos anuales).

Las variables también se pueden clasificar en:

- Variables unidimensionales: solo recogen información sobre una característica (por ejemplo: edad de los alumnos de una clase).
- Variables bidimensionales: recogen información sobre dos características de la población (por ejemplo: edad y altura de los alumnos de una clase).
- Variables pluridimensionales: recogen información sobre tres o más características (por ejemplo: edad, altura y peso de los alumnos de una clase).

Por su parte, las variables cuantitativas se pueden clasificar en discretas y continuas:

- Discretas: solo pueden tomar valores enteros (1, 2, 8, -4, etc.). Por ejemplo: número de hermanos (puede ser 1, 2, 3..., etc., pero, por ejemplo, nunca podrá ser 3.45).
- Continuas: pueden tomar cualquier valor real dentro de un intervalo. Por ejemplo, la velocidad de un vehículo puede ser 90.4 km/h, 94.57 km/h...etc.

Cuando se estudia el comportamiento de una variable hay que distinguir los siguientes conceptos:

- Individuo: cualquier elemento que porte información sobre el fenómeno que se estudia. Así, si estudiamos la altura de los niños de una clase, cada alumno es un individuo; si se estudia el precio de la vivienda, cada vivienda es un individuo.
- Población: conjunto de todos los individuos (personas, objetos, animales, etc.) que porten información sobre el fenómeno que se estudia. Por ejemplo, si se estudia el precio de la vivienda en una ciudad, la población será el total de las viviendas de dicha ciudad.
- Muestra: subconjunto que seleccionado de una población. Por ejemplo, si se estudia el precio de la vivienda de una ciudad, lo normal será no recoger información sobre todas las viviendas de la ciudad (sería una labor muy compleja), sino que se suele seleccionar un subgrupo (muestra) que se entienda que es suficientemente representativo.

Las variables aleatorias son variables que son seleccionadas al azar o por procesos aleatorios. [29]

A continuación, se presentan las mediciones estadísticas utilizadas en este trabajo de investigación.

6.7.1. Medidas de tendencia

Media aritmética

La media aritmética de n valores, es igual a la suma de todos ellos dividida entre n [29].

Mediana

Es el punto central de una serie de datos ordenados de forma ascendente o descendente [29].

Moda

La moda de un conjunto de datos numéricos es el valor que más se repite, es decir, el que tiene el mayor número de frecuencias absolutas. La moda puede ser no única e inclusive no existir [29].

6.7.2. Medidas de dispersión

La dispersión mide que tan alejados están un conjunto de valores respecto a su media aritmética. Así, cuanto menos disperso sea el conjunto, más cerca del valor medio se encontrarán sus valores. Este aspecto es de vital importancia para el estudio de investigaciones. Se llaman medidas de dispersión aquellas que permiten retratar la distancia de los valores de la variable a un cierto valor central, o que permiten identificar la concentración de los datos en un cierto sector del recorrido de la variable. Se trata de coeficientes para variables cuantitativas [29].

Rango

El rango de una distribución es la diferencia entre el valor máximo (M) y el valor mínimo (m) de la variable estadística [29].

Varianza

La varianza mide la mayor o menor dispersión de los valores de la variable respecto a la media aritmética. Cuanto mayor sea la varianza mayor dispersión existirá y por tanto, menor representatividad tendrá la media aritmética [29].

Desviación estándar

La desviación estándar o desviación típica se define como la raíz cuadrada de los cuadrados de las desviaciones de los valores de la variable respecto a su media. La desviación estándar es una medida estadística de la dispersión de un grupo o población. Una gran desviación estándar indica que la población está muy dispersa respecto de la media. Una desviación estándar pequeña indica que la población está muy compacta alrededor de la media [29].

6.7.3. Medidas de distribución

Oblicuidad

Esta medida nos permite identificar si los datos se distribuyen de forma uniforme alrededor del punto central (Media aritmética). La asimetría presenta tres estados diferentes, como se muestra en la Figura 5, cada uno de los cuales define de forma concisa como están distribuidos los datos respecto al eje de asimetría. Se dice que la asimetría es positiva cuando la mayoría de los datos se encuentran por encima del valor de la media aritmética, la curva es simétrica cuando se distribuyen aproximadamente la misma cantidad de valores en ambos lados de la media y se conoce como asimetría

negativa cuando la mayor cantidad de datos se aglomeran en los valores menores que la media [30].

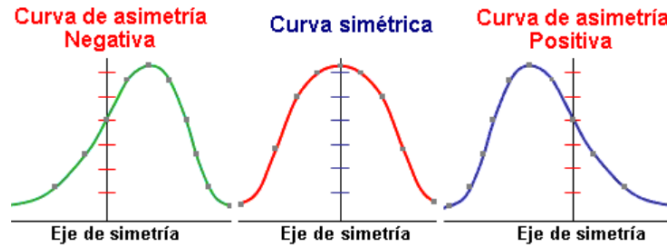


Figura 5: Tipos de oblicuidad o simetría [30]

Curtosis

Esta medida determina el grado de concentración que presentan los valores en la región central de la distribución. Por medio del Coeficiente de Curtosis, podemos identificar si existe una gran concentración de valores (Leptocúrtica), una concentración normal (Mesocúrtica) ó una baja concentración (Platicúrtica) [30], como se muestra en la Figura 6.

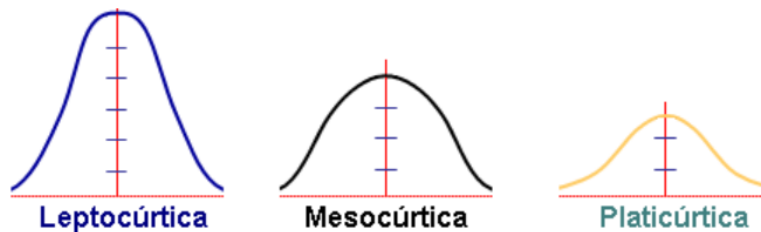


Figura 6: Tipos de Curtosis [30]

6.8. Validación cruzada

Es una técnica utilizada para evaluar los resultados de un análisis estadístico y garantizar que son independientes de la partición entre datos de entrenamiento y prueba. Consiste en repetir y calcular la media aritmética obtenida de las medidas de evaluación sobre diferentes particiones. Se utiliza en entornos donde el objetivo principal es la predicción y se quiere estimar cómo de preciso es un modelo que se llevará a cabo a la práctica. Es una técnica muy utilizada en proyectos de inteligencia artificial para validar modelos generados. [31]

La validación cruzada proviene de la mejora del método de retención o *holdout method*. Este consiste en dividir en dos conjuntos complementarios los datos de muestra, realizar el análisis de un subconjunto (denominado datos de entrenamiento o

training set), y validar el análisis en el otro subconjunto (denominado datos de prueba o *test set*), de forma que la función de aproximación solo se ajusta con el conjunto de datos de entrenamiento y a partir de aquí calcula los valores de salida para el conjunto de datos de prueba (valores que no ha analizado antes). La ventaja de este método es que es muy rápido a la hora de computar. Sin embargo, este método no es demasiado preciso debido a la variación de los resultados obtenidos para diferentes datos de entrenamiento. La evaluación puede depender en gran medida de cómo es la división entre datos de entrenamiento y de prueba, y por lo tanto puede ser significativamente diferente en función de cómo se realice esta división. Debido a estas carencias aparece el concepto de validación cruzada. [31]

6.8.1. Tipos de validaciones cruzadas

Validación cruzada de K iteraciones

En la validación cruzada de K iteraciones o *K-fold crossvalidation* los datos de muestra se dividen en K subconjuntos. Uno de los subconjuntos se utiliza como datos de prueba y el resto (K-1) como datos de entrenamiento. El proceso de validación cruzada es repetido durante k iteraciones, con cada uno de los posibles subconjuntos de datos de prueba. Finalmente se realiza la media aritmética de los resultados de cada iteración para obtener un único resultado. Este método es muy preciso puesto que evaluamos a partir de K combinaciones de datos de entrenamiento y de prueba, pero aun así tiene una desventaja, y es que, a diferencia del método de retención, es lento desde el punto de vista computacional. En la práctica, la elección del número de iteraciones depende de la medida del conjunto de datos. Lo más común es utilizar la validación cruzada de 10 iteraciones (*10-fold cross-validation*). [31]



Figura 7: Validación cruzada de K iteraciones con K=4 [31]

Validación cruzada aleatoria

Este método consiste en dividir aleatoriamente el conjunto de datos de entrenamiento y el conjunto de datos de prueba. Para cada división la función de aproxima-

ción se ajusta a partir de los datos de entrenamiento y calcula los valores de salida para el conjunto de datos de prueba. El resultado final se corresponde a la media aritmética de los valores obtenidos para las diferentes divisiones. La ventaja de este método es que la división de datos entrenamiento-prueba no depende del número de iteraciones. Pero, en cambio, con este método hay algunas muestras que quedan sin evaluar y otras que se evalúan más de una vez, es decir, los subconjuntos de prueba y entrenamiento se pueden solapar. [31]

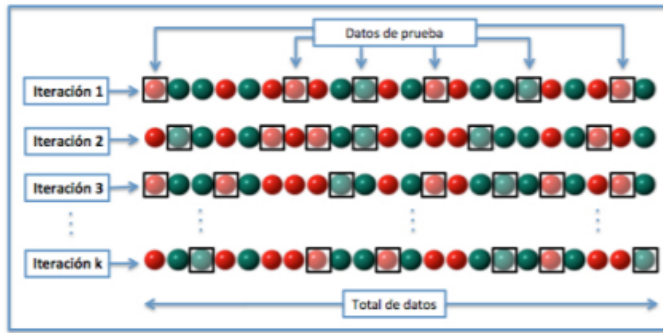


Figura 8: Validación cruzada aleatoria con K iteraciones [31]

Validación cruzada

La validación cruzada dejando uno fuera o *Leave-one-out cross-validation* (LOOCV) implica separar los datos de forma que para cada iteración tengamos una sola muestra para los datos de prueba y todo el resto conformando los datos de entrenamiento. La evaluación viene dada por el error, y en este tipo de validación cruzada el error es muy bajo, pero en cambio, a nivel computacional es muy costoso, puesto que se tienen que realizar un elevado número de iteraciones, tantas como N muestras tengamos y para cada una analizar los datos tanto de entrenamiento como de prueba. [31]

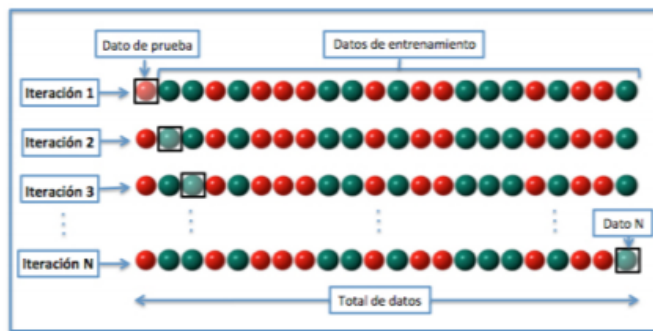


Figura 9: Validación cruzada dejando uno fuera (LOOCV)[31]

Lista de librerías

A continuación, se enumeran todas las librerías y módulos de Python utilizados a lo largo de este trabajo.

1. *speech_recognition* [32]
2. *pyttsx* [33]
3. *chatterbot* [7]
4. *re* [34]
5. *unicodedata* [35]
6. *random* [36]
7. *sys* [37]
8. *pyaudio* [38]
9. *wave* [39]
10. *scipy* [40]
11. *numpy* [41]
12. *pandas* [42]
13. *sklearn* [43]
14. *matplotlib* [44]

15. *os* [45]
16. *stat* [46]
17. *pyserial* [47]

8.1. Metodología

Para crear un programa de reconocimiento de voz con **Python** fue necesario instalar la librería *speech_recognition*. Esta se instaló desde el CMD con el comando "Pip install SpeechRecognition". Dicha librería facilitó sobremanera el proceso de reconocimiento de voz, lo cual se logró mediante las funciones del método "Recognizer" y el método "Microphone". Como primer paso, se crearon instancias de estos dos métodos.

El siguiente paso fue definir un umbral de sensibilidad adecuado. La función "energy_threshold" del método "Recognizer" permite manipular esta cantidad manualmente. Se hizo uso también de la función "adjust_for_ambient_noise" del mismo método.

Tras seleccionar el umbral, se entró en un ciclo infinito de escuchar y convertir a texto. Se escuchó el micrófono mediante la función "listen", también del método "Recognize". Posteriormente, lo escuchado se convirtió a texto mediante el motor de STT [16] seleccionado. En este caso, se utilizó la función "recognizer_google" para utilizar el motor STT [16] de google. A esta función se le enviaron como parámetros el audio a reconocer y el idioma en el que este se va a reconocer. En el caso de este motor, el código para identificar el idioma español es "es-us". Para mostrar el resultado del reconocimiento en pantalla se decodificó la cadena de texto y se convirtió a una variable de tipo **unicode**.

Finalmente, se aplicó programación defensiva para evitar que el programa termine espontáneamente debido a algún error. Se crearon excepciones para los siguientes casos:

- Excepción de audio - Cuando no fue posible descifrar el audio.
- Excepción de conexión - Cuando no fue posible conectarse a google (o al motor que se este utilizando). Generalmente, esta excepción se da cuando no existe una conexión a Internet.
- Excepción de tiempo - Cuando el tiempo de espera ha sido excedido.

Para la última excepción, se definió un tiempo de espera. Este se envió como parámetro al llamar a la función "listen".

8.2. Resultados

```
Esperando entrada de audio...
Entrada de audio recibida, reconociendo palabras...

Usted dijo Hola
```

Figura 10: Salida en consola para la prueba de STT

```
Ajustando al ruido del ambiente...
El umbral del sensibilidad minimo debe ser de 607.8
Umbral de sensibilidad setteado a 1607.8
```

Figura 11: Umbral de sensibilidad definido para la prueba de STT

```
Esperando entrada de audio...
Entrada de audio recibida, reconociendo palabras...

No fue posible descifrar palabras en el audio
```

Figura 12: Excepción de audio lanzada

```
Esperando entrada de audio...
Entrada de audio recibida, reconociendo palabras...

No fue posible establecer conexión con el servidor; recognition
connection failed: [Errno 11001] getaddrinfo failed
```

Figura 13: Excepción de conexión lanzada

```
Esperando entrada de audio...
Tiempo de espera excedido - no se detectó etrada de audio)
```

Figura 14: Excepción de tiempo lanzada

8.3. Discusión

El resultado, Figura 10, fue el texto generado por el programa (la palabra "hola") a partir del audio escuchado (Se dijo en voz alta la palabra "hola"). El audio fue obtenido mediante el micrófono de la computadora.

El umbral de sensibilidad o "energy threshold" se refiere a la cantidad de sonido que se considerará ruido. Mientras más pequeño el valor, más sensible es el programa. Esto significa que para lugares silenciosos debe utilizarse un umbral bajo y para lugares ruidosos uno más alto. Si esta cantidad se define más baja de lo necesario, el programa será demasiado sensible al ruido. Esto resulta en malas lecturas (pues la voz se confunde con el ruido) o lecturas muy lentas (debido a que al programa le toma más trabajo diferenciar la voz del ruido); en otras palabras, el programa considerará el ruido como voz y proporcionará resultados erróneos o le tomará más tiempo proporcionar un resultado correcto. Por otro lado, si esta cantidad se define más alta de lo necesario, la voz será considerada ruido y el programa no responderá a las entradas de voz ya que "no las escucha".

La función "adjust_for_ambient_noise" fue útil para la definición del umbral de sensibilidad. Esta necesita de unos segundos de silencio al inicio del programa. Así, detecta el nivel de ruido que existirá al recibir entradas de voz y ajusta el "energy_threshold" para detectar sonidos superiores al umbral de ruido. En algunos casos, este método no funciona del todo bien. Sin embargo, la cantidad que este le asigna al umbral de sensibilidad puede utilizarse como referencia para calibrar el sistema. En este caso, se detectó el nivel de ruido con dicha función y el umbral de sensibilidad se definió 1000 unidades más arriba para asegurar que se erradicara todo el ruido. El umbral mínimo durante esta prueba necesitaba ser aproximadamente de 607 y se definió en 1607 como se observa en la Figura 11

Se utilizó el motor STT [16] de google pues fue el que presentó los mejores resultados. La manera de evaluar la efectividad de los motores fue realizar múltiples intentos con cada uno y discriminar por exactitud y velocidad. Para cambiar de motor, basta con cambiar la función "recognize_google" por la correspondiente al motor deseado. Es necesario revisar la documentación de cada motor para identificar los idiomas con los que funciona y cómo configurar estos.

