

RAE

1. **TIPO DE DOCUMENTO:** Trabajo de grado para optar al título de Ingeniero de Sonido.
2. **TÍTULO:** Detección y clasificación de eventos sonoros no deseados presentes en el ruido de tráfico rodado a través de redes neuronales
3. **AUTOR(ES):** Hernández Goyeneche, Edward Salomón; Medina Medina, Manuel Jaime Miguel.
4. **LUGAR:** Bogotá D.C., República de Colombia.
5. **FECHA:**
6. **PALABRAS CLAVE:** *Detección, Clasificación, Eventos sonoros, Ruido, Tráfico rodado, Redes neuronales*
7. **DESCRIPCIÓN DEL TRABAJO:** Este proyecto presenta el desarrollo de una Red Neuronal Convolutiva para la clasificación y eliminación de Eventos Sonoros No Deseados de grabaciones de tráfico rodado de la ciudad de Bogotá, Colombia. El algoritmo permite, además, el cálculo del Nivel de Presión Sonora de los mismos, y se analiza el aporte de los mismos al ruido por tráfico rodado.
8. **LÍNEA DE INVESTIGACIÓN:** Diseño de sistemas de sonido
9. **METODOLOGÍA:** El presente proyecto se desarrolla desde el método objetivista y a su vez presenta un enfoque ingenieril cuantitativo correlacionando las variables propuestas entendiendo que se quiere analizar el comportamiento de las variables independientes según las variables dependientes, dado que se realizan mediciones y evaluaciones del contenido de eventos sonoros anómalos presentes en el ruido de tráfico rodado en un punto de Bogotá.
10. **CONCLUSIONES:** El algoritmo desarrollado presenta una exactitud del 87% en la clasificación de las clases determinadas, y una precisión de más del 83% en la identificación de cada categoría. Dado un método de cálculo con error promedio cercano a 0.03 dB, se detecta una diferencia medible de Nivel de Presión Sonora al producirse eliminación de Eventos Sonoros No Deseados de grabaciones en campo.

**Detección y clasificación de eventos sonoros no deseados presentes en el ruido de tráfico
rodado a través de redes neuronales**

Manuel Jaime Miguel Medina Medina

Edward Salomón Hernández Goyeneche

Trabajo de Grado presentado para optar al título de Ingeniero de Sonido

Asesor: Belman Jahir Rodríguez Niño, Magíster (MSc) en Inteligencia Artificial.

Asesor: Óscar Esneider Acosta Agudelo, Magíster (MSc) en Acústica y Vibraciones.



Universidad de San Buenaventura

Facultad de Ingeniería (Bogotá)

Ingeniería de Sonido

Bogotá D.C., Colombia

2022

Citar/How to cite	Medina Medina <i>et al.</i> [1]
Referencia/Reference	[1] M. Medina Medina <i>et al.</i> , “Detección y clasificación de eventos sonoros no deseados presentes en el ruido de tráfico rodado a través de redes neuronales”, Trabajo de grado profesional, Ingeniería de Sonido, Universidad de San Buenaventura Bogotá (Cundinamarca), 2022.
Estilo/Style: IEEE (2020)	



Biblioteca Digital (Repositorio)

www.bibliotecadigital.usb.edu.co

Bibliotecas Universidad de San Buenaventura

Biblioteca Fray Alberto Montealegre O.F.M. - Bogotá.

Biblioteca Fray Arturo Calle Restrepo O.F.M. - Medellín, Bello, Armenia, Ibagué.

Departamento de Biblioteca - Cali.

Biblioteca Central Fray Antonio de Marchena – Cartagena.

Universidad de San Buenaventura Colombia - www.usb.edu.co

Bogotá - www.usbbog.edu.co

Medellín - www.usbmed.edu.co

Cali - www.usbcali.edu.co

Cartagena - www.usbctg.edu.co

Editorial Bonaventuriana - www.editorialbonaventuriana.usb.edu.co

Revistas científicas – www.revistas.usb.edu.co

TABLA DE CONTENIDO

Capítulo 1. Generalidades	6
1.1. Antecedentes.....	6
1.2. Planteamiento del problema	14
1.3. Justificación y pregunta de Investigación	16
1.4. Objetivo General.....	19
1.5. Objetivos Específicos	19
1.6. Alcances y Limitaciones.....	19
Capítulo 2. Marco Conceptual	21
2.1. Ruido ambiental.....	21
2.1.1. Ruido por tráfico rodado	21
2.1.2. Descriptores de ruido por tráfico rodado	22
2.1.3. Nivel Equivalente Máximo (NEM).....	22
2.1.4. Nivel Equivalente Día-Noche:	23
2.1.5. Nivel Promedio	23
2.2. Fundamentos de procesamiento digital de señales	23
2.2.1. Series y transformadas de Fourier.....	24
2.2.2. Espectrogramas	25
2.2.3. Coeficientes cepstrales en las frecuencias de Mel	25
2.3. Fundamentos de Inteligencia Artificial:	26
2.3.1. Sistemas supervisados y no supervisados	27
2.3.2. Red Recurrente.....	28
2.3.3. Red Convolucional.....	28
2.3.4. Feedforward	29
2.3.5. Matriz de confusión	29

2.3.6.	Batch size	30
2.3.7.	Epoch	30
2.3.8.	Iteración	31
2.3.9.	Precisión.....	31
2.3.10.	Exactitud.....	31
2.3.11.	Pérdida.....	32
2.3.12.	Llanura espectral.....	32
Capítulo 3. Metodología.....		34
3.1.	Recopilación de Datos	35
3.2.	Determinación de Variables	38
3.3.	Determinación de Variables y Elementos.....	38
Capítulo 4. Desarrollo de Ingeniería		41
4.1.	Equipos Utilizados.....	41
4.2.	Captura de audio e identificación de clases	43
4.3.	Separación de muestras.....	44
4.4.	Análisis de características	47
4.4.1.	Análisis de duración.....	47
4.4.2.	Análisis de nivel promedio	49
4.4.3.	Análisis de espectro en frecuencia	52
4.5.	Aleatorización de los datos	54
4.6.	Obtención fórmula para cálculo de dBSPL	57
4.6.1.	Cálculo valor RMS	67
4.7.	Desarrollo del algoritmo	68
Capítulo 5. Presentación y Análisis de Resultados		90
5.1.	Métricas de evaluación	90

5.1.1.	Análisis de precisión por categoría	91
5.1.2.	Evaluación de exactitud de algoritmo	92
5.2.	Evaluación con muestras externas	93
5.2.1.	Instrumentos utilizados	93
5.2.2.	Proceso de recolección de datos.....	97
5.2.3.	Comparación con datos obtenidos	99
5.3.	Análisis aporte de ESNDs a cálculo de Nivel de Presión Sonora.....	103
5.3.1.	Primer punto de grabación	103
5.3.2.	Segundo punto de grabación	104
Capítulo 6. Conclusiones.....		105
Capítulo 7. Referencias		108

LISTA DE TABLAS

Tabla 1. Valores de SPL y Pascales con valor de calibración 0,01. Fuente propia.	58
Tabla 2. Valores de SPL y Pascales con valor de calibración 0,05. Fuente propia.	60
Tabla 3. Complemento de ecuación respecto a valor de calibración. Fuente propia.....	62
Tabla 4. Coeficiente asociado a diferentes valores de calibración (Pascales). Fuente propia.	63
Tabla 5. Error entre predicción y datos. Fuente propia.	101

LISTA DE FIGURAS

Figura 1. Frecuencia de aparición de eventos sonoros seleccionados en audios estudiados. Fuente Propia.....	17
Figura 2. Escala Mel, representando distancia percibida entre tonos. Fuente: (Sahoo et al., 2021).	26
Figura 3. Diagrama de Red Neuronal Feedforward. Fuente: (Gupta, 2017)	29
Figura 4. Matriz de Confusión Binaria. Fuente: (Barrios, 2019).....	30
Figura 5. Diagrama de bloques de la Metodología Empleada. Fuente propia.....	34
Figura 6. Punto de vista desde el punto de grabación. Fuente propia.	44
Figura 7. Espectrograma de tres muestras de ruido de chillido de frenos. Fuente propia.	45
Figura 8. Espectrograma de tres muestras de ruido por salida de aire a presión. Fuente propia. ...	45
Figura 9. Espectrograma de tres muestras de ruido de claxon. Fuente propia.	46
Figura 10. Diagrama de caja y bigotes de distribución de longitudes para cada categoría. Fuente propia.	48
Figura 11. Diagrama de caja y bigotes de distribución de Nivel Sonoro (RMS) para cada categoría. Fuente propia.	50
Figura 12. Diagrama de caja y bigotes de distribución de llanura espectral para cada categoría. Fuente propia.	52
Figura 13. Gráfica cálculo de MatLab con valor de calibración 0,01. Fuente propia.	59
Figura 14. Gráfica cálculo de MatLab con valor de calibración 0.01 (lineal). Fuente propia.	59
Figura 15. Gráfica cálculo de MatLab con valor de calibración 0,05. Fuente propia.	60
Figura 16. Gráfica cálculo de MatLab con valor de calibración 0.05 (lineal). Fuente propia.	61
Figura 17. Gráfica cálculo de complemento fórmula. Fuente propia.	62