Para este caso, debido a que el idioma utilizado no fue inglés, fue necesario decodificar el texto obtenido para mostrarlo en pantalla. En el caso de versiones de Python 2, las cadenas de texto se codifican como bytes, por lo tanto se decodifican y se convierten a unicode. Para python 3 solo hace falta convertir a unicode.

En cuanto a las excepciones definidas, la primera es lanzada cuando el programa es incapaz de descifrar un audio. La Figura 12 muestra esta excepción. Para probarla, se aplaudió en lugar de hablar en el momento en el que el programa esperaba un audio. Por lo tanto, el programa detecta un audio; sin embargo, al momento de intentar convertirlo a texto lanza la excepción, pues no es posible convertir aplausos a palabras en texto. Lo mismo sucede para cualquier audio que el programa considere

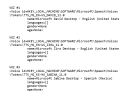
no puede convertirse a texto. La segunda excepción es lanzada cuando la función "recognize_google" no puede establecer conexión con el servidor, google en este caso. Para probarla, la computadora en la que se realizaron estas pruebas fue desconectada del internet y el programa lanzó la excepción que se muestra en la Figura 13. Por último, la tercera excepción es lanzada cuando, en el tiempo definido, el micrófono no recibe una entrada de audio superior al umbral de sensibilidad. Para ponerla a prueba, se mantuvo silencio mientras el programa esperaba una entrada y se obtuvo el resultado en la Figura 14.

9.1. Metodología

Desde el CMD, con el comando "Pip install pyttsx", se instaló la librería utilizada para la generación de la voz y se instanció en el código mediante el método "init". Antes de generar habla, fue necesario seleccionar la voz a utilizar. Para ello, se obtuvieron las voces que contiene la librería con la función "getProperty" del método "init". Cada una de las voces obtenidas poseía un "id", y por medio de este y de la función "setProperty" se selecciona una voz. Para reproducir una frase utilizando la voz seleccionada, bastó llamar a la función "say" y enviarle como parámetro la frase, seguida de la utilización de la función "runAndWait".

Tras seleccionar una voz, a esta se le realizaron cambios adicionales en velocidad también con la función "setProperty". La velocidad por defecto es de 200 palabras por minuto, pero esta se modificó a 150 palabras por minuto. El volumen, otra propiedad también manipulable mediante la función "setProperty", es un valor de tipo **float** entre 0 y 1. Este es 1 por defecto y se utilizó así en este programa.

9.2. Resultados



```
pyttsx> init
pyttsx> getProperty
pyttsx> setProperty
pyttsx> say
pyttsx> runAndWait
```

Figura 15: Información de las voces disponibles con pyttsx

9.3. Discusión

La Figura 15 muestra las tres voces obtenidas mediante la librería *Pyttssx*. Para cada voz, se observaron dos datos importantes: el lenguaje que maneja y su id, que es el identificador con el que puede utilizarse la voz mediante la función "setProperty". La voz seleccionada fue la de Microsoft Sabina Desktop, pues era la única de las tres que funcionaba en español. Finalmente, la velocidad o rate definido para la voz fue de 150, pues se consideró la voz sonaba más natural a esta velocidad que en la predeterminada.

10.1. Metodología

Para este apartado se hizo uso de las siguientes librerías:

1. *chatterbot*
2. *pyttsx*
3. *speech_recognition*
4. *re*
5. *unicodedata*
6. *random*
7. *sys*

Básicamente, se creó un **chatbot** y se le incorporaron los módulos de reconocimiento de voz y generación del habla con algunas funciones más. La librería *chatterbot* facilitó la creación del chatbot.

Para crear uno, se creó una instancia del método "Chatbot". Esta instancia fue después entrenada con una base de datos definida previamente. En este caso, se entrenó con diferentes archivos de texto. Estos archivos contenían listas de conversación que el chatbot puede seguir. El entrenamiento se llevó a cabo seleccionando la técnica "ListTrainer" mediante la función "set_trainer" y luego cargando cada archivo

de entrenamiento con la función "train". Cabe mencionar que "ListTrainer" es para entrenar con listas de datos; por lo tanto, fue necesario abrir desde el programa los archivos de texto y guardar cada línea en una lista mediante la función "readlines". Este entrenamiento se llevó a cabo únicamente la primera vez que se corrió el programa, pues para las siguientes corridas el programa ya conoce la base de datos.

Para utilizar el chatbot, se entró en un ciclo *while* donde se espera una entrada continuamente y al recibirla se proporciona una respuesta. Con texto, esto se resume a una entrada del teclado que se envía al chatbot entrenado por medio de la función "get_response", quien con esta misma función devuelve una respuesta, la cual es impresa en pantalla.

Con el chatbot creado, se incorporó el módulo de reconocimiento de voz STT [16]. La entrada del teclado se sustituyó por el resultado del reconocimiento de voz (el cual es texto). A su vez, la respuesta del chatbot se envió como parámetro al código para generación del habla TTS [17]. Toda respuesta del chatbot que ingresaba al módulo TTS debió ser ajustada primero. Se ignoraron todos los caracteres que no podían ser codificados en formato utf-8 a excepción de la letra ñ, la cual se sustituyó por el sonido "ni". Se ajustaron también de esta misma forma todos los textos obtenidos a partir del reconocimiento de voz.

También, se le atribuyó al chatbot la facultad de iniciar conversaciones por su cuenta, pues a este punto solo era capaz de responder cuando el usuario se dirigía a él. Para hacer posible esto, se hace uso de la excepción que se lanza al vencerse el tiempo de espera. Se agrega en ella una selección arbitraria (mediante el método "random") de una lista con frases útiles para iniciar una conversación o comentarios simpáticos. Así, si el usuario no interactúa con el chatbot, este comienza la interacción. Además, se hizo uso de la excepción lanzada cuando el audio no puede descifrarse. Al lanzarse esta excepción, el programa muestra en pantalla un mensaje para el usuario donde se especifica que el chatbot no fue capaz de entenderle o escucharle. Adicionalmente, se le configuró una breve introducción con el usuario. Esta consistió en un par de frases predefinidas donde el chatbot se presenta y solicita al usuario escribir su nombre.

Finalmente, como complemento, mediante comunicación serial se creó un vínculo con el módulo de control del movimiento de motores [5]. De esto se habla en la siguiente sección, pues no es un aspecto crucial en este módulo de conversación; mas es un aspecto de suma importancia para la implementación de este módulo en animatrónica.

10.2. Resultados

```
Tesoro:
'Hola, cómo te llamas?'
'Porfavor escribeme tu nombre'
Nombre:

Belen

Tesoro:
'Mucho gusto, Belen'
'Mi nombre es Tesoro'
------(Preparando...)-----
------(Energy threshold minimo 3347.18717647)-----
------(Energy threshold setteado a 4347.18717647)-----
```

Figura 16: Introducción al programa de conversación

```
...
Belen:
Buenos días Cómo estás
Tesoro:
'- Estoy bien, ¿y tú?'
...
Belen:
Yo también estoy bien
Tesoro:
'- Que bueno.'
...
Belen:
puedo ayudarte en algo
Tesoro:
'- Sí, tengo una pregunta.'
...
Belen:
cuál es tu pregunta
Tesoro:
'- "¿Puedo pedir prestada una taza de azúcar?"'
...
Belen:
lo siento no tengo
Tesoro:
'- Gracias de todas formas.'
...
Belen:
Y cuál es tu color favorito
Tesoro:
'- Azul'
...
Belen:
y te gusta la música
Tesoro:
'me gusta bailar'
```

Figura 17: Primera parte de una conversación con el chatbot

```

.....
'me gusta bailar'
...
Belen:
qué te gusta bailar
Tesoro:
'no se'
...
Belen:
Eres un robot
Tesoro:
'- Sí, lo soy.'
...
Belen:
adiós
Tesoro:
'que es eso'
...
Belen:
significa que me voy
Tesoro:
'- Significa que sólo se vive una vez. ¿Dónde has oído eso?'
...
iniciando conversacion
' - No hay problema.
,

```

Figura 18: Segunda parte de una conversación con el chatbot

```

iniciando conversacion
' - Complejo es mejor que complicado.
,
...
Belen:
Y de dónde sacas eso
Tesoro:
'- Bello es mejor que feo.'
...
Belen:
dónde has escuchado eso
Tesoro:
'Que cosa'
...
Belen:
tus dichos
Tesoro:
'a mi me gusta la musica'
...
iniciando conversacion
' - Plano es mejor que anidado.
,
...
Belen:
Tienes muchos dichos

```

Figura 19: Chatbot iniciando la conversación con el usuario

```

Tesoro:
'- "¿Qué quieres saber?"'
...
Belen:
------(Chatbot no te escucho)-----
...
Belen:
que sí eres una computadora
Tesoro:
'- Sí, lo soy.'
...
iniciando conversacion
' - Sí, tengo una pregunta.
,

```

Figura 20: Comportamiento del programa cuando la entrada de audio no pudo ser descifrada

10.3. Discusión

No es necesario ingresar un dato que se encuentre explícitamente en la base de datos, pues *chatterbot* busca en la base de datos la opción más parecida.

El acople del que se habla en la metodología fue necesario debido a que el módulo TTS [17] no es capaz de procesar caracteres como tildes, signos de apertura tanto de exclamación como de interrogación, ni la letra ñ, entre otros caracteres propios del idioma español. Fue necesario ajustar también todos los textos obtenidos del módulo de reconocimiento de voz, pues debido a que el chatbot aprende de las conversaciones con el usuario, estas entradas también son guardadas en su base de datos de posibles respuestas y el chatbot podría requerir pronunciarlas (es decir, hacer uso del módulo TTS) en algún momento.

El nombre del usuario es la única entrada por teclado. Esto para asegurar que el chatbot lo pronuncie correctamente, pues puede que no todos los nombres se encuentren en la base de datos de google de reconocimiento de voz.

La Figura 16 muestra las frases con las que inicia el programa de conversación. En esta prueba, el chatbot fue llamado Tesoro. Puede observarse acá también la calibración del umbral de sensibilidad. Las frases que Tesoro pronunció mediante el módulo TTS [17] se encuentran entre comillas. Las Figuras 17 y 18 muestran una conversación continua con Tesoro, se muestran también las entradas de audio del usuario ya convertidas a texto mediante el módulo de reconocimiento de voz. En la Figura 19 puede observarse como Tesoro inicia una conversación por su cuenta cuando no recibe interacción por parte del usuario. Por último, la Figura 20 muestra el comportamiento de Tesoro cuando no fue capaz de descifrar el audio.

Codificación y transmisión de generadores de movimiento

11.1. Metodología

En conjunto con el encargado del módulo de control del movimiento de motores [5], se definieron algunas palabras que podrían tener un gesto o un ademán relacionado. Estas fueron definidas no solo en base al significado de las palabras sino también considerando la posibilidad mecánica del animatrónico. Además, se definieron banderas en común para la ejecución de este proceso. A estas palabras se les llamó "generadoras del movimiento".

Además de contar con frases específicas que generan un movimiento específico, se definieron banderas para indicar que el chatbot iniciará a pronunciar una frase, que terminará de pronunciar la frase y si la frase que esta por pronunciar representa alguna emoción. Las banderas de inicio y fin de conversación son enviadas cada vez que el sistema de conversación hace uso del motor TTS [17].

Posteriormente, se creó un archivo de texto conteniendo las palabras definidas y su respectiva bandera. En el programa creado para la emulación de la conversación, se importó el módulo *serial* de la librería *pyserial* y se abrió una comunicación serial en el puerto deseado con una velocidad de transmisión (*baudrate*) de 9600. Se abrió el archivo de texto y se guardó en una lista. Mediante la función "contains" se revisó si la respuesta del chatbot contenía alguna de las palabras definidas como generadoras de movimiento en el archivo de texto. De contenerla, se envió mediante comunicación serial la bandera correspondiente a la frase o palabra.

11.2. Resultados

Desencadenante	Valor
Finaliza motor TTS	0
Inicia motor TTS	1
"Hola", "Buenos días", "Saludos", "Adiós"	2
"Derecha"	3
"Tengo una pregunta"	4
"Izquierda"	5
"Sí"	6
"No"	7
"Qué", "Quién", "Cómo", "Cuándo", "Dónde", "Cuál", "Cuánto", "Cuántos"	8
"No sé", "No entiendo"	9
"Máscara"	10
"Pistola"	11
"Oído"	12
"Música"	13
"Bueno", "Ok", "Fino"	14
"Paz", "Dos"	15
Frase sin emoción	16
Frase denotando felicidad	17
Frase denotando tristeza	18
Frase denotando enojo	19

Cuadro 1: Desencadenantes de un movimiento con su respectiva bandera

11.3. Discusión

Se utilizó una velocidad de transmisión o *baudrate* de 9600 debido a que con este valor trabaja el módulo de movimiento [5]. El Cuadro 1 muestra todas las palabras y/o acciones que generan un movimiento, cada una con su correspondiente bandera. El módulo de movimiento recibe estas banderas y mueve los motores correspondientes. En el Cuadro 20 en los anexos, pueden encontrarse las acciones que realiza el módulo de movimiento al recibir cada una de las banderas.

Para hacer posible el movimiento de la boca del animatrónico, se definieron dos banderas. Una se envía para denotar que el chatbot está por hablar y otra para indicar que ha terminado de hablar. Con estas banderas, el módulo de movimiento [5] emula el movimiento de la boca al hablar.

Finalmente, ya que el animatrónico en el que se implementará el sistema es capaz de mostrar expresiones faciales de alegría, enojo y tristeza, se clasificaron algunas frases dentro de estas emociones y también se les asignaron banderas. Fue también necesaria la creación de una bandera correspondiente a la expresión neutral para indicarle al animatrónico que es momento de terminar la expresión facial requerida.

12.1. Metodología

Para obtener grabaciones desde el micrófono de la computadora para después analizarlas, es necesario contar con las dos librerías siguientes: *Pyaudio* y *Wave*.

Primero, se definieron los parámetros de audio de: formato, número de canales, frecuencia de muestreo, y la cantidad de muestras (*frames*) en el buffer. Respectivamente, se definieron como 16 bits (mediante la función "pyaudio.paInt16"), 1 canal (sonido mono), 44100 Hz y 1024 muestras. Adicionalmente, se definió el tiempo de grabación deseado y el nombre del archivo WAV donde se guardaría la grabación. No es necesario que este archivo exista, si existe se sobrescribirá y si no se creará.

Se creó una instancia de *Pyaudio* por medio del método "PyAudio". Luego, se enviaron los parámetros definidos inicialmente a la función "open" de esta instancia. Con ayuda de un ciclo *for*, se realizó la grabación y los datos se guardaron en una lista. La grabación debe cerrarse al terminar al igual que debe indicársele a la instancia de *pyaudio* que la grabación ha terminado.

Se creó un archivo WAV con los datos. Para ello, se abrió el archivo definido previamente y se le definieron los mismos parámetros que para la grabación realizada. Configurados dichos parámetros, se escribió el archivo WAV con la lista que contenía los datos de audio. Por último debe cerrarse el archivo WAV.

12.1.1. Análisis de intensidad

Se necesitan las siguientes librerías de Python:

1. *scipy*
2. *matplotlib*
3. *numpy*

Debe abrirse el archivo WAV a analizar. La función "wavfile.read" del método "io" de la librería *scipy* permite abrir un archivo de audio y extraer los dos parámetros fundamentales que lo describen: la frecuencia de muestreo y los datos en sí. Para obtener los tiempos en los que se tomó cada dato, se llevó a cabo una división de los datos entre la frecuencia de muestreo y esta cantidad se envió como parámetro a la función "arange" de *numpy*.

Se aplicó a la señal un filtro pasabanda tipo Butterworth de 6to orden para obtener únicamente la voz. Se eligieron 200 y 3500 como frecuencias de corte. Este filtrado puede realizarse mediante las funciones "butter" y "lfilter" del método "signal" en la librería *scipy*.

Posteriormente, debido a que el sonido es de un solo canal (mono), la intensidad ya filtrada pudo graficarse fácilmente mediante la función "fill_between" del método "pyplot" de la librería *matplotlib*. A esta función se le enviaron como parámetros los tiempos y los datos de audio.

Para iniciar con el análisis de patrones de emoción, se solicitó a varios sujetos que leyeran el siguiente texto en una ventana de 15 segundos mientras personificaban cada línea con la emoción que se mencionaba:

Esta es mi voz neutral,
El enojo tiende a presentar mayor intensidad,
El aburrimiento es así,
La alegría o sorpresa también es intensa,
Por último, la tristeza también es suave.

Así, se obtuvo una gráfica por sujeto. Cada gráfica con 5 entonaciones diferentes para poder ser comparadas.

12.1.2. Análisis de frecuencia

Se necesitan las siguientes librerías de Python:

1. *scipy*

2. *matplotlib*

3. *numpy*

Al igual que para la intensidad, es necesario obtener primero los datos y la frecuencia de muestreo; estos se obtuvieron de la misma manera mencionada en el apartado de análisis de intensidad. Debido a que el audio es mono, bastó con calcular el número de datos, los tiempos y el período (inversa de la frecuencia de muestreo).

Se obtuvo el valor absoluto del espectro de Fourier para los datos, mediante el método "fft" de la librería *scipy*. Este resultado se normalizó multiplicando por el número de muestras y dividiendo dentro de dos. Las frecuencias correspondientes a cada dato del espectro de Fourier se obtuvieron también con *scipy*, mediante la función "fftfreq" del método "fftpack". Las frecuencias también fueron normalizadas de la misma forma, multiplicando por el número de muestras y dividiendo dentro de dos.

Por último, solo hace falta visualizar los datos. Se utilizó la función "plot" del método "pyplot" para ello.

Para comprobar el funcionamiento del programa, se ingresaron archivos WAV que contenían únicamente un tono realizado con una flauta. La gráfica debía mostrar la frecuencia correspondiente a la nota tocada.

12.2. Resultados

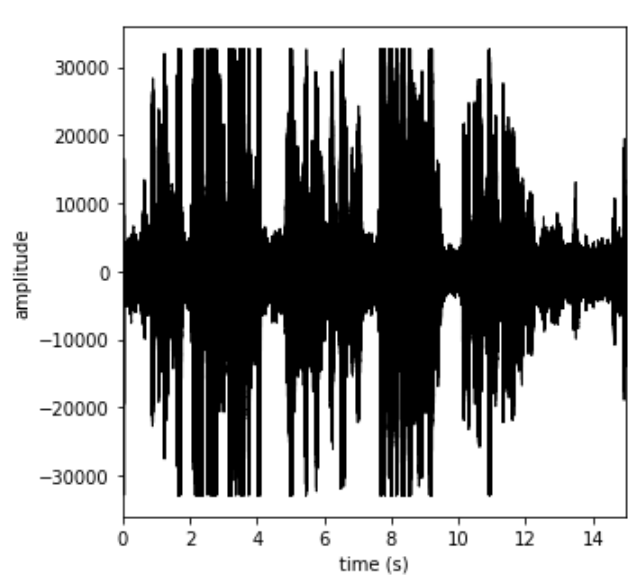


Figura 21: Prueba de intensidad sujeto 1

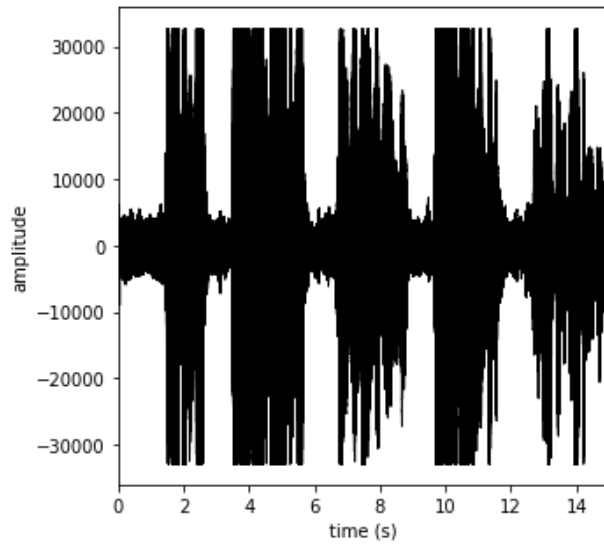


Figura 22: Prueba de intensidad sujeto 2

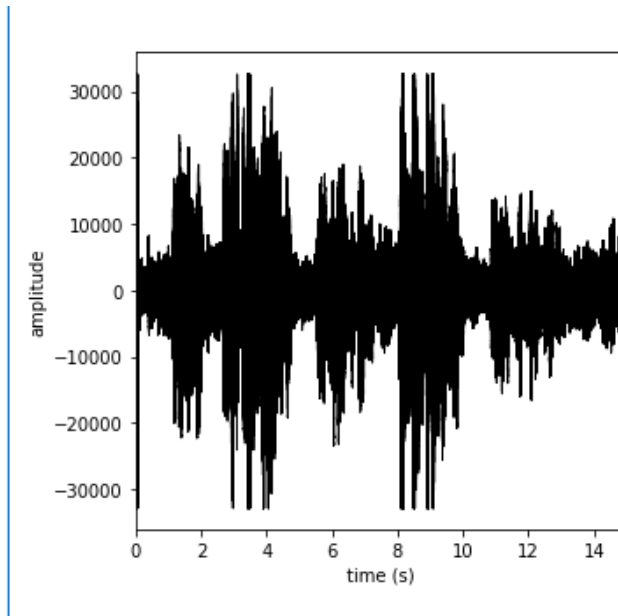


Figura 23: Prueba de intensidad sujeto 3

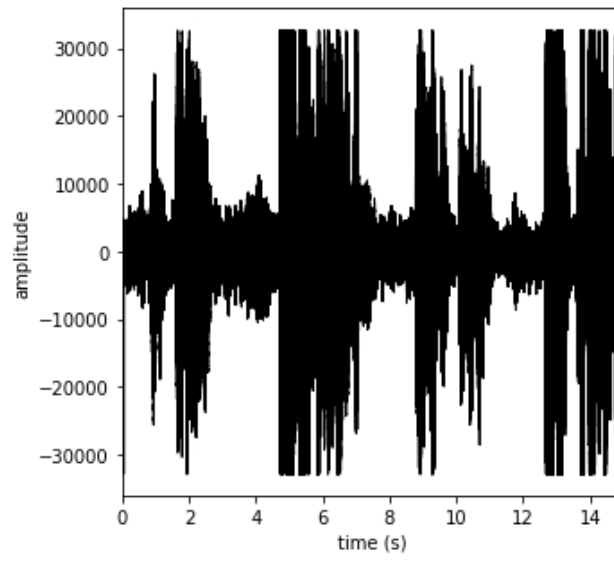


Figura 24: Prueba de intensidad sujeto 4

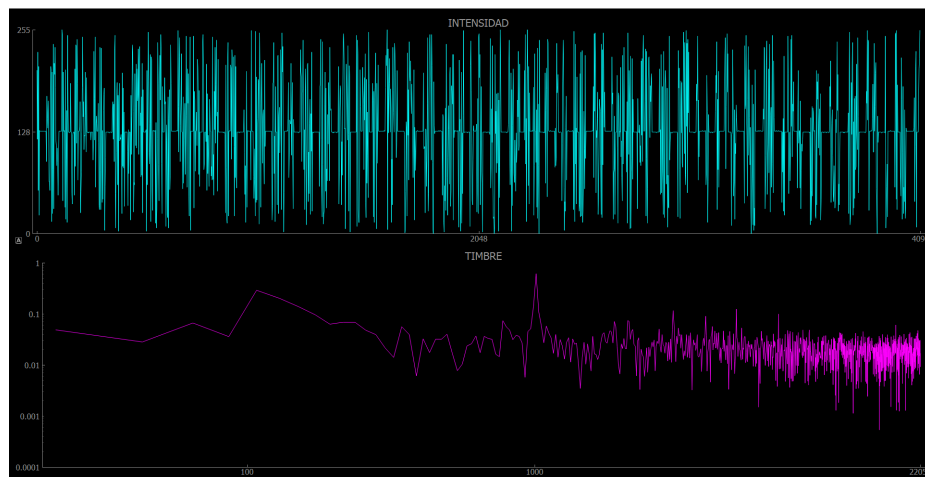


Figura 25: Amplitud y frecuencia nota Si

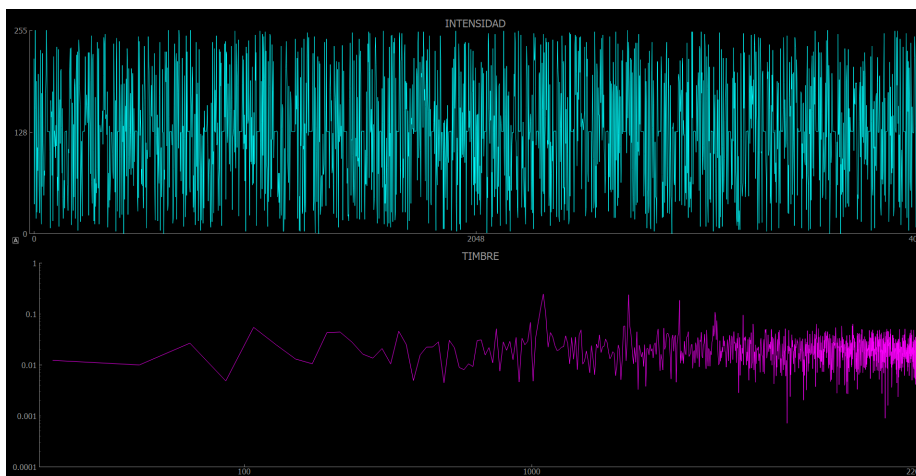


Figura 26: Amplitud y frecuencia nota Do

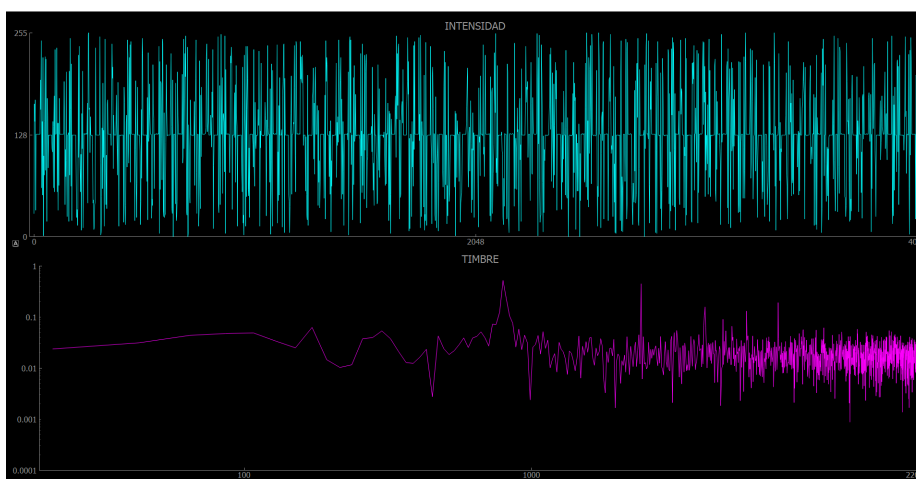


Figura 27: Amplitud y frecuencia nota Sol

12.3. Discusión

Antes de analizar audio, es necesario contar con el audio como tal; es decir, obtenerlo. Idealmente, se cuenta con grabaciones en formato WAV (pues este formato es fácilmente manipulable). De lo contrario, el primer paso para analizar audio es realizar grabaciones y guardarlas en formato WAV. Dicho procedimiento podría realizarse con cualquier editor de audio. Sin embargo, si en el futuro se busca realizar un análisis de audio en tiempo real, lo más conveniente es lograr generar un archivo WAV desde Python, que es el lenguaje utilizado en este trabajo.

Puesto que la señal a analizar era la voz, con el fin de disminuir lo mejor posible el ruido se le aplicó un filtro pasabanda butterworth de sexto orden a la señal. Se eligieron frecuencias de corte de 200 y 3500 Hz debido a que la voz humana está comprendida generalmente entre los 300 y 3400 Hz.

Para realizar un análisis de intensidad, es necesario mostrar estos datos en una gráfica. Así, pueden compararse visualmente las muestras de audio. Debido a esto, no fue necesario realizar un análisis en tiempo real para esta etapa; pues era más conveniente observar cada porción de datos.

Las Figuras 21 a 24 muestran los resultados de intensidad de cuatro sujetos diferentes. Observando únicamente estos cuatro resultados puede deducirse que no existe una referencia al evaluar intensidad en las voces de las personas: para cada persona en cada situación la referencia es diferente. En otras palabras, no puede decirse que una emoción presenta una intensidad en algún rango de decibeles. Por ejemplo, el sujeto 3, en la Figura 23 fue quien presentó los resultados más claros. Sujetos como el 1 y el 2 poseen una voz muy fuerte (su escala en decibeles es alta) y es difícil discernir sus resultados en la escala utilizada como estándar para todos.

Aunque hasta el momento no pudo encontrarse la relación numérica, si pudo observarse la relación entre intensidad y emoción. Utilizando la Figura 23, puede observarse que el sujeto comenzó a hablar con su voz neutral aproximadamente tras transcurrir un segundo. Tras tres segundos inició con la emoción de enojo y se observa un aumento en la amplitud. Seguido a esto vino el aburrimiento y se observa una disminución en la amplitud. Esta amplitud es aún más pequeña que la de la voz neutral. Posteriormente, la alegría elevó de nuevo la intensidad de la onda. Al final, la tristeza muestra la amplitud más pequeña. Este patrón de aumentos y disminuciones en amplitud según la emoción tiende a presentarse en todos los sujetos. Cabe mencionar que el sujeto 4 fue el único que no terminó de personificar todas las emociones en 15 segundos, la tristeza no tuvo lugar en el audio.

Para las pruebas de frecuencia sí se utilizó un programa de lectura continua, aunque pudo haberse utilizado la primera versión (no continua) creada sin problemas. La primera versión trabajaba de la misma forma que el programa utilizado para intensidad: graba una porción de datos y despliega una gráfica con el espectro en frecuencia. Se utilizó la versión continua únicamente porque era más fácil realizar las pruebas con este programa; bastaba con correrlo y podía observarse una gráfica en tiempo real del espectro en frecuencia.

A diferencia de las pruebas de intensidad (las cuales se realizaron para evidenciar el comportamiento en intensidad de las diferentes emociones), las pruebas en frecuencia tenían como objetivo únicamente corroborar el funcionamiento del programa creado para obtener las frecuencias de un audio, y no evidenciar el comportamiento en frecuencia de las emociones en la voz. No se realizaron pruebas con emociones pues visualmente no son relevantes. No es posible observar un espectro de Fourier (entiéndase, frecuencias contra amplitud) y realizar un análisis sencillo con respecto a la identificación de patrones relevantes en el análisis de emociones en la voz. Esto debido a que la voz cuenta con un sinfín de armónicos. Es decir, existen muchas frecuencias en juego en un mismo instante. Por lo tanto, el análisis frecuencial depende totalmente de un análisis estadístico del mismo para poder concluir sobre él. Este análisis estadístico (tanto de la intensidad como de la frecuencia) se trata en el apartado de Reconocimiento de patrones de emoción. Por todo lo expuesto ante-

riormente, para verificar el funcionamiento del programa que identifica frecuencias se trabajó con una flauta dulce, pues los instrumentos musicales producen notas con frecuencias definidas que permiten evaluar el algoritmo.

Debido a que para las pruebas en frecuencia se trabajó con una flauta dulce, no fue necesario aplicar el filtro que se aplica en las pruebas de voz. La Figura 25 corresponde a una nota Si, la lectura en frecuencia muestra un valor alrededor de los 1000Hz. La Figura 26, correspondiente a un Do, muestra una lectura muy parecida a la nota anterior pero un poco mayor. Finalmente, la Figura 27 muestra una nota Sol cuya frecuencia es significativamente diferente a las otras dos, siendo menor. Puede corroborarse la certeza de estos resultados observando las frecuencias para el Si de quinta octava, el Do de sexta octava y el Sol de quinta octava de la Figura 28.