Figura 18. Gráfica de coeficientes (Pascuales). Fuente propia.....	63
Figura 19. Creación de rutas de entrenamiento y validación. Fuente propia	69
Figura 20. Importación de librerías. Fuente propia.	69
Figura 21. Verificación de cabeceros de archivo. Fuente propia.	70
Figura 22. Verificación lectura de archivos. Fuente propia.....	70
Figura 23. Declaración de clases. Fuente propia.	71
Figura 24. Definición de funciones de procesamiento. Fuente propia.	71
Figura 25. Definición de padding. Fuente propia.	72
Figura 26. Definición de propiedades de archivos. Fuente propia.	72
Figura 27. Definición de funciones para actualización de índices. Fuente propia.	73
Figura 28. Funciones generadoras de datos. Fuente propia.....	74
Figura 29. Función para aplicar padding. Fuente propia.	75
Figura 30. Definición red neuronal, neuronas. Fuente propia.	76
Figura 31. Funciones para matriz de confusión, y gráficas de pérdida y exactitud. Fuente propia.	77
Figura 32. Procesamiento de datasets establecidos. Fuente propia.	78
Figura 33. Impresión forma red neuronal, parte 1. Fuente propia.	79
Figura 34. Impresión forma red neuronal, parte 2. Fuente propia.	80
Figura 35. Entrenamiento de modelo. Fuente propia.	80
Figura 36. Gráfica de pérdida a lo largo de entrenamiento. Fuente propia.	81
Figura 37. Gráfica de exactitud a lo largo de entrenamiento. Fuente propia.....	81
Figura 38. Generación matriz de confusión. Fuente propia.	82
Figura 39. Creación arreglo con predicciones. Fuente propia.	82

Figura 40. Lectura y preprocesamiento de archivo nuevo. Fuente propia.....	83
Figura 41. Fragmentación y cálculo de nivel RMS de segmentos. Fuente propia.	83
Figura 42. Guardado de fragmentos en carpeta temporal. Fuente propia.....	84
Figura 43. Procesamiento y predicción de datos. Fuente propia.	84
Figura 44. Creación de arreglo y conteo de eventos sonoros. Fuente propia.	85
Figura 45. Generación de archivos derivativos. Fuente propia.	86
Figura 46. Cálculo Nivel Sonoro Equivalente para audios. Fuente propia.	87
Figura 47. Generación archivo de Excel con predicciones y niveles sonoros. Fuente propia.....	88
Figura 48. Inclusión audio depurado a archivo de Excel. Fuente propia.	89
Figura 49. Matriz de confusión (con convención de colores) para el algoritmo generado. Fuente Propia.....	91
Figura 50. Micrófono de medición miniDSP UMIK-1. Fuente: (miniDSP, n.d.)	93
Figura 51. Pistófono SVANTEK SV30A. Fuente: Dr. Jordan Design, n.d.)	94
Figura 52. Trípode metálico. Fuente: (Dixten, n.d.).....	94
Figura 53. Apeman A80. Fuente: (Memorycow, n.d.)	95
Figura 54. Captura de software HandyRec. Fuente: (Apkpure, n.d.)	95
Figura 55. Logo SVANTEK SVANPC++. Fuente: (Svantek, n.d.).....	96
Figura 56. Estación meteorológica Vantage Vue. Fuente: (Davis Instruments, n.d.)	96
Figura 57. Puntos de grabación. Fuentes: (IDECA, 2015), (Flaticon, n.d.)	97
Figura 58. Vista desde primer punto de grabación. Fuente propia.....	98
Figura 59. Vista desde segundo punto de grabación. Fuente propia.	98
Figura 60. Diagrama de caja y bigotes de error entre predicción y datos. Fuente Propia	102

RESUMEN

El ruido ambiental aumenta año tras año convirtiéndose en una preocupación creciente en las zonas urbanas y suburbanas, especialmente en las grandes ciudades, ya que no solo causa molestias a los ciudadanos, sino también efectos nocivos para las personas. La mayoría de ellos se centran en problemas relacionados con la salud, siendo especialmente preocupante el impacto del ruido en los niños, cuyo grupo de población es especialmente vulnerable. Otras investigaciones también han mostrado los efectos de la contaminación acústica en la concentración, el sueño y el estrés.

La detección de eventos de audio (DEA), o la detección de eventos de sonido, se puede utilizar en muchos escenarios, y se ha dedicado una gran cantidad de trabajo de investigación a este tema. Por otra parte, el ruido por tráfico rodado se define como la interacción del pavimento y la superficie del neumático, en la actualidad es uno de los principales contaminantes acústicos en las ciudades. El proyecto plantea la detección y clasificación de eventos sonoros no deseados presentes en el ruido por tráfico rodado a través de redes neuronales y a su vez analizar el aporte energético de dichos eventos sonoros en el ruido de tráfico rodado más aun sabiendo que los eventos aquí trabajados no hacen parte del estudio del ruido de tráfico, sino que son aquellos eventos sonoros no deseados o anómalos presentes en este tipo de ruido.