Con el fin de sondear las características útiles para el reconocimiento de emociones, se realizaron programas para analizar audio en intensidad y frecuencia. Se analizará también la duración, pero este parámetro es más simple de obtener y puede derivarse del análisis de intensidad o de frecuencia. El timbre fue descartado debido a que, en cuanto a la voz, se consideró estaba contenido en el espectro de frecuencias. Por último, la espacialidad fue descartada debido a que se consideró irrelevante en este estudio, pues en cualquier lugar en el planeta donde se intente replicar este estudio la espacialidad será la misma. Sumado a esto, a pesar de que la respiración también es un parámetro importante en la detección de emociones en la voz, esta se descartó debido a que no es posible obtenerla en una grabación. Para obtenerla, sería necesario contar con un monitoreo del ritmo cardíaco del usuario.

Frecuencia (en Hertzios) de las notas musicales

x

		0	1	2	3	4	5	6	7	8
n=1	do		32.7	65.41	130.81	261.63	523.25	1046.50	2093.00	4186.01
n=2	do#		34.65	69.30	138.59	277.18	554.37	1108.73	2217.46	4434.92
n=3	re		36.71	73.42	146.83	293.66	587.33	1174.66	2349.32	4698.64
n=4	re#		38.89	77.78	155.56	311.13	622.25	1244.51	2489.02	4978.03
n=5	mi		41.2	82.41	164.81	329.63	659.26	1318.51	2637.02	5274.04
n=6	fa	21.826	43.65	87.31	174.61	349.23	698.46	1396.91	2793.83	5587.65
n=7	fa#	23.125	46.25	92.50	185.00	369.99	739.99	1479.98	2959.96	5919.91
n=8	sol	24.50	49.00	98.00	196.00	392.00	783.99	1567.98	3135.96	6271.93
n=9	sol#	25.96	51.91	103.83	207.65	415.30	830.61	1661.22	3322.44	
n=10	la	27.50	55.00	110.00	220.00	440.00	880.00	1760.00	3520.00	
n=11	la#	29.14	58.27	116.54	233.08	466.00	932.33	1864.66	3729.31	
n=12	si	30.87	61.74	123.47	246.94	493.88	987.77	1975.53	3951.07	

Figura 28: Frecuencias de las notas musicales

13.1. Metodología

13.1.1. Parametrización de los archivos de audio

Comparar directamente un archivo de audio con otro para determinar si tienen relación no es adecuado, pues dos archivos pueden ser muy diferentes y ambos representar alegría. Es decir, tomar los datos que describen un archivo de audio en intensidad no es una opción pertinente para este estudio. Por otro lado, la estadística descriptiva presenta una mejor alternativa: describir cada archivo de audio por la distribución de sus datos, tanto en intensidad como en frecuencia para así poder comparar con otros audios.

Se obtuvieron del archivo de audio los siguientes datos:

1. Sus intensidades en el tiempo (los datos obtenidos del archivo WAV).
2. Los tiempos en que se dan dichas intensidades (relación entre la frecuencia de muestreo y los datos, ambos obtenidos del archivo WAV).
3. La duración del archivo de audio (corresponde al último tiempo registrado en los tiempos en que se dan los distintos valores para la intensidad).
4. Las amplitudes obtenidas al aplicarle la transformada rápida de Fourier.
5. Las frecuencias correspondientes a las amplitudes obtenidas de la transformada rápida de Fourier.

Se les aplicó estadística descriptiva a las intensidades y a las amplitudes, así como se obtuvieron las frecuencias correspondientes a amplitudes importantes. El método "stats" de la librería *scipy* y la librería *numpy* son útiles para los cálculos estadísticos. Para cada archivo de audio se definieron entonces los siguientes 25 parámetros:

1. Conteo de datos de la intensidad (es decir, número de datos obtenidos en intensidad)
2. Intensidad mínima
3. Intensidad máxima
4. Media de la intensidad
5. Mediana de la intensidad
6. Varianza de la intensidad
7. Desviación estándar de la intensidad
8. Oblicuidad de la intensidad
9. Curtosis de la intensidad
10. Amplitud máxima en el dominio de frecuencia
11. Amplitud mínima en el dominio de frecuencia
12. Rango de amplitudes en el dominio de frecuencia
13. Frecuencia con la máxima amplitud
14. Frecuencia con la mínima amplitud
15. Conteo de datos de frecuencias (es decir, número de datos obtenidos en frecuencias)
16. Frecuencia más alta alcanzada
17. Media de las amplitudes en el dominio de la frecuencia
18. Mediana de las amplitudes en el dominio de la frecuencia
19. Varianza de las amplitudes en el dominio de la frecuencia
20. Desviación las amplitudes en el dominio de la frecuencia
21. Oblicuidad las amplitudes en el dominio de la frecuencia
22. Curtosis de las amplitudes en el dominio de la frecuencia
23. Velocidad (Básicamente duración en segundos del audio, último dato de los instantes de tiempo obtenidos)

24. Sexo (Dato booleano donde 0 representa mujer y 1 representa hombre)
25. Emoción (Este campo solo se llena para los archivos de la base de datos a utilizar, para las grabaciones de prueba se omite pues es el campo en evaluación. Este es el único dato no numérico, pues se trata de una cadena de texto)

Tras el cálculo de estas cantidades, se unen todos en una sola cadena de texto, donde cada parámetro está separado por comas.

13.1.2. Construcción de la base de datos

Se obtuvo la base de datos de emociones [48] mediante la Asociación Europea de Recursos Lingüísticos (ELRA, por sus siglas en inglés) [49]. Los archivos de audio fueron convertidos a formato WAV mediante el software Audacity, para poder ser manipulados en Python. Se tomó una muestra de 20 archivos de audio por cada una de las seis emociones (enojo, disgusto, miedo, alegría, sorpresa y tristeza), esto para ambos sexos. Se trabajó con las 10 primeras frases de la base de datos:

- Con estoico respeto a la justicia adyacente guardó sus flechas
- Fue inyectado en el abdomen y en una pierna
- La tensión volvió a aumentar el domingo
- Cuando todavía eran baratos el vodka y el caviar
- Sino el país mental de un patriota enloquecido
- Abría sus puertas a estos flacos alumnos afroamericanos
- Palestina seguía en estado paupérrimo después de su duunvirato
- El presidente de la federación portuguesa de fútbol
- Bajo los efectos de la hipnosis tocaba la guzla perfectamente
- El oftalmólogo dijo que le hizo entrar en un proceso subfebril

Se necesitó de las librerías *os* y *stat* poder recorrer las carpetas en la computadora donde están guardados los archivos de audio. Así, se recorre cada carpeta y se obtiene cada uno de los 239 archivos (eran 240 pero uno de los archivos se encontraba dañado, por lo que se omitió). A cada archivo se le aplica el filtro para la voz y se obtienen de él los 25 parámetros mencionados anteriormente. Esta información es guardada línea a línea en un archivo de texto, donde cada línea representa un archivo de audio. Y se obtiene así una base de datos útil en aplicaciones de aprendizaje de máquina.

13.2. Resultados

```
1 | 69632,-9849.0,17624.0,68.6596679688,60.0,2745233.95325,1656.87475485,1.50172925361,12.6759594936,6276947.155960179,40.
82643862161482,6276906.329521557,549.4025735294117,7928.7683823529405,34816,7999.77022059,245362.206562,149007.944209,1.
31610029559e+11,362780.966369,4.82031064888,33.6640863868,4.3519375,0,ENOJO
2 | 51712,-8268.0,18965.0,68.7478341584,60.0,2135802.04026,1461.43834638,1.75440019036,13.1364571276,4061997.2417910784,40.
49544216588911,4061956.7463489124,195.2351485148515,7606.126227623762,25856,7999.69059406,165426.848575,71037.
5205243,83569362030,2,289083.659224,4.64100156613,30.5267095087,3.2319375,0,ENOJO
3 | 49664,-10676.0,22629.0,67.9393524485,56.0,3473850.79059,1863.82692077,1.53044558118,12.9124718226,4742503.256514521,45.
508540063988704,4742457.747974457,197.16494845360822,7721.005154639174,24832,7999.67783505,231254.624982,140640.131448,1.
19505097856e+11,345695.093769,4.58130758043,28.6059340775,3.1039375,0,ENOJO
4 | 54272,-11444.0,22590.0,69.0499889446,58.0,2641946.56407,1625.40658423,2.76646976941,24.3098082761,4121785.6619020407,53.
520152625200915,4121732.1417494155,193.10141509433964,7813.67924528302,27136,7999.70518868,213108.659677,95840.
4625926,98485950138.6,313824.712441,3.28599417737,15.7379895873,3.3919375,0,ENOJO
5 | 55296,-15713.0,31011.0,68.6403537326,60.0,4265627.61449,2065.33958818,1.89239580385,20.4637136962,5424505.678429549,46.
01540220014709,5424459.663027349,211.8055555555557,7913.4837962962965,27648,7999.71064815,271428.485491,140470.50177,1.
62719775817e+11,403385.393659,4.12755834859,24.8776718165,3.4559375,0,ENOJO
6 | 68608,-9523.0,17839.0,68.9623076026,60.0,1297457.03336,1139.05971457,2.43080584227,19.9071596475,4731366.0,50.
91834470871387,4731315.00165529,0.0,7646.222014023373,34304,7999.76679104,184003.224491,117460.512901,55811317316.
0,236244.100015,4.141884104896,28.111617299,4.2879375,0,ENOJO
7 | 72192,-6112.0,15050.0,68.5173149379,60.0,824432.566938,907.98269088,2.14782399258,18.1025018152,4946402.0,36.
661844290441245,4946365.338155709,0.0,7920.212765957446,36096,7999.77836879,143201.978102,92212.634444,39688457635.
5,199219.621613,4.86943549045,38.6689283989,4.5119375,0,ENOJO
8 | 56320,-10620.0,17206.0,71.9355823864,64.0,1715885.48577,1309.91812178,1.37949328304,12.6045531096,4051412.0,66.
06142779189904,4051345.938572208,0.0,7935.511363636364,28160,7999.71590909,189955.663291,128489.894386,61138397904.
5,247261.800334,4.49057857746,29.2979920324,3.5199375,0,ENOJO
9 | 61952,-15123.0,22366.0,69.0886815599,56.0,3604934.95801,1898.66662635,1.93969087017,17.2216583522,4280182.0,25.
1290079479792,4280156.870092052,0.0,7663.223140495868,30976,7999.74173554,288849.977585,155184.992364,1.
40490045113e+11,374820.016959,2.88743162791,11.7522121399,3.8719375,0,ENOJO
```

Figura 29: Fragmento de la base de datos codificada

13.3. Discusión

Para implementar aprendizaje de máquina, fue necesario traducir los archivos de audio a un formato que la computadora pueda interpretar mejor: números. Con el fin de implementar aprendizaje de máquina supervisado, es necesario contar con una base de datos de clasificación. En este caso, la base de datos debe contar con diferentes entradas de audio etiquetadas con la emoción que representan. La base de datos utilizada, cuenta con las grabaciones en español de un hombre y una mujer, cada uno repite cada una de las 184 frases con las 6 diferentes emociones y también variaciones del estilo neutral (las cuales se omitieron para este análisis, pues neutral no es una emoción). Debido a que estas grabaciones venían en formato L16, fue necesario convertirlas a WAV para su manipulación.

La Figura 29 muestra un fragmento de la base de datos creada, visualizada desde el editor de texto **Sublime Text**. Pueden observarse los 24 parámetros separados por comas, seguidos por la emoción en la que dicha entrada se clasifica. Se muestran las nueve primeras entradas de la base de datos.

Validación de la parametrización definida

14.1. Metodología

Fue necesario contar con la librería *pandas* y la librería *sklearn*. Mediante el método "read_csv" de la librería *pandas*, se obtuvo la información de la base de datos contenida en el archivo de texto. Se le enviaron como parámetros el archivo de texto y una lista con los nombres de cada uno de los 25 parámetros contenidos en el archivo de texto. Así se obtuvo el llamado *dataset* o conjunto de datos.

Se creó una instancia del método "tree" de la librería *sklearn*, utilizando la función "DecisionTreeClassifier". Este es el clasificador. Mediante la función "fit" del clasificador, se le indicó a este cuáles eran los parámetros de descripción y cuál era el parámetro de clasificación (el de clasificación era la emoción y el resto describían el archivo de audio).

Luego, se utilizó el método "export_graphviz" para extraer la información del clasificador como un archivo de texto. Se ingresó a la página web de *graphviz* [50], donde se encuentra una herramienta para graficar el árbol de decisiones. Se copió el contenido del archivo de texto con la información del clasificador generado en Python y se pegó en la herramienta. Se obtuvo así el árbol de decisiones gráfico.

Finalmente, se buscó qué variables fueron omitidas por el árbol de decisiones y se hizo un conteo de las ocurrencias de las que sí aparecían.

14.2. Resultados

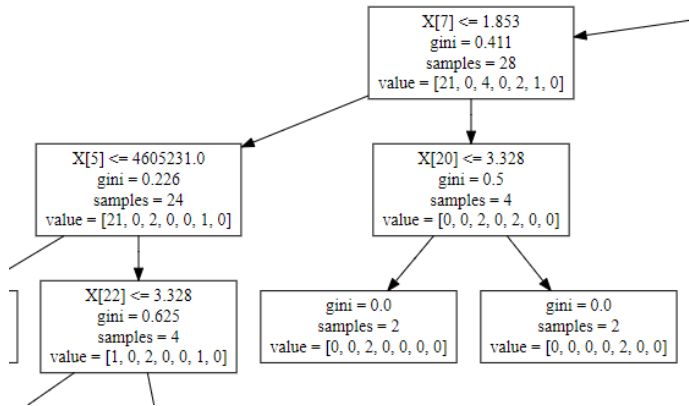


Figura 30: Fragmento del árbol de decisiones realizado con 24 parámetros.

ID	Nombre del parámetro	Número de ocurrencias
0	Conteo de datos de la intensidad	2
1	Intensidad mínima	5
2	Intensidad máxima	5
3	Media de la intensidad	3
4	Mediana de la intensidad	3
5	Varianza de la intensidad	4
6	Desviación estándar de la intensidad	3
7	Oblicuidad de la intensidad	7
8	Curtosis de la intensidad	7
9	Amplitud máxima en el dominio de frecuencia	2
10	Amplitud mínima en el dominio de frecuencia	3
11	Rango de amplitudes en el dominio de frecuencia	3
12	Frecuencia con la máxima amplitud	9
13	Frecuencia con la mínima amplitud	6
14	Conteo de datos de frecuencias	2
15	Frecuencia más alta	5
16	Media de las amplitudes en el dominio de la frecuencia	3
17	Mediana de las amplitudes en el dominio de la frecuencia	3
18	Varianza de las amplitudes en el dominio de la frecuencia	2
19	Desviación estándar de las amplitudes en el dominio de la frecuencia	1
20	Oblicuidad de las amplitudes en el dominio de la frecuencia	9
21	Curtosis de las amplitudes en el dominio de la frecuencia	5
22	Velocidad	3
23	Sexo	2

Cuadro 2: Ocurrencias de los 24 parámetros evaluados con el árbol de decisiones.

14.3. Discusión

Hasta el momento se cuenta con una base de datos funcional para evaluar la posibilidad de reconocer emociones en la voz. Sin embargo, es importante determinar si los 24 parámetros son relevantes (es necesario recordar que el parámetro número 25 es la clasificación de la emoción, por lo que no es necesario evaluar si es relevante o no). Por ello, se hizo uso del algoritmo de aprendizaje de maquina supervisado del Árbol de decisiones (CART), con el que puede observarse gráficamente el peso de cada variable. Por peso, se refiere a la cantidad de veces que esta aparece en el árbol de decisiones, lo cual se traduce a la cantidad de veces que el algoritmo recurrió a dicha variable para realizar una clasificación.

La Figura 30 muestra únicamente un fragmento del árbol de decisiones, pues este resultó ser muy grande. El fragmento cumple con el objetivo: demostrar cómo se evaluó el árbol en su totalidad. En este pueden observarse seis "hojas" del árbol de las 97 que este desplegó. Cada hoja contaba con cuatro datos:

- La lista *X* que contenía cada uno de los parámetros. El parámetro que se utilizó en cada hoja para clasificar puede identificarse por el índice que contiene *X*.
- El valor de gini que indica la desigualdad entre los datos que aún falta por clasificar. El 0 corresponde a una igualdad total y el 1 a una desigualdad total.
- La variable *samples* que indica el número de datos o muestras que se están evaluando en esa hoja.
- El arreglo *value* que indica a qué clasificación pertenecen los datos que se están evaluando en ese momento (en este caso, la emoción a la que pertenecen).

Así, para determinar las variables relevantes se realizó un conteo de las hojas en las que aparecían. El Cuadro 2 muestra las ocurrencias de cada variable. Las variables que no aparecen en el árbol de decisiones son las variables irrelevantes para la clasificación. En este caso, las 24 variables evaluadas aparecieron por lo que se concluyó que todas son útiles en el reconocimiento de emociones. Por otro lado, entre más ocurrencias presente una variable, mayor importancia tiene en la clasificación de emociones. Las variables más relevantes, con 9 ocurrencias cada una, resultaron ser la frecuencia con la amplitud máxima y la oblicuidad de las amplitudes en el dominio de la frecuencia. Estas seguidas por la oblicuidad de la intensidad y la curtosis de la intensidad, las cuales presentaron 7 ocurrencias cada una. La variable menos relevante resultó ser la desviación estándar de las amplitudes en el dominio de la frecuencia, pues fue la única en presentar una sola ocurrencia. Sin embargo, no pudo descartarse debido a que fue necesitada al menos una vez para clasificar emociones.

Evaluación de la probabilidad de reconocer emociones en la voz

15.1. Metodología

Son importantes tres librerías para este procedimiento:

1. *pandas*
2. *matplotlib*
3. *sklearn*

Como primer paso, se cargó el archivo de texto que contiene a la base de datos y se creó una lista con los encabezados de las columnas de dicha base de datos. Como se mencionó con anterioridad, se creó un conjunto de datos mediante la librería *pandas*.

Se separó la columna de clasificación (la que contiene la emoción, en este caso) del resto de los datos colocando ambas partes en dos variables diferentes y se procedió a entrenar al programa con estos datos mediante la función "model_selection.train_test_split" de la librería *sklearn*. Posteriormente, se creó un arreglo con los 6 algoritmos de aprendizaje de máquina supervisado a evaluar (LR, LDA, KNN, CART, NB y SVM).

Por medio de un ciclo *for*, se realizó la validación cruzada para cada uno de los métodos anteriores. De la validación cruzada, se obtiene la media y la desviación estándar de los resultados de cada algoritmo. Se determinó el mejor algoritmo.

Para el mejor algoritmo, se obtuvo su probabilidad de acierto, su matriz de confusión y su reporte de clasificación, mediante las funciones "accuracy_score", "confusion_matrix" y "classification_report" respectivamente. Del reporte de clasificación, se determinaron las tres combinaciones de tres emociones con mejor probabilidad de ser detectadas (las tres emociones con mejor precisión, las tres emociones con mejor llamado y las tres emociones con mejor puntuación F1) y, finalmente, se repitió todo el procedimiento tres veces, para cada una de las tres combinaciones con el fin de comparar cuál era la combinación que presentaba mejores resultados.

15.2. Resultados

Algoritmo	Probabilidad media de aciertos	Desviación estándar
LR	0.188421	0.074944
LDA	0.544211	0.149611
KNN	0.172632	0.087843
CART	0.460526	0.095065
NB	0.309737	0.099570
SVM	0.135789	0.066316

Cuadro 3: Comparación de algoritmos a seis emociones

Linear Discriminant Analysis (LDA)

Probabilidad de Acierto	0.4792
-------------------------	--------

Cuadro 4: Probabilidad de acierto a seis emociones del algoritmo con el mejor resultado

	Alegría	Disgusto	Enojo	Miedo	Sorpresa	Tristeza
Alegría	7	1	0	2	0	0
Disgusto	1	2	1	1	0	1
Enojo	3	1	4	1	2	0
Miedo	0	0	0	3	1	0
Sorpresa	2	3	0	1	4	0
Tristeza	0	0	0	4	0	3

Cuadro 5: Matriz de confusión entre las seis emociones, utilizando el algoritmo LDA

	Precisión	Llamada	Puntuación F1	Soporte
Alegría	0.54	0.70	0.61	10
Disgusto	0.29	0.33	0.31	6
Enojo	0.80	0.36	0.50	11
Miedo	0.25	0.75	0.38	4
Sorpresa	0.57	0.40	0.47	10
Tristeza	0.75	0.43	0.55	7
Promedio	0.58	0.48	0.49	48

Cuadro 6: Reporte de clasificación para seis emociones, utilizando el algoritmo LDA

Algoritmo	Probabilidad media de aciertos	Desviación estándar
LR	0.394444	0.120416
LDA	0.864444	0.084882
KNN	0.426667	0.065772
CART	0.656667	0.133837
NB	0.493333	0.159799
SVM	0.187778	0.093419

Cuadro 7: Comparación de algoritmos a tres emociones: enojo, sorpresa y tristeza

Linear Discriminant Analysis (LDA)

Probabilidad de Acierto	0.8333
-------------------------	--------

Cuadro 8: Probabilidad de acierto a tres emociones (enojo, sorpresa y tristeza) del algoritmo con el mejor resultado

	Enojo	Sorpresa	Tristeza
Enojo	6	1	0
Sorpresa	1	5	2
Tristeza	0	0	9

Cuadro 9: Matriz de confusión entre enojo, sorpresa y tristeza, utilizando el algoritmo LDA

	Precisión	Llamada	Puntuación F1	Soporte
Enojo	0.86	0.86	0.86	7
Sorpresa	0.83	0.62	0.71	8
Tristeza	0.82	1.00	0.90	9
Promedio	0.83	0.83	0.83	24

Cuadro 10: Reporte de clasificación para tres emociones: enojo, sorpresa y tristeza, utilizando el algoritmo LDA

Algoritmo	Probabilidad media de aciertos	Desviación estándar
LR	0.290000	0.138907
LDA	0.755556	0.115470
KNN	0.362222	0.122746
CART	0.787778	0.121254
NB	0.543333	0.204363
SVM	0.207778	0.104000

Cuadro 11: Comparación de algoritmos a tres emociones: alegría, miedo y tristeza

DecisionTreeClassifier (CART)

Probabilidad de Acierto	0.8333
-------------------------	--------

Cuadro 12: Probabilidad de acierto a tres emociones (alegría, miedo y tristeza) del algoritmo con el mejor resultado

	Alegría	Miedo	Tristeza
Alegría	7	1	0
Miedo	0	7	0
Tristeza	0	3	6

Cuadro 13: Matriz de confusión entre alegría, miedo y tristeza, utilizando el algoritmo LDA

	Precisión	Llamada	Puntuación F1	Soporte
Alegría	1.00	0.88	0.93	8
Miedo	0.64	1.00	0.78	7
Tristeza	1.00	0.67	0.80	9
Promedio	0.89	0.83	0.84	24

Cuadro 14: Reporte de clasificación para tres emociones: alegría, miedo y tristeza, utilizando el algoritmo LDA

Algoritmo	Probabilidad media de aciertos	Desviación estándar
LR	0.366667	0.135628
LDA	0.810000	0.108463
KNN	0.295556	0.127966
CART	0.698889	0.144568
NB	0.512222	0.194838
SVM	0.207778	0.1040006

Cuadro 15: Comparación de algoritmos a tres emociones: alegría, enojo y tristeza

Linear Discriminant Analysis (LDA)	
Probabilidad de Acierto	0.4792

Cuadro 16: Probabilidad de acierto a tres emociones (alegría, enojo y tristeza) del algoritmo con el mejor resultado

	Alegría	Enojo	Tristeza
Alegría	6	2	0
Enojo	5	2	0
Tristeza	0	0	9

Cuadro 17: Matriz de confusión entre alegría, enojo y tristeza, utilizando el algoritmo LDA

	Precisión	Llamada	Puntuación F1	Soporte
Alegría	0.55	0.75	0.63	8
Enojo	0.50	0.29	0.36	7
Tristeza	1.00	1.00	1.00	9
Promedio	0.70	0.71	0.69	24

Cuadro 18: Reporte de clasificación para tres emociones: alegría, enojo y tristeza, utilizando el algoritmo LDA

15.3. Discusión

El Cuadro 3 muestra la probabilidad media de aciertos y la desviación estándar para cada uno de los seis algoritmos aplicados en la clasificación de las seis emociones. El algoritmo con la mejor probabilidad media de aciertos (0.544211) es el LDA. A pesar de contar con la desviación estándar más grande (0.149611), su peor resultado para la media (0.3946) es aún mejor que el de la mayoría de los otros algoritmos. Además, podría alcanzar una media de acierto de hasta 0.693822 debido a su desviación estándar. Por lo tanto, se determinó que este algoritmo era el mejor clasificador de emociones.

Analizando específicamente el LDA, este presentó una probabilidad de acierto de 0.4792, como muestra el Cuadro 4. En cuanto a su matriz de confusión, detallada en el Cuadro 5 puede decirse lo siguiente:

- De un total de diez veces, el algoritmo identificó la alegría como disgusto una vez y como miedo dos veces.
- De seis veces, confundió al disgusto una vez con todas las emociones a excepción de la tristeza.
- De once veces, confundió al enojo con alegría tres veces, con disgusto una vez, con miedo una vez y con sorpresa dos veces.

- De cuatro veces, confundió al miedo con sorpresa una vez.
- De diez veces, confundió a la sorpresa con alegría una vez, con disgusto tres veces y con miedo una vez.
- De siete veces, confundió a la tristeza con miedo cuatro veces.

Más general, puede decirse que las emociones más fáciles de distinguir fueron el miedo y la tristeza, pues el algoritmo no las clasificó erróneamente tan seguido. En contraste, las más difíciles de distinguir resultaron ser el enojo y el disgusto, pues el algoritmo las confundió con casi todas las emociones. Por otro lado, puede decirse que el enojo y la tristeza son las emociones que menos confusión generan con las demás, pues aparecieron como confusiones para pocas emociones. También, puede observarse que el miedo es la única emoción que causó confusión con todas las demás emociones.

El Cuadro 6 muestra estadísticas importantes sobre la clasificación. La columna de precisión indica la probabilidad de esa emoción de no ser confundida con las demás; es decir, entre más cercana a uno esa cantidad, mayor es la probabilidad del algoritmo de determinar que una muestra NO es de esa emoción. La columna de llamada indica la probabilidad de esa emoción de ser identificada; entre más cercana a uno la cantidad, mayor la probabilidad del algoritmo de detectar dicha emoción en una muestra. La columna de puntuación F1 es únicamente el cálculo de la media armónica entre precisión y llamada, por lo que entre más cercana a uno esta cantidad, mejor caracterizada está la emoción. Finalmente, la columna de soporte indica cuántas ocurrencias de cada emoción se dieron durante la prueba.

Del Cuadro 6 pueden inferirse combinaciones útiles para mejorar la detección del algoritmo. Si se toman las tres emociones mejor detectadas y el análisis se reduce a ellas, es posible que la clasificación mejore. Por lo tanto, se tomaron tres combinaciones diferentes:

- Las tres emociones con mejor precisión: enojo, sorpresa y tristeza. Estas emociones resultaron ser las que menos confusión generan a las demás.
- Las tres emociones con mejor llamada: alegría, miedo y tristeza. Estas emociones resultaron ser las más fáciles de identificar.
- Las tres emociones con mejor puntuación F1: alegría, enojo y tristeza. Estas emociones resultaron ser las que mejores resultados generales presentaron.

Cabe mencionar que entonces la emoción más fácil de identificar correctamente es la tristeza, pues su "llamada" nos dice que es fácil de detectar y su "precisión" que no tiende a ser una clasificación por error. Se llevó a cabo nuevamente el procedimiento reducido a tres emociones para las tres combinaciones diferentes.

En cuanto al mejor algoritmo de clasificación a tres emociones, el Cuadro 7 para las mejores emociones según precisión y el Cuadro 15 para las mejores emociones

según puntuación F1, revelaron que se trataba nuevamente del LDA. Sin embargo, para las mejores emociones según llamado, el Cuadro 11 reveló que el mejor algoritmo era el CART. Comparando según probabilidad de acierto, el cuál se observa en los Cuadros 8, 12 y 16, se obtuvo el peor resultado para la combinación de emoción según la puntuación F1, con 0.4792, mientras que precisión y llamada empataron con 0.8333.

Sobre las matrices de confusión, el empate se encuentra entre llamada y puntuación F1, pues tanto la matriz del Cuadro 13 como la del Cuadro 17 presentan cuatro ceros, mientras que la matriz en el Cuadro 9, correspondiente a precisión presenta únicamente tres ceros.

Finalmente, al observar los Cuadros 10, 14 y 18 se determinó que las tres emociones con mejor punteo de llamada presentaron los mejores resultados promedio en precisión, llamada y puntuación F1.

Por lo tanto, utilizando el algoritmo CART, las emociones de Alegría, Miedo y Tristeza, las cuales resultaron ser las más fáciles de identificar en el análisis de seis emociones con el algoritmo LDA (pues presentaban mejores resultados para el parámetro de llamada en el reporte de clasificación), se definieron como las tres emociones principales a evaluar en las pruebas de reconocimiento de emociones.

16.1. Metodología

Se realizaron grabaciones con 41 sujetos de prueba, once fueron mujeres y treinta hombres. A cada uno se le asignó una de las diez frases ya contenidas en la base de datos y se les solicitó repetir la frase siete veces: primero en un tono neutral y luego con cada una de las seis emociones con las que se han trabajado. Se realizaron grabaciones de cada repetición, resultando en 287 grabaciones. Sin embargo, las 41 grabaciones en tono neutral no fueron utilizadas y cuatro grabaciones resultaron estar dañadas al momento de utilizarlas por lo que fueron omitidas. Por lo tanto, se recaudaron 242 grabaciones de prueba, de las cuales se trabajó con aproximadamente 120 a la vez pues las pruebas se realizaron a tres emociones y no a seis. Estas grabaciones fueron realizadas con el equipo de la cabina de audio de la Universidad Del Valle de Guatemala para reducir al máximo el ruido y obtener una mejor calidad de grabación.

Las grabaciones se realizaron bajo las siguientes restricciones:

- Grabaciones realizadas de pie
- Distancia de una cuarta entre el sujeto y el micrófono
- Un solo canal (mono)
- Velocidad de muestreo a 16 kHz
- Profundidad de 16 bits
- Grabaciones exportadas en formato .wav

- La grabación se aumentó en decibeles por software para asemejarse a la base de datos

Para llevar a cabo las pruebas, es necesario crear clasificadores de cada algoritmo a evaluar. Se utilizaron nuevamente los mismos seis algoritmos supervisados. A cada clasificador, se le enviaron los datos de la base de datos separados en parámetros y clasificación para entrenarlos. El código de pruebas consistió en iterar entre las carpetas de grabaciones para pruebas, codificar cada grabación como se hizo con las de la base de datos y enviar la codificación a cada clasificador mediante el método "predict" para obtener la predicción de cada uno. Se llevó un conteo de los aciertos para poder evaluar el mejor algoritmo. Esto se realizó con los tres sets de tres emociones definidos en el apartado anterior.

16.2. Resultados

	LR	LDA	KNN	CART	NB	SVM
Enojo, sorpresa y tristeza	24.79	51.24	21.49	52.89	35.54	33.06
Miedo, alegría, tristeza	23.97	40.50	28.93	49.59	19.83	33.06
Enojo, alegría, tristeza	41.32	42.98	33.88	55.37	30.58	33.06

Cuadro 19: Porcentajes de acierto obtenidos por cada uno de los seis algoritmos supervisados, por cada set de tres emociones

16.3. Discusión

Para evaluar la posibilidad de detectar emociones en la voz, es necesario hacer pruebas con datos ajenos a la base de datos. Por ello, se realizaron grabaciones con varios sujetos de prueba.