Como parte de un estudio inicial, se tomaron muestras sonoras sobre dos calles de la ciudad de Bogotá, con el propósito de hacer un conteo de eventos sonoros y delimitar el campo de acción del presente trabajo. Teniendo en cuenta los resultados, y el tiempo dispuesto para el desarrollo del presente proyecto, se decide escoger los tres eventos más recurrentes en el estudio para las muestras como lo son el aire a presión, el claxon y el sonido de frenos. El desarrollo del proyecto se centra en un punto de una de las avenidas principales de la ciudad de Bogotá, en donde se detectó la presencia significativa de los eventos sonoros no deseados mencionados.

***Palabras clave:** Detección, Clasificación, Eventos sonoros, Ruido, Tráfico rodado, Redes neuronales*

ABSTRACT

Environmental noise increases year after year becoming a growing concern in urban and suburban areas, especially in large cities, since it not only causes inconvenience to citizens, but also harmful effects for people. Most of them focus on health-related problems, with the impact of noise on children, whose population group is especially vulnerable, being of particular concern. Other research has also shown the effects of noise pollution on concentration, sleep and stress.

Audio event detection (AED), or sound event detection, can be used in many scenarios, and a great deal of research work has been devoted to this topic. On the other hand, noise from road traffic is defined as the interaction of the pavement and the tire surface, and is currently one of the main noise pollutants in cities. The project proposes the detection and classification of unwanted sound events present in road traffic noise through neural networks and at the same time analyze the energy contribution of said sound events in road traffic noise, even more so knowing that the events studied here do not are not part of the study of traffic noise, but rather are those unwanted or anomalous sound events present in this type of noise.

As part of an initial study, sound samples were taken on two streets in the city of Bogotá, with the purpose of counting sound events and delimiting the field of action of this work. Taking into account the results, and the time available for the development of this project, it is decided to choose the three most recurrent events in the study for the samples, such as air pressure, the horn and the sound of brakes. The development of the project focuses on a point of one of the main

avenues of the city of Bogotá, where the significant presence of the aforementioned unwanted sound events was detected.

Keywords: *Detection, Classification, Sound events, Noise, Road traffic, Neural networks.*

INTRODUCCIÓN

De acuerdo a la AEC, (2019), el ruido se entiende como un sonido excesivo y molesto; a muy alto nivel puede constituirse en un factor de riesgo materializado que contribuye a la pérdida temporal de la audición y su prolongación en el tiempo provocará pérdidas permanentes. El ruido es actualmente uno de los factores de contaminación auditiva más importantes en el mundo debido al desarrollo que ha tenido la sociedad. Hoy en día encontramos grandes multinacionales con enormes equipos de manufactura que les ayudan a mejorar sus procesos, pero al trabajar con este tipo de maquinaria se tiene afectaciones importantes como lo es el ruido que generan al estar en operación; más sin embargo al ser actividades industriales, se tienen espacios fijos donde se pueden realizar los respectivos tratamientos acústicos para el control del ruido que emanan. Por otra parte, nos encontramos con una de las fuentes de ruido más comunes en la sociedad, el cual es producto de estas grandes empresas de manufactura y ensamblaje como lo es el automóvil.

Los modos de transporte también han venido evolucionando según las necesidades de las personas y puede encontrarse una larga lista de ellos, unos generando mayores niveles de ruido que otros, pero todos generando aportes a los niveles de ruido presentes en las zonas urbanas, dado que los niveles de ruido por estas fuentes móviles han superado los niveles permitidos por la normativa actual, se han venido desarrollando métodos para reducir estos niveles ruido, dado que entre las fuentes de ruido, el ruido de tráfico rodado es uno de los principales contaminantes acústicos en las ciudades (Alías et al., 2018) y alrededor de un millón de años de vida sana se pierden cada año a causa del ruido en Europa occidental según el informe mundial sobre el envejecimiento y la salud del año 2015 publicado por la Organización Mundial de la Salud (OMS), la contaminación auditiva es el segundo problema de salud pública en las ciudades. De acuerdo con encuestas de Percepción Ciudadana realizadas en Colombia (Ipsos Napoleón Franco, 2013), el ruido es la tercera

preocupación ambiental de los habitantes. Uno de los mayores contaminantes por ruido en las ciudades es el tráfico rodado, cientos de carros recorriendo las calles y avenidas todos los días, unos de carga, otros de transporte masivo, otros de transporte particular pero siempre hay presencia de tráfico rodado en las calles, generando estos altos niveles de ruido.

El propósito de esta investigación es realizar un algoritmo que permita la detección de 3 clases de ruido presentes en el tráfico rodado pero que no se tienen en cuenta en los modelos de predicción de ruido actuales, como el claxon, el aire a presión utilizados en algunos vehículos y ruido de las pastillas de frenos de algunos carros, estas clases salen de análisis previos sobre la Avenida Ciudad de Cali en Bogotá. El algoritmo también pretende presentar cual sería el aporte energético que podrían llegar a tener estos sonidos en las mediciones de ruido por tráfico rodado y así poder determinar el grado de participación que tienen estos clásicos sonidos del tráfico rodado en la ciudad.

Capítulo 1. Generalidades

1.1. Antecedentes

Condiciones de tránsito vehicular y uso de un modelo para la predicción de ruido por tráfico rodado en un entorno local de la ciudad de Bogotá-Colombia

Óscar Acosta Agudelo, Carlos Montenegro Marín, Paulo Gaona García (2020)

Este artículo presenta un estudio del ruido por tráfico rodado en la ciudad de Bogotá, con el fin de comprender sus causas, y posibles motivos por los cuales las predicciones de nivel de ruido de tráfico pueden fallar. Para este fin, se toman indicadores brindados por el gobierno local (Secretaría Distrital de Movilidad y RUNT), mediciones hechas en campo y simulaciones realizadas por medio del modelo para cálculo del ruido por tráfico rodado NMPB-Routes 96. Logran establecer que, al menos bajo el uso de este sistema de simulación, las estimaciones subestiman el nivel de ruido de tráfico real en la mayor parte de las ocasiones (62.5% de las veces en el estudio). Se intuye que estas diferencias pueden estar dadas por fuentes de ruido ajenas al objeto de estudio, por lo cual un sistema de detección y eliminación de estos eventos de las muestras podría presentar números mucho más cercanos a la realidad.

LIFE DYNAMAP: Making Dynamic Noise Maps A Reality

Patrizia Belluci, Laura Peruzzi, Giovanni Zambon (2018)

El artículo trata sobre el desarrollo de LIFE DYNAMAP, un sistema para generación de mapas de ruido en tiempo real basados en sensores de bajo costo para cumplir con la Directiva Europea 2002/49/EC. Estos sensores se instalaron en la autopista A90 que rodea a Roma y un área

en Milán, Italia. Se plantea como un proyecto con una duración de 5 años, de los cuales se han trabajado algo más de 3 a la fecha de publicación del artículo. Logran obtener mapas de ruido de las zonas evaluadas y el modelo de organización de datos para el análisis estadístico que se asume es de tipo exploratorio dada la fase experimental en la que se encuentra el proyecto, aparte de un algoritmo que opera sobre el flujo de audio capturado por los sensores acústicos e identifica los eventos acústicos no relacionados con el tráfico rodado basando su diseño en una detección por clasificación para eliminar ruido no deseado (ERND) de las capturas, permitiendo limpiar estas muestras y dar una representación más típica del ruido en las diferentes zonas.

Éste trabajo es un masivo proyecto para la proyección de ruido urbano, y aunque su foco principal es la realización de mapas de ruido, es de gran ayuda pues parte de su sustento es la eliminación de eventos sonoros no deseados, objetivo del presente proyecto. Los acercamientos hechos por los autores a este tema serán de gran ayuda para el desarrollo del presente trabajo, no sólo en la parte técnica, sino también en el desarrollo de la justificación y problemática asociada con el enfoque de éste trabajo.

Detection Of Anomalous Noise Events On Low-Capacity Acoustic Nodes For Dynamic Roadtraffic Noise Mapping Within An Hybrid WASN

Rosa Ma Alsina-Pagès, Francesc Alías, Joan Claudi Socorró, Ferran Orga (2018)

Este artículo habla sobre la detección de ESNDs por medio de la implementación de Redes de Sensores Acústicos Inalámbricos (WASN). Esta red utiliza sensores de Alta y Baja capacidad; estos últimos requieren en promedio 1/6 del procesamiento que los anteriores. Para esto, se hace el uso del análisis espectral basado en coeficientes MEL. La fase experimental se hizo a partir de las bases de datos generadas a partir de los resultados de LIFE DYNAMAP.

Su conclusión principal es que la propuesta de implementar detectores de eventos de ruido anómalo para sensores acústicos de baja calidad usados en tiempo real, es viable tanto en el comportamiento de carga computacional como en la precisión de los resultados en un entorno urbano, por ende, su implementación bajaría los costos de uso manteniendo un desempeño consistente con respecto a los ANED de alta capacidad. Sus resultados son basados en los resultados de las redes de sensores de bajo costo de LIFE DYNAMAP para un proyecto híbrido, con sensores de alta y baja capacidad. Aunque el presente trabajo no presente un proyecto de esta escala o aplicación, este trabajo (junto con otros de este apartado) están enfocados al desarrollo de detectores de ESND, y la parte técnica y teórica puede ser de gran ayuda.