En el Cuadro 19 se observan los porcentajes de acierto obtenidos por cada algoritmo supervisado, para cada uno de los tres sets de emociones definidos en el capítulo anterior. No se obtuvo ningún porcentaje mayor al 60 %, por lo que no puede asegurarse que es posible detectar emociones en la voz. Sin embargo, se obtuvieron porcentajes prometedores en los algoritmos de LDA y CART. Para las emociones de miedo, alegría y tristeza se obtuvo un porcentaje de acierto del 55.37% con el algoritmo CART. Este fue el mejor porcentaje presentado.

A pesar de que no es posible afirmar que pueden detectarse emociones en la voz, tampoco puede descartarse la posibilidad debido a los siguientes criterios:

- El árbol de decisiones realizado al evaluar qué parámetros eran relevantes no descartó ningún parámetro. Puede sospecharse que aún no se han identificado todos los parámetros cruciales en la detección de emociones.

- Solo se tomaron 20 datos por emoción para el entrenamiento. La base de datos adquirida cuenta con 368 datos por cada emoción. El número de datos de entrenamiento podría ser muy pequeño.
- Se utilizaron como pruebas todos los archivos recaudados de los sujetos de prueba, a excepción de los cuatro archivos corruptos. Dentro de los sujetos de prueba, existían quienes presentaban emociones poco definidas en sus grabaciones. Podrían extraerse los archivos considerados no emocionales y repetir la validación.

Fue posible implementar un sistema capaz de reconocer efectivamente patrones de voz y hablar con un tono similar al humano basándose en tecnologías de STT y TTS. Con ello, se cumple con el primer objetivo específico planteado.

Incorporando un poco de aprendizaje de máquinas a las tecnologías antes mencionadas fue posible dotar al sistema con la capacidad de conversar, cumpliendo así con el segundo objetivo específico planteado. Para este sistema, el aprendizaje de máquinas se utilizó en un menor porcentaje que en el tema de detección de emociones. Esta tecnología está presente al seleccionar las respuestas en una conversación. En cuanto al desempeño de este sistema, es posible llevar una conversación con el programa; más la interacción no es tan fluida como se esperaba. Esto podría mejorarse implementando un sistema de reconocimiento de voz (STT) en tiempo real y no basado en grabaciones de audio aisladas.

Mediante comunicación serial y definiendo una base de datos adicional, fue posible completar el tercer objetivo específico: Codificar movimientos respuesta accionados por voz.

Para el reconocimiento de emociones, se determinó que la emoción más difícil de clasificar es el disgusto. Las otras cinco emociones pueden identificarse casi sin problemas si se evalúan como máximo tres emociones a la vez. Clasificando entre las seis emociones a la vez y utilizando como pruebas elementos propios de la base de datos, el algoritmo presentó dificultades; pues se obtuvo como máximo un porcentaje de acierto del 69%. Puesto que con estas pruebas no hay lugar a que se trate de datos que el algoritmo no llegó a conocer en su entrenamiento, la mejora del porcentaje de acierto radica en un aumento de la selección de parámetros relevantes para la detección de emociones.

Utilizando como pruebas datos registrados en la base de datos de emociones con la que se entrenaron los algoritmos, puede obtenerse hasta un 95% de probabilidad de

acierto para las emociones de enojo, sorpresa y tristeza; Hasta un 90% para alegría, miedo y tristeza; y hasta un 92% para alegría, enojo y tristeza. Estos resultados involucran una alta probabilidad de detectar emociones con datos que no sean propios de la base de datos.

Para datos externos a la base de datos (es decir, datos que no se encuentran en la base de datos), el porcentaje de acierto más alto obtenido fue de 55%, para las emociones de enojo, alegría y tristeza. Este bajo resultado puede deberse a que no se le proporcionaron suficientes datos de entrenamiento al algoritmo, a que no se determinaron todos los parámetros relevantes en la detección de emociones y/o a que algunos de los sujetos de prueba presentaron emociones muy débiles. Esto significa que, haciéndose cargo de estos aspectos, el porcentaje de detección para emociones con datos ajenos a la base de datos puede mejorarse. Este procedimiento completó el último objetivo específico: Determinar la factibilidad y viabilidad de detectar estados emocionales por medio de la voz.

Queda así completado el objetivo general de este trabajo: Conseguir que el animatrónico pueda interpretar el idioma español y reaccionar según sea conveniente, respondiendo verbalmente similar a un ser humano y llevar a cabo un estudio sobre la detección de emociones en la voz. Al construir un sistema de conversación capaz de reconocer patrones de voz mediante tecnologías STT y hablar con un tono de voz similar al de un ser humano mediante tecnologías TTS se consigue poder interpretar el idioma español y configurar las diferentes formas de respuesta del animatrónico: verbalmente (es decir, pronunciado palabras) o con acciones (entiéndase, señales enviadas por comunicación serial que le indican al módulo de movimientos [5] la acción que debe realizar). El estudio sobre la detección de emociones en la voz reveló que sí existen patrones vocales que posibilitan diferenciar estados anímicos, sin embargo, no pudieron identificarse todos los factores cruciales para poder implementar un programa capaz de diferenciar emociones en la voz.

Para el sistema de conversación, se recomienda implementar un programa de STT que funcione en tiempo real y no sea basado en grabación momentánea de datos, pues esto mejoraría la interacción con el usuario debido a que el programa proporcionaría respuestas más rápido. También, se recomienda buscar una manera de mantener el contexto en las conversaciones, pues aunque el programa responde de manera acertada a las entradas por voz, no es capaz de recordar de qué se hablaba anteriormente.

En cuanto al estudio de detección de emociones, sería conveniente implementar un programa en tiempo real también. Esto con el fin de observar en vivo el funcionamiento del programa. También, se recomienda aumentar la cantidad de parámetros relevantes para la clasificación de emociones. Con ello, talvez podría mejorarse el rendimiento del programa aún trabajando con las 6 emociones.

-
- [1] M. Benegas, en *Trabajo de graduación en modalidad de Tesis: Diseño Mecánico e Implementación de Cabeza Robótica Animatrónica*, 2018.
 - [2] B. Rodas, en *Trabajo de graduación en modalidad de Tesis: Diseño e Implementación de mano y antebrazo animatrónico antropomorfo*, 2018.
 - [3] F. Zelada, en *Diseño y construcción de un brazo de 5 grados de libertad para un robot humanoide de tamaño real*, 2018.
 - [4] L. Ruano, en *Sistema de detección de rostro y reconocimiento de expresiones faciales a través de gestos para robot animatrónico*, 2018.
 - [5] A. García, “Banderas definidas para instrucciones con módulo de interacción por voz”, en *Trabajo de graduación en modalidad de Tesis: Módulo de alimentación y control de movimiento de servo motores para pirata animatrónico*, 2018.
 - [6] S. Saha y C. Marsh, *Jasper | Documentation*. dirección: <https://jasperproject.github.io/documentation/configuration/> (visitado 03-06-2018).
 - [7] Gunther Cox, *About ChatterBot — ChatterBot 0.8.7 documentation*, 2018. dirección: <https://chatterbot.readthedocs.io/en/stable/> (visitado 28-09-2018).
 - [8] Existor, *Evie - Eviebot.com - IA avatar femenino y amiga - avatar con emociones - comunicación, servicio de atención al cliente, juegos, robots - Habla muchos idiomas*. dirección: <https://www.eviebot.com/es/> (visitado 28-09-2018).
 - [9] Cleverscript, *Cleverbot.com - a clever bot - speak to an AI with some Actual Intelligence?*, 2018. dirección: <https://www.cleverbot.com/> (visitado 28-09-2018).
 - [10] Pandorabots, *Mitsuku*. dirección: <https://www.pandorabots.com/mitsuku/> (visitado 28-09-2018).
 - [11] Ediciones Universidad de Salamanca., *TECNICAS DE RECONOCIMIENTO AUTOMÁTICO DE EMOCIONES*. Ediciones Universidad Salamanca. dirección: <http://www.redalyc.org/html/2010/201017296007/>.

- [12] M. E. Sánchez-Gutiérrez, E. M. Albornoz, F. Martínez-Licona, H. L. Rufiner y J. Goddard, “Deep Learning for Emotional Speech Recognition”, dirección: http://sinc.unl.edu.ar/sinc-publications/2014/SAMRG14/sinc%7B%5C_%7DSAMRG14.pdf.
- [13] I. Luengo, U. / Ehu, A. Urquijo, E. Navas, I. Hernández y J. Sánchez, “Reconocimiento automático de emociones utilizando parámetros prosódicos”, inf. téc., 2005. dirección: <https://pdfs.semanticscholar.org/4873/1f65d06bfa53bd2a1b46c715799af676ef01.pdf>.
- [14] K. Karpouzis, G. Votsis, G. Moschovitis y S. Kollias, “Emotion recognition using feature extraction and 3-d models”, en *CSCC'99 Proceedings*, 1999, págs. 5371-5376.
- [15] S. Ben-Yacoub, Y. Abdeljaoued y E. Mayoraz, “Fusion of face and speech data for person identity verification.”, en *IEEE Transactions on Neural Networks*, 10, IEEE, 1999, págs. 1065-1074.
- [16] Techopedia, *What is Speech-to-Text Software? - Definition from Techopedia*. dirección: <https://www.techopedia.com/definition/23767/speech-to-text-software> (visitado 03-06-2018).
- [17] M. Rouse, *What is text-to-speech (TTS)? - Definition from WhatIs.com*. dirección: <https://searchmobilecomputing.techtarget.com/definition/text-to-speech> (visitado 03-06-2018).
- [18] M. Guadalupe, *PARAMETROS DEL SONIDO | Blog de MUSICA*. dirección: <https://musicaliceocarmelo.wordpress.com/2010/05/04/parametros-del-sonido/> (visitado 03-06-2018).
- [19] UVA, *Parámetros fundamentales del audio digital*. dirección: https://www.lpi.tel.uva.es/%7B~%7Dnacho/docencia/ing%7B%5C_%7Dond%7B%5C_%7D1/trabajos%7B%5C_%7D01%7B%5C_%7D02/formatos%7B%5C_%7Daudio%7B%5C_%7Ddigital/html/param.htm (visitado 03-06-2018).
- [20] Apowersoft, *Conocimiento general del formato WAV*. dirección: <https://www.apowersoft.es/que-es-el-formato-wav.html> (visitado 03-06-2018).
- [21] IBM Watson, *Audio formats*, 2018. dirección: <https://console.bluemix.net/docs/services/speech-to-text/audio-formats.html%7B%5C#%7Daudio-formats> (visitado 12-09-2018).
- [22] National Instruments, *Enseñe Conceptos Difíciles: Dominio de Frecuencia en Medidas - National Instruments*. dirección: <http://www.ni.com/tutorial/13042/es/> (visitado 30-09-2018).
- [23] A. Franco, *Transformada rápida de Fourier (I)*, 2016. dirección: http://www.sc.ehu.es/sbweb/fisica3/datos/fourier/fourier%7B%5C_%7D1.html (visitado 30-09-2018).
- [24] CircuitLib, *Voice Band Pass Filter*. dirección: https://www.circuitlib.com/index.php/schematics/product/46-voice-band-pass/category%7B%5C_%7Dpathway-32 (visitado 30-09-2018).
- [25] A. Diez, *Machine Learning, la base de la inteligencia artificial*. dirección: <https://o microno.elespanol.com/2016/06/machine-learning-inteligencia-artificial/> (visitado 03-06-2018).

- [26] Raona Enginyers, *Los 10 Algoritmos esenciales en Machine Learning - Raona*, 2017. dirección: <https://www.raona.com/los-10-algoritmos-esenciales-machine-learning/> (visitado 28-09-2018).
- [27] “Tema 1: Análisis Discriminante Lineal Introducción al Análisis Discriminante”, inf. téc. dirección: <http://halweb.uc3m.es/esp/Personal/personas/jmmarin/esp/DM/tema1dm.pdf>.
- [28] S. Ruiz, *El algoritmo K-NN y su importancia en el modelado de datos | Analítica web*. dirección: <https://www.analiticaweb.es/algoritmo-knn-modelado-datos/> (visitado 29-09-2018).
- [29] E. Descriptiva Autor, J. Manuel y B. Espinosa, “MATEMÁTICAS BÁSICAS ESTADÍSTICA DESCRIPTIVA DEFINICIÓN Y CLASIFICACIÓN DE VARIABLES”, inf. téc. dirección: http://132.248.164.227/publicaciones/docs/apuntes%7B%5C_%7Dmatematicas/34.%20Estadistica%20Descriptiva.pdf.
- [30] A. G. Martinez, *Asimetría Y Curtosis: Medidas Estadísticas de Distribución - Curso de Spss Gratis*, 2015. dirección: <http://www.spssfree.com/curso-de-spss/analisis-descriptivo/medidas-de-distribucion-curtosis-asimetria.html> (visitado 28-09-2018).
- [31] Sistemas UP, “Validación cruzada”, inf. téc. dirección: <https://ingsistemastelesup.files.wordpress.com/2017/03/validacion-cruzada.pdf>.
- [32] Python Programming, *Speech Recognition | Python*, 2018. dirección: <https://pythonprogramminglanguage.com/speech-recognition/> (visitado 28-09-2018).
- [33] Peter Parente, *Using pyttsx — pyttsx 1.0 documentation*, 2009. dirección: <https://pyttsx.readthedocs.io/en/v1.0/engine.html> (visitado 28-09-2018).
- [34] Python Software Foundation, *7.2. re — Regular expression operations — Python 2.7.15 documentation*, 2018. dirección: <https://docs.python.org/2/library/re.html> (visitado 28-09-2018).
- [35] —, *7.9. unicodedata — Unicode Database — Python 2.7.15 documentation*, 2018. dirección: <https://docs.python.org/2/library/unicodedata.html> (visitado 28-09-2018).
- [36] —, *9.6. random — Generate pseudo-random numbers — Python 2.7.15 documentation*, 2018. dirección: <https://docs.python.org/2/library/random.html> (visitado 28-09-2018).
- [37] —, *28.1. sys — System-specific parameters and functions — Python 2.7.15 documentation*, 2018. dirección: <https://docs.python.org/2/library/sys.html> (visitado 28-09-2018).
- [38] Hubert Pham, *PyAudio Documentation — PyAudio 0.2.11 documentation*, 2006. dirección: <https://people.csail.mit.edu/hubert/pyaudio/docs/> (visitado 28-09-2018).
- [39] Python Software Foundation, *21.5. wave — Read and write WAV files — Python 2.7.15 documentation*, 2018. dirección: <https://docs.python.org/2/library/wave.html> (visitado 28-09-2018).
- [40] The SciPy Community, *Statistical functions (scipy.stats) — SciPy v1.1.0 Reference Guide*, 2018. dirección: <https://docs.scipy.org/doc/scipy/reference/stats.html> (visitado 28-09-2018).

- [41] —, *Statistics — NumPy v1.15 Manual*, 2018. dirección: <https://docs.scipy.org/doc/numpy/reference/routines.statistics.html> (visitado 28-09-2018).
- [42] PyData Org, *pandas: powerful Python data analysis toolkit — pandas 0.23.4 documentation*. dirección: <https://pandas.pydata.org/pandas-docs/stable/> (visitado 28-09-2018).
- [43] Scikit-Learn Developers, *API Reference — scikit-learn 0.20.0 documentation*, 2018. dirección: <http://scikit-learn.org/stable/modules/classes.html> (visitado 28-09-2018).
- [44] The Matplotlib Development Team, *Matplotlib: Python plotting — Matplotlib 3.0.0 documentation*, 2018. dirección: <https://matplotlib.org/> (visitado 28-09-2018).
- [45] Python Software Foundation, *15.1. os — Miscellaneous operating system interfaces — Python 2.7.15 documentation*, 2018. dirección: <https://docs.python.org/2/library/os.html> (visitado 28-09-2018).
- [46] —, *10.3. stat — Interpreting stat() results — Python 2.7.15 documentation*, 2018. dirección: <https://docs.python.org/2/library/stat.html> (visitado 28-09-2018).
- [47] C. Liechti, *Welcome to pySerial's documentation — pySerial 3.0 documentation*, 2015. dirección: <https://pythonhosted.org/pyserial/> (visitado 01-10-2018).
- [48] ELRA, *Emotional speech synthesis database — ELRA Catalogue*. dirección: <http://catalog.elra.info/en-us/repository/browse/emotional-speech-synthesis-database/629db920a9e811e7a093ac9e1701ca021bdb22603cbc4702a3b6f592d250e427/> (visitado 03-06-2018).
- [49] —, *About*, 2015. dirección: <http://www.elra.info/en/about/> (visitado 28-09-2018).
- [50] Webgraphviz, *Webgraphviz*. dirección: <http://webgraphviz.com/> (visitado 28-09-2018).