Weakly-Supervised Audio Event Detection Using Event-Specific Gaussian Filters And Fully Convolutional Networks

Ting-Wei Su, Jen-Yu Liu, Yi-Hsuan Yang (2017)

Artículo sobre un modelo de eliminación de eventos sonoros no deseados basado totalmente en redes convolucionales, el cual utiliza data débilmente supervisada, es decir, potencialmente con gran cantidad de ruido e información irrelevante. Para la separación de estos datos se hace uso de filtros gaussianos, y se requiere únicamente que quienes lleven a cabo el entrenamiento ingresen las etiquetas de nivel de clip para los audios, en vez de tener que clasificar y limpiar la información individualmente para el entrenamiento. Mencionan que, aunque el modelo fue desarrollado satisfactoriamente, fue entrenado por una cantidad de datos muy limitada.

Para el desarrollo eficiente de este proyecto es útil analizar métodos para agilizar el proceso de entrenamiento de la red neuronal que se utilizará para obtener los resultados deseados. De esta manera, este artículo presenta una integración perfecta entre la optimización del proceso de

clasificación de los datos de entrenamiento (por medio de filtros gaussianos que logran identificar información importante aún en una muestra sin clasificar muy ruidosa); y la aplicación de dicha optimización en un algoritmo para la detección de ESND.

De implementarse el método propuesto, la mano de obra por parte de los ejecutores del proyecto puede reducirse considerablemente, evitando el arduo proceso de clasificación manual de datos de entrenamiento.

Detection Of Anomalous Noise Events For Real-Time Road-Traffic Noise Mapping: The DYNAMAP's Project Case Study.

Rosa Ma Alsina-Pagès, Francesc Alías, Joan Claudi Socorró, Ferran Orga (2018)

Este artículo estudia y expone los diferentes métodos propuestos por Belluci, Peruzzi y Zambon en el proyecto LIFE DYNAMAP para la implementación de algoritmos detectores de ESND en los sensores de bajo costo propuestos en dicho proyecto. Además de hacer un análisis de cada acercamiento propuesto, se finaliza con un análisis de los desafíos a futuro para sistemas de generación de mapas de ruido basados en redes WASN.

Éste trabajo se enfoca en describir los hitos claves del proyecto LIFE DYNAMAP para desarrollar un proceso automático para detectar y eliminar aquellos eventos acústicos no relacionados con el ruido del tráfico en el cálculo del mapa de ruido basado en WASN. El artículo presenta temas como la detección de eventos acústicos en entornos urbanos, el detector de eventos en entornos de la vida real, la detección de ANE en tiempo real, la caracterización de la acústica, caracterización de la vida real urbana y suburbana, el impacto de los ANE en el *LAeq* y los desafíos operativos de WASN en el mundo real. Si bien dicho trabajo nos da un argumento para el desarrollo del presente proyecto, una indagación centrada en los métodos de eliminación de eventos no deseados de las muestras de ruido de tráfico tomadas puede resultar más nutritiva para el desarrollo de la propuesta aquí presentada.

Diseño e implementación de una plataforma digital bajo el concepto de Smart Cities para visualizar el impacto por ruido en los entornos locales de cinco universidades de la ciudad de Bogotá.

Iván Felipe Pérez Naranjo, Oscar Andrés Mendoza Rivera (2020)

En este documento se ve reflejado el diseño e implementación de una plataforma digital que tenga en cuenta el concepto emergente de Smart Cities en donde se observa el impacto causado por ruido en cinco zonas diferentes de la ciudad de Bogotá, Colombia y De acuerdo con los resultados que obtuvieron, muestran que entre 281 y 230 residentes están siendo expuestos a una cantidad de ruido mayor a 75 dB(A), en el horario diurno (7:01 – 21:00) analizado y según la NIOSH (National

Insitutute for Ocupational Safety and Health) una persona puede estar expuesta un máximo de 8 horas a un nivel de 85 dB(A), por ende afirman que algunos residentes de estas zonas están siendo afectados por estos niveles en fachada. El documento parte de una revisión bibliográfica que permite conocer el concepto de Smart City, el cual hace referencia a una forma de desarrollo urbano que tiene en cuenta la sostenibilidad y que satisface las necesidades de instituciones, empresas y habitantes en todos los aspectos posibles, ya sea económico, ambiental y social, empleando para ello las tecnologías de la información y la comunicación (TIC). Se trabaja el concepto de ruido ambiental el cual es un sonido indeseado producto de actividades desarrolladas dentro de un espacio urbano que son capaces de afectar la calidad de vida de las personas. Para realizar la plataforma se realizaron protocolos para la medición de ruido ambiental en diversas universidades según la normativa nacional e internacional correspondiente.

Detection and Classification of Acoustic Scenes and Events

Dan Stowell *, Dimitrios Giannoulis, Emmanouil Benetos, Mathieu Lagrange y Mark D. Plumbley (2015)

En este artículo se habla sobre el estado del arte en la clasificación automática de escenas de audio y la detección y clasificación automática de eventos de audio. Evidencian trabajos previos, así como el estado del arte representado por las presentaciones al desafío de varios grupos de investigación. También proporcionan detalles sobre la organización del desafío, de modo que la experiencia como anfitriones del desafío que pueda ser útil para aquellos que organizan desafíos en dominios similares. Sus resultados muestran que la mayoría de sistemas presentados usaban entrenamiento discriminativo y muchos de los mejores ejecutantes usaban un SVM o Support Vector Machine como clasificador final, dado que es un algoritmo de aprendizaje supervisado que

se utiliza en muchos problemas de clasificación y regresión, incluidas aplicaciones médicas de procesamiento de señales y reconocimiento de imágenes y voz.

A su vez se muestran tareas de clasificación de escenas acústicas y detección de eventos de sonido dentro de una escena, ambos han sido estudiados en la literatura reciente. Se analiza su relación con otras tareas de escucha de la máquina y se describen los enfoques estándar adoptados. Luego, hablan acerca de las campañas de evaluación recientes en la escucha de máquinas, que establecen el contexto para la campaña aquí expuesta.

Eliminación de eventos anómalos para la generación automatizada de mapas de ruido del tráfico

Rosa Ma Alsina-Pagès, Francesc Alías, Joan Claudi Socoró, Ferran Orga, Roberto Benocci, Giovanni Zambon (2018)

En este trabajo, se comparan la eliminación manual y automática de ANE. Este último se basa en dos versiones del Detector de eventos de ruido anómalo (ANED) diseñado para detectar ANE dentro de un WASN en tiempo real como un clasificador de dos clases. Los experimentos en 4 horas y 44 minutos de datos de audio de la vida real muestran tasas de error similares entre todos los métodos de anotación considerados. Sin embargo, el análisis detallado de los experimentos revela, por un lado, anotaciones manuales inconsistentes en ciertas situaciones de etiquetado no ANE, donde se observan decisiones no coincidentes según expertos; y, por otro lado, la disminución de la precisión general de los enfoques basados en ANED debido al gran número de falsas alarmas en el caso de la clase RTN. Por lo tanto, aunque los resultados demuestran la viabilidad de la eliminación de ANE, se deben realizar más investigaciones para seguir mejorando

la automatización de la anotación de ANE. Los experimentos en 4h y 44min de datos de audio de la vida real muestran tasas de error similares entre todos los métodos de anotación considerados.

Estudio del rendimiento de un algoritmo de superación generalizado para la detección de eventos de ruido provocados por el tráfico vial.

Al Brown, Bert De Coensel (2018)

Este artículo investiga el desempeño de un algoritmo para la detección de eventos de ruido, construido con base en la literatura sobre eventos de ruido por tráfico rodado. Para ello es acoplado a un modelo de emisión que tiene en cuenta las distribuciones de los niveles de potencia acústica de los vehículos individualmente, se utiliza para simular historias de tiempo de una hora del nivel de ruido en la proximidad de una carretera, para un conjunto exhaustivo de condiciones de flujo de tráfico, composición y distancia de propagación en ubicaciones sin blindaje. Luego se evalúa la validez y confiabilidad del número de eventos de ruido detectados por el algoritmo generalizado en estos historiales de tiempo de una hora para un rango de conjuntos de parámetros de algoritmo. Al descartar conjuntos de parámetros que no resulten en un algoritmo que devuelva recuentos válidos o confiables, y al examinar la redundancia en los restantes, se identifica un pequeño número de conjuntos de parámetros representativos, que puede resultar útil en la construcción de indicadores basados en eventos complementarios a las medidas de energía equivalente al ruido de tráfico vial.

1.2. Planteamiento del problema

El ruido ambiental aumenta año tras año y se ha convertido en una preocupación creciente en las zonas urbanas y suburbanas, especialmente en las grandes ciudades, ya que no solo causa molestias a los ciudadanos, sino también efectos nocivos para las personas. La mayoría de ellos se centran en problemas relacionados con la salud, siendo especialmente preocupante el impacto del ruido en los niños, cuyo grupo de población es especialmente vulnerable. Otras investigaciones también han mostrado los efectos de la contaminación acústica en la concentración, el sueño y el estrés. Finalmente, cabe mencionar que la exposición al ruido no solo afecta a la salud, sino que también puede afectar a aspectos sociales y económicos, (Alías et al., 2018).