20.1. Código en Python

```
import speech_recognition as sr #Libreria para el reconocimiento de voz

r = sr.Recognizer()
m = sr.Microphone()

try:
    print("Ajustando al ruido del ambiente...")
    with m as source: r.adjust_for_ambient_noise(source)
    umbralminimo= round(r.energy_threshold, 1) #Se redondea el resultado a un decimal
    print("El umbral del sensibilidad minimo debe ser de {}".format(umbralminimo))
    with m as source: r.energy_threshold=umbralminimo+1000 #Se aumenta el umbral
    with m as source: r.dynamic_energy_threshold = False #Se impide que el umbral cambie
    print("Umbral de sensibilidad setteado a {}".format(r.energy_threshold))
    print
    #-----Ciclo de reconocimiento de audio-----
    while True:
        print("Esperando entrada de audio...")
        try:
            with m as source: audio = r.listen(source, timeout=3.0)
            print("Entrada de audio recibida, reconociendo palabras...")
            print
            # Se reconoce el audio con google
            value = r.recognize_google(audio, language="es-us")
            #Se codifica el string en el formato predeterminado de salida
            if str is bytes: # this version of Python uses bytes for strings (Python 2)
                value=(u"Usted dijo {}".format(value).encode("utf-8"))
            else: # this version of Python uses unicode for strings (Python 3+)
                value=("Usted dijo {}".format(value))
            print value
            print
        except sr.WaitTimeoutError:
            print "Tiempo de espera excedido - no se detectó entrada de audio)"
        except sr.UnknownValueError:
            print("No fue posible descifrar palabras en el audio")
    pass
```

Figura 31: Código para el reconocimiento de voz

```

import pyttsx #Libreria para la generacion del habla

cont=1 #Variable para llevar el conteo de las voces disponibles
engine = pyttsx.init()
engine2= pyttsx.init()
#-----Para elegir voces-----
voices = engine.getProperty('voices')
for voice in voices:
    engine.setProperty('voice', voice.id) # changes the voice
    engine.say('The quick brown fox jumped over the lazy dog.')
    print "VOZ #" +str(cont)
    print voice
    print
    cont+=1
engine.runAndWait()

#-----Probando la voz elegida-----
#Voz en español disponible:
voice="HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Speech\Voices\Tokens\TTS_MS_ES-MX_SABINA_11.0'

#Configurando la voz y la velocidad del motor
rate=engine.getProperty('rate')
print rate
engine.setProperty('rate', 150)
engine.setProperty('voice', voice) # changes the voice

#Ejecucion
frase="Esta es una prueba de voz"
print frase
engine.say(frase)
engine.runAndWait()

```

Figura 32: Código para la generación del habla

```

1 # -*- coding: utf-8 -*-
2 from chatterbot import ChatBot
3 from chatterbot.trainers import ListTrainer
4 from chatterbot.trainers import ChatterBotCorpusTrainer
5 import pyttsx
6 import re
7 import unicodedata
8 import speech_recognition as sr
9 import sys
10 import random
11
12
13 #-----Definición de funciones-----
14 def sinAcento(text):
15 #Función para eliminar todos los caracteres que el TTS no puede pronunciar
16 #text = text.lower()
17 try:
18     text = unicode(text, 'utf-8')
19 except (TypeError, NameError): # unicode is a default on python 3
20     pass
21 text = unicodedata.normalize('NFD', text)
22 text = text.encode('ascii', 'ignore')
23 text = text.decode('utf-8')
24 text = re.sub('[ ]+', ' ', text)
25 text = re.sub('[0-9a-zA-Z_]', '', text)
26 text=re.sub("ñ", "ni", text)
27 text= text.replace(" ", "")
28 return text
29
30 def hablar(frase):
31 #Función para utilizar el TTS
32 print ""+frase+"
33 engine.say(sinAcento(frase))
34 engine.runAndWait()
35
36 #-----Definiciones para reconocimiento de audio-----
37 r = sr.Recognizer()
38 m = sr.Microphone()
39
40 #-----Definición de la voz del chatbot-----
41 voice="HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Speech\Voices\Tokens\TTS_MS_ES-NX_SABINA_11.0"
42 engine = pyttsx.init()
43 engine.setProperty('rate', 150)
44 engine.setProperty('voice', voice)
45
46 #-----Entrenando chatbot-----
47 nombreChatbot="tesoro"
48 bot = ChatBot(nombreChatbot)
49
50 #-----Eliminar memoria del Chatbot (Descomentar y correr programa) -----
51 # =====
52 # bot.storage.drop()
53 # print "Memoria borrada"
54 # sys.exit()
55 # =====
56
57 #-----Entrenamiento, solo es necesario la primera vez (Mantener Comentado)-----
58 # =====
59 # bot.set_trainer(ListTrainer)
60 # conv = open('Conversaciones.txt', 'r').readlines()
61 # bot.train(conv)
62 # conv = open('Saludos.txt', 'r').readlines()
63 # bot.train(conv)
64 # conv = open('Trivia.txt', 'r').readlines()
65 # bot.train(conv)
66 # conv = open('Varios.txt', 'r').readlines()
67 # bot.train(conv)
68 # =====
69
70 #-----Introducción-----
71 nombre="Elen"
72 print nombreChatbot+" : "
73 hablar("Hola, cómo te llamas?")
74 hablar("Por favor escríbeme tu nombre")
75 print "Nombre: "
76 nombre=raw_input() #Arreglar problema con tildes
77
78 print nombreChatbot+" : "
79 hablar("Mucho gusto. "+str(nombre))
80 hablar ("Mi nombre es "+str(nombreChatbot))
81 starters = open('Conversaciones.txt', 'r').readlines()
82
83 #-----Conversación-----
84 try:
85     print("-----[Preparando...]-----")
86     with m as source: r.adjust_for_ambient_noise(source)
87     print("-----[Energy threshold mínimo {}]-----".format(r.energy_threshold))
88     with m as source: r.energy_threshold= r.energy_threshold+1000
89     print("-----[Energy threshold setteado a {}]-----".format(r.energy_threshold))
90     print
91     hablar("Ahora podremos conversar")
92     while True:
93         print "..."
94         try:
95             with m as source: audio = r.listen(source, timeout=5.0)
96             with m as source: r.dynamic_energy_threshold = False
97             print nombre+" : "
98             # Reconociendo audio con google
99             value = r.recognize_google(audio, language="es-us")
100             print value
101             #Debido a los caracteres extraños, los strings deben manipularse
102             if str is bytes: # this version of Python uses bytes for strings (Python 2)
103                 #print(u"Diiste {}".format(value).encode("utf-8"))
104                 value=format(value).encode("utf-8")
105             else: # this version of Python uses unicode for strings (Python 3+)
106                 #print("Diiste {}".format(value))
107                 value=format(value)
108             value=sinAcento(str(value))
109             response = bot.get_response(str(value))
110             print nombreChatbot+" : "
111             response=unicode(response)
112             #print response
113             hablar(response)
114         except sr.WaitTimeoutError:
115             print "iniciando conversación"
116             empezar=random.choice(starters)
117             hablar(empezar)
118         except sr.UnknownValueError:
119             print("-----[Chatbot no te escucha]-----")
120         except sr.RequestError as e:
121             print("-----[Google no entendio...]----- ; {}".format(e))
122         except KeyboardInterrupt, EOFError, SystemExit:
123             break
124     pass
125
126 except KeyboardInterrupt:
127     break
128 except KeyboardInterrupt:
129     pass

```

Figura 33: Código para la emulación de conversación humana

```

1
2 from scipy.io import wavfile
3 from matplotlib import pyplot as plt
4 import numpy as np
5
6 import pyaudio
7 import wave
8
9 FORMAT = pyaudio.paInt16
10 CHANNELS = 2
11 RATE = 44100
12 CHUNK = 1024
13 RECORD_SECONDS = 15
14 WAVE_OUTPUT_FILENAME = "Prueba.wav"#Intensidad.wav
15
16 audio = pyaudio.PyAudio()
17
18 #Grabar Datos
19 print "Inicia grabacion"
20 Grabacion = audio.open(format=FORMAT, channels=CHANNELS,
21                         rate=RATE, input=True,
22                         frames_per_buffer=CHUNK)
23 print "grabando..."
24 frames = []
25
26 for i in range(0, int(RATE / CHUNK * RECORD_SECONDS)):
27     data = Grabacion.read(CHUNK)
28     frames.append(data)
29
30 print "Grabacion terminada"
31 Grabacion.stop_stream()
32 Grabacion.close()
33 audio.terminate()
34
35 #Guardar La grabacion
36 waveFile = wave.open(WAVE_OUTPUT_FILENAME, 'wb')
37 waveFile.setnchannels(CHANNELS)
38 waveFile.setsampwidth(audio.get_sample_size(FORMAT))
39 waveFile.setframerate(RATE)
40 waveFile.writeframes(b''.join(frames))
41 waveFile.close()
42
43 # Cargar Los datos para graficar
44 samplerate, data = wavfile.read("Prueba.wav")
45 times = np.arange(len(data))/float(samplerate)
46 print samplerate
47 print data
48 print len (data)
49
50 #Plottear
51
52 #-----Plot ESTEREO (CHANNELS =2)
53 # =====
54 # plt.figure(figsize=(5, 5))
55 # plt.fill_between(times, data[:,0], data[:,1], color='k') #stereo
56 # =====
57
58 #print data[:,0]
59
60 #-----Plot MONO (CHANNELS =1)
61 plt.figure(figsize=(5, 5))
62 plt.fill_between(times, data) #mono
63
64 #-----Plot ESTEREO CANALES SEPARADOS (CHANNELS =2)
65 #plt.figure(figsize=(5, 5))
66 #plt.fill_between(times, data[:,0]) #mono
67 #plt.figure(figsize=(5, 5))
68 #plt.fill_between(times, data[:,1])
69 #print np.average(data[:,0])
70 #print np.average(data[:,1])
71 plt.xlim(times[0], times[-1])
72 plt.xlabel('time (s)')
73 plt.ylabel('amplitude')
74
75 #Guardar el plot
76 plt.savefig('plot.png', dpi=100)
77 plt.show()
78

```

Figura 34: Código para las pruebas de intensidad


```

1 import pyaudio
2 import numpy as np
3 from scipy import fft
4 import time
5
6 # to display in separate Tk window
7
8 # constants
9 CHUNK = 1024 * 8           # samples per frame
10 FORMAT = pyaudio.paInt16  # audio format (bytes per sample?)
11 CHANNELS = 1              # single channel for microphone
12 RATE = 44100              # samples per second
13
14 # pyaudio class instance
15 p = pyaudio.PyAudio()
16
17 # stream object to get data from microphone
18 stream = p.open(
19     format=FORMAT,
20     channels=CHANNELS,
21     rate=RATE,
22     input=True,
23     output=True,
24     frames_per_buffer=CHUNK
25 )
26
27
28 print('stream started')
29
30 xf = np.linspace(0, RATE, CHUNK)    # frequencies (spectrum)
31 xf = xf[0:CHUNK/2]
32
33 # for measuring frame rate
34 frame_count = 0
35 start_time = time.time()
36
37 while True:
38
39     # binary data
40     data = stream.read(CHUNK)
41
42     # convert data to integers, make np array, then offset it by 127
43     data_int = np.fromstring(data, dtype= np.int16)
44
45     # create np array and offset by 128
46     data_np = data_int
47
48
49     # compute FFT and update line
50     yf = fft(data_int)
51     yf_abs = np.abs(yf[0:CHUNK]) / (128 * CHUNK)
52     yf_abs = yf_abs[0:CHUNK/2]
53
54     if(np.amax(yf_abs)>1.8):
55         max_yf = np.amax(yf_abs)
56         index = np.nonzero(yf_abs==max_yf)[0][0]
57         freqLeida= abs(xf[index])
58         if(freqLeida > 200):
59             yf_abs[0:index].fill(0)
60             print(freqLeida, index, yf_abs[index])
61

```

Figura 35: Código para las pruebas de frecuencia


```

1# -*- coding: utf-8 -*-
2"""
3Created on Tue Aug 21 16:10:36 2018
4
5@author: Ma. Belen
6"""
7
8# Librerias utilizadas
9import pandas as pd
10import numpy as np
11from sklearn import tree
12
13#-----Cargar datos-----
14fileName= "EMOTIONDB.txt"
15names = ['cntV', 'minV', 'maxV', 'avgV', 'medV', 'varV', 'desV', 'oblV',
16         'curV', 'picomax', 'picomin', 'rangof', 'freqmax', 'freqmin',
17         'contF', 'maxF', 'avgF', 'medF', 'varF', 'desF', 'oblF',
18         'curF', 'velF', 'sexo', 'Emotion']
19
20dataset = pd.read_csv(fileName, names=names)
21
22emo_classifier = tree.DecisionTreeClassifier()
23emo_classifier.fit(dataset[['cntV', 'minV', 'maxV', 'avgV', 'medV',
24                          'varV', 'desV', 'oblV', 'curV', 'picomax',
25                          'picomin', 'rangof', 'freqmax', 'freqmin',
26                          'contF', 'maxF', 'avgF', 'medF', 'varF', 'desF',
27                          'oblF', 'curF', 'velF', 'sexo']],
28                dataset['Emotion'])
29
30print ">>>> Trained Emotion_classifier <<<<<"
31print emo_classifier
32
33with open("emotion_classifier.txt", "w") as f:
34    f = tree.export_graphviz(emo_classifier, out_file=f)
35

```

Figura 37: Código para la creación del árbol de decisiones

```

1#-----Cargar Librerias-----
2import pandas
3from pandas.plotting import scatter_matrix
4import matplotlib.pyplot as plt
5from sklearn import model_selection
6from sklearn.metrics import classification_report
7from sklearn.metrics import confusion_matrix
8from sklearn.metrics import accuracy_score
9from sklearn.linear_model import LogisticRegression
10from sklearn.tree import DecisionTreeClassifier
11from sklearn.neighbors import KNeighborsClassifier
12from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
13from sklearn.naive_bayes import GaussianNB
14from sklearn.svm import SVC
15
16#-----Cargar datos-----
17fileName = "EMOTIONDB3MAT.txt"
18names = ['cntV', 'minV', 'maxV', 'avgV', 'medV', 'varV', 'desV', 'oblV',
19         'curV', 'picomax', 'picomin', 'rangof', 'freqmax', 'freqmin',
20         'contF', 'maxF', 'avgF', 'medF', 'varF', 'desF', 'oblF',
21         'curF', 'velF', 'sexo', 'Emotion'] #24 parametros
22#-----
23# names = ['cntV', 'minV', 'maxV', 'rangov', 'avgV', 'medV', 'modV', 'varV',
24         'desV', 'oblV', 'curV', 'corrV', 'pcorrV', 'picomax', 'picomin',
25         'rangof', 'freqmax', 'freqmin', 'contF', 'maxF', 'avgF', 'medF',
26         'modF', 'varF', 'desF', 'oblF', 'curF', 'corrF', 'pcorrF', 'velF',
27         'sexo', 'Emotion'] #31 parametros
28#-----
29#-----
30# names = ['minV', 'maxV', 'medV', 'varV', 'oblV', 'curV', 'freqmax', 'freqmin',
31         'maxF', 'oblF', 'curF', 'Emotion'] #11 parametros
32#-----
33#-----
34dataset = pandas.read_csv(fileName, names=names)
35
36#-----Resumen de Datos-----
37# shape
38print "DIMENSIONES DEL DATASET: "
39print (dataset.shape)
40# head
41print "PRIMEROS 20 DATOS: "
42print(dataset.head(20))
43# descriptions
44print "ESTADISTICAS: "
45print(dataset.describe())
46# class distribution
47print "DISTRIBUCIÓN DE CLASES: "
48print(dataset.groupby('Emotion').size())
49
50#-----Graficas-----
51print "GRÁFICAS POR VARIABLE: "
52# box and whisker plots
53print "Diagramas de caja y bigotes:"
54dataset.plot(kind='box', subplots=True, layout=(31,1), sharex=False, sharey=False,
55            figsize=(5,30))
56plt.show()
57# histograms
58print "Histogramas:"
59dataset.hist(figsize=(12,12))
60plt.show()
61print "GRÁFICAS MULTIVARIABLE: "
62# scatter plot matrix
63print "Las gráficas con comportamiento en diagonal muestran relación entre pares de variables"
64scatter_matrix(dataset, figsize=(15,15))
65plt.show()
66
67#-----Algoritmos-----
68# Dividir datos
69array = dataset.values
70X = array[:,0:24] #datos sin resultados (descripcion de grabaciones)
71Y = array[:,24] #columna de resultados (emociones)
72validation_size = 0.20
73seed = 7
74X_train, X_validation, Y_train, Y_validation = model_selection.train_test_split(X, Y, test_size=validation_size, random_state=seed)
75
76# Opciones de prueba
77seed = 7
78scoring = 'accuracy'
79
80# Algoritmos
81models = []
82models.append(('LR', LogisticRegression()))
83models.append(('LDA', LinearDiscriminantAnalysis()))
84models.append(('KNN', KNeighborsClassifier()))
85models.append(('CART', DecisionTreeClassifier()))
86models.append(('NB', GaussianNB()))
87models.append(('SVM', SVC()))
88# Evaluando el modelo de turno
89results = []
90names = []
91for name, model in models:
92    kfold = model_selection.KFold(n_splits=10, random_state=seed)
93    cv_results = model_selection.cross_val_score(model, X_train, Y_train, cv=kfold, scoring=scoring)
94    results.append(cv_results)
95    names.append(name)
96    msg = "%s: %f (%f)" % (name, cv_results.mean(), cv_results.std())
97    print(msg)
98
99#Comparar algoritmos
100fig = plt.figure(figsize=(10,10))
101fig.suptitle('Algorithm Comparison')
102ax = fig.add_subplot(111)
103plt.boxplot(results)
104ax.set_xticklabels(names)
105plt.show()
106
107# Make predictions on validation dataset
108print
109knn = DecisionTreeClassifier()
110knn.fit(X_train, Y_train)
111predictions = knn.predict(X_validation)
112print(accuracy_score(Y_validation, predictions))
113print(confusion_matrix(Y_validation, predictions))
114print(classification_report(Y_validation, predictions))
115

```

Figura 38: Código para la evaluación de la posibilidad de detectar emociones en la voz

20.2. Codificación de movimientos

Valor	Instrucción	Descripción
0	Terminar conversación	Termina movimiento de la boca
1	Iniciar conversación	Inicia movimiento de la boca
2	Mover la muñeca	Ademán de saludo o despedida
3	Apuntar derecha	
4	Apuntar arriba	
5	Apuntar izquierda	
6	Asentir	Movimiento de la cabeza para asentir
7	Negar	Movimiento de la cabeza para negar
8	Pregunta	Ademán de pregunta
9	Confusión	Similar a encogerse de hombros
10	Máscara	
11	Pistola	
12	Oreja	Señalar el oído
13	Música	
14	Pulgar arriba	
15	Señal de paz	
16	Cara neutral	
17	Felicidad	Gestos de felicidad
18	Tristeza	Gestos de tristeza
19	Enojo	Gestos de enojo

[5]

Cuadro 20: Interpretación por parte del módulo de *Control del movimiento de motores* de las banderas de movimiento codificadas

20.3. Árbol de decisiones

```

1 digraph Tree {
2   node [shape=box];
3   0 [label="X[20] <= 5.142\ngini = 0.814\nsamples = 359\nvalue = [40, 39, 40, 40, 120, 40, 40]"];
4   1 [label="X[20] <= 3.638\ngini = 0.797\nsamples = 132\nvalue = [35, 12, 28, 6, 21, 30, 0]"];
5   0 -> 1 [labeldistance=2.5, labelangle=45, headlabel="True"];
6   2 [label="X[7] <= 1.853\ngini = 0.411\nsamples = 28\nvalue = [21, 0, 4, 0, 2, 1, 0]"];
7   1 -> 2;
8   3 [label="X[5] <= 4605231.0\ngini = 0.226\nsamples = 24\nvalue = [21, 0, 2, 0, 0, 1, 0]"];
9   2 -> 3;
10  4 [label="gini = 0.0\nsamples = 20\nvalue = [20, 0, 0, 0, 0, 0, 0]"];
11  3 -> 4;
12  5 [label="X[22] <= 3.328\ngini = 0.625\nsamples = 4\nvalue = [1, 0, 2, 0, 0, 1, 0]"];
13  3 -> 5;
14  6 [label="X[5] <= 5239647.0\ngini = 0.5\nsamples = 2\nvalue = [1, 0, 0, 0, 0, 1, 0]"];
15  5 -> 6;
16  7 [label="gini = 0.0\nsamples = 1\nvalue = [1, 0, 0, 0, 0, 0, 0]"];
17  6 -> 7;
18  8 [label="gini = 0.0\nsamples = 1\nvalue = [0, 0, 0, 0, 0, 1, 0]"];
19  6 -> 8;
20  9 [label="gini = 0.0\nsamples = 2\nvalue = [0, 0, 2, 0, 0, 0, 0]"];
21  5 -> 9;
22  10 [label="X[20] <= 3.328\ngini = 0.5\nsamples = 4\nvalue = [0, 0, 2, 0, 2, 0, 0]"];
23  2 -> 10;
24  11 [label="gini = 0.0\nsamples = 2\nvalue = [0, 0, 2, 0, 0, 0, 0]"];
25  10 -> 11;
26  12 [label="gini = 0.0\nsamples = 2\nvalue = [0, 0, 0, 0, 2, 0, 0]"];
27  10 -> 12;

```

Figura 40: Fragmento del código generado automáticamente para visualizar el árbol de decisiones en Webgraphviz

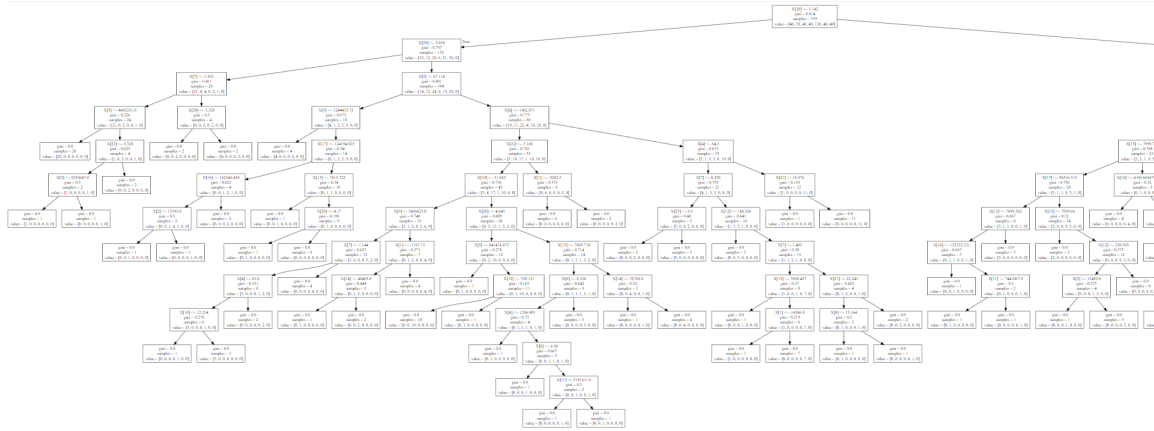


Figura 41: Porción 1 del árbol generado por Webgraphviz

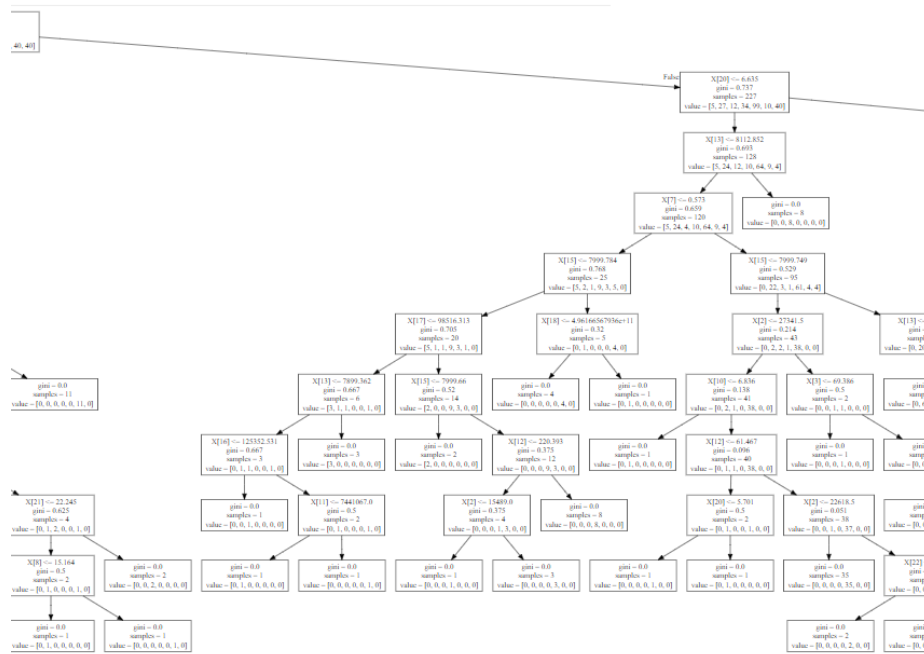


Figura 42: Porción 2 del árbol generado por Webgraphviz

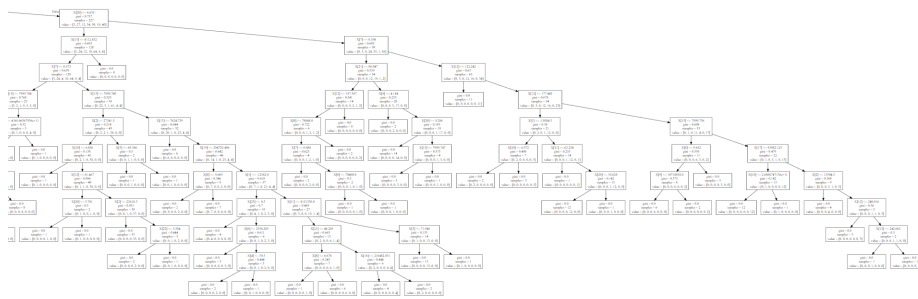


Figura 43: Porción 3 del árbol generado por Webgraphviz

20.4. Base de datos

Emotional speech synthesis database

39

* View resource name in all available languages

ISLRN: 477-238-467-792-9

ID: [ELRA-S0329](#)

This database contains the recordings of one male and one female Spanish professional speakers recorded in a noise-reduced room. It consists in recordings and annotations of read text material in neutral style plus six MPEG expressions, all in fast, slow, soft and loud speech styles. The text material is composed of 184 items including phonetically balanced sentences, digits and isolated words. The text material was the same for all the modes and styles, giving a total of 3h 59min recorded speech for the male speaker and 3h 53min for the female speaker. The Emotional speech synthesis database was created within the scope of the Interface EU funded project. [Read Less](#)

Figura 44: Base de datos de emociones encontrada

BON DE COMMANDE

Des réception de ce bon de commande, un contrat d'utilisation vous sera envoyé. Celui-ci devra être retourné à ELDA dûment signé.

Retournez ce bon de commande à l'adresse suivante:

ELDA S.A.S.
9 rue des Cordeliers
75013 Paris, France
Tél: +33 1 43 13 33 33 - Fax: +33 1 43 13 33 30
Email: map@elda.org

(* champs obligatoires)

Nom, prénom* : Hernández, María Belén
Société* : Universidad Del Valle de Guatemala
Département : Ciudad de Guatemala
Adresse* : AP 04, 11-95 zona 15, villa Hermosa III
Code postal* : 01015 ville* : Guatemala pays* : Guatemala
Tél.* : +502 41442295
Fax : _____
Email : mariaabelenb@gmail.com

Nombre de TVA ou numéro inter-communautaire (obligatoire pour les États membres de l'Union européenne) : _____

Êtes-vous membre d'ELRA : OUI NON

Vous désirez commander les ressources désignées ci-dessous :

Ref. ELRA	Désignation de la ressource	Type d'utilisation ¹	Quantité	Prix ²
<u>ELRA-S0329</u>	<u>Emotional speech synthesis database</u>	<u>R</u>	<u>1</u>	<u>A D.O</u>

¹ R = Usage de recherche, C = Usage commercial
² Frais d'envoi non inclus

Date* : _____ Signature* :

ELDA SAS au capital de 250000 EUR - 9 rue des Cordeliers 75013 Paris - France
Tél: +33 1 43 13 33 33 - Fax: +33 1 43 13 33 30 - Email: map@elda.org
SIRET 401 781 876 000 40 - NAF 4202Z R.C.S. PARIS 0 401 781 876 (53814791) - TVA N° FR-5402781876

Figura 45: Formulario para solicitud de la base de datos

animatrónico Referente a la animatrónica, la cual es la técnica que mediante el uso de mecanismos robóticos o electrónicos, simula el aspecto y comportamiento de los seres vivos empleando marionetas u otros muñecos mecánicos. Se caracterizan por tener un aspecto físico antropomórfico. Son creados para ser programados y controlados remotamente, reproducir sonido y recrear movimientos ya sean sencillos o de gran complejidad. La sofisticación de estos robots depende del uso o servicio que vayan a cumplir, ya que por ejemplo, se puede recrear únicamente el lomo de un oso o crear todo el animal completo. 11

Audacity Es un grabador y editor de sonido desarrollado por una serie de profesionales encabezados por Dominic Mazzoni en 1999. La aplicación es software libre y de código abierto, además de multiplataforma y que ha contado con numerosos desarrolladores en los últimos años. Actualmente es el editor de audio más utilizado dentro de la tecnología GNU/Linux. 59

chatbot Es un programa informático con el que es posible mantener una conversación, tanto si queremos pedirle algún tipo de información o que lleve a cabo una acción. 41

compansión En procesamiento de señales, audio analógico, telecomunicaciones y termodinámica, es un método aplicable a señales para mejorar la transmisión de las mismas en canales limitados. Está formado por dos procesos: compresión y expansión (compressing y expanding en inglés respectivamente). 20

defasaje Diferencia de fase entre dos fenómenos alternativos de la misma frecuencia. 20

float Variable que almacena valores de punto flotante, habitualmente con una precisión de 32 bits. 39

plataforma Sistema que permite la ejecución de diversas aplicaciones bajo un mismo entorno. 13

Python Es un lenguaje de programación interpretado cuya filosofía hace hincapié en una sintaxis que favorezca un código legible. Se trata de un lenguaje de programación multiparadigma, ya que soporta orientación a objetos, programación imperativa y, en menor medida, programación funcional. Es un lenguaje interpretado, usa tipado dinámico y es multiplataforma. 35

software Es una palabra que proviene del idioma inglés, pero que gracias a la masificación de uso, ha sido aceptada por la Real Academia Española. Según la RAE, el software es un conjunto de programas, instrucciones y reglas informáticas que permiten ejecutar distintas tareas en una computadora. 15

Sublime Text Es un editor de texto y editor de código fuente está escrito en C++ y Python para los plugins. Desarrollado originalmente como una extensión de Vim, con el tiempo fue creando una identidad propia, por esto aún conserva un modo de edición tipo vi llamado Vintage mode. 60

umbral Cantidad mínima necesaria para que un fenómeno sea perceptible. 18

unicode Es el estándar de codificación de caracteres universal utilizado para la representación de texto para procesamiento del equipo. Proporciona una manera consistente de codificación de texto multilingüe y facilita el intercambio de archivos de texto internacionales. 35