Según la Organización Mundial de la Salud (OMS), al menos un millón de años de vida sana se pierden cada año a causa del ruido relacionado con el tráfico en Europa occidental. Se afirmó que el ruido del transporte por sí solo representa el 36% de la carga total de morbilidad atribuible a la planificación urbana, un porcentaje incluso mayor que el causado por la contaminación atmosférica en Barcelona, (Alías et al., 2018). Asimismo, se ha descubierto que vivir con una exposición continua a altos niveles de ruido de tráfico es perjudicial para la salud humana, ya que afecta la calidad de vida de las personas que viven en áreas urbanas y suburbanas, (Alsina-Pagès et al., 2018). De acuerdo a Carlos Ortega y Rodrigo Sepúlveda, a través de un reportaje de El Tiempo (2019), en Bogotá el 80% del ruido es generado por los vehículos en su desplazamiento, emanando niveles entre 70 y 80 decibeles, los cuales generalmente se encuentran por encima de los valores definidos en la Resolución 0627 de 2006, la cual es la norma nacional.

De acuerdo con la Agencia Ambiental Europea (2014), se define el tráfico rodado como la combinación del ruido por rodadura y el ruido por propulsión. El ruido por rodadura es generado

por la interacción de los neumáticos de un vehículo con el pavimento, estimando la velocidad en que éste hace un aporte significativo a partir de los 40 km/h para la mayoría de vehículos, y 70 km/h para camiones; por debajo de dichas velocidades, se considera el aporte más importante al ruido por rodadura el ruido por propulsión, generado por el motor, exhosto, y transmisión del vehículo. En la actualidad se cuenta con varios modelos alrededor del mundo para la predicción de ruido de tráfico rodado como por ejemplo el modelo NMPB Routes 2008 (Francia), RLS 90 (Alemania) y el CNOSSOS (Unión Europea) entre otros, cada uno de estos métodos presenta limitaciones y formas características del modelo propio según la región donde es usado. Hoy en día Colombia no se cuenta con un modelo de predicción propio que permita hacer la predicción de este tipo de ruido y es por eso que deben ajustarse estas al modelo que más se encuentre semejante a las características del tráfico en Colombia y más específicamente en Bogotá donde las vías son un caos y pueden verse volquetas de modelos antiguos transitando por zonas residenciales o compartiendo el mismo carril de un automóvil. Dicho esto, es evidente que las necesidades no son las mismas que las que se contemplan en los modelos europeos actuales, lo cual podría traer diferencias entre los valores calculados bajo los modelos y los valores medidos en campo debido a que las características propias del ambiente bogotano no son las mismas que en Europa.

La detección de eventos de audio (DEA), o la detección de eventos de sonido, se puede utilizar en muchos escenarios, y se ha dedicado una gran cantidad de trabajo de investigación a este tema. Por ejemplo, en la seguridad del hogar pueden alertarnos cuando se producen eventos como gritos o disparos, (Su et al., 2017), también en el reconocimiento de especies animales presentes en un ecosistema y en este caso detectar sonidos anómalos presentes en el ruido de tráfico rodado para en un futuro mejorar las condiciones del análisis de los resultados en las mediciones de ruido por tráfico rodado. Es por esto que el proyecto plantea la detección y clasificación de eventos sonoros

no deseados presentes en el ruido por tráfico rodado a través de redes neuronales y a su vez analizar el aporte energético de dichos eventos sonoros en el ruido de tráfico rodado más aun sabiendo que los eventos aquí trabajados no hacen parte del estudio del ruido de tráfico, sino que son aquellos eventos sonoros no deseados o anómalos presentes en este tipo de ruido.

1.3. Justificación y pregunta de Investigación

Para representar de manera apropiada el paisaje sonoro (entendido, de acuerdo a Southworth (1969), como el ambiente sonoro de una ciudad al ser percibido por el ser humano) de una ciudad en específico, es de vital importancia identificar qué lo compone, y por qué no todo elemento sonoro que ocurra dentro de la misma puede considerarse como un aporte a su paisaje. En este orden de ideas, ha de identificarse que hay elementos del sonido de una ciudad que son constantes y relativamente invariables en el tiempo. Se alinea esto con las ideas explicadas en el gran proyecto LIFE DYNAMAP (Belluci, Peruzzi, Zambon, 2018), pues se argumenta allí que para dar una representación en tiempo real del paisaje sonoro de las ciudades allí estudiadas (Roma y Milán, Italia), es esencial separar todos aquellos elementos los cuales no hagan parte del ruido de tráfico rodado.

Para poder crear un algoritmo que detecte correctamente y notifique eventos sonoros no deseados es muy importante no sólo definir qué compone este tipo de fenómenos, sino el criterio mediante el cual se trabajarán. Dada la diferencia en locación de aplicación de otros trabajos sobre los que se ha trabajado con la ciudad de Bogotá, usar las determinaciones de Eventos Sonoros No Deseados hechas por dichos trabajos es posiblemente poco representativo del ambiente sonoro de la ciudad de Bogotá. Algunas de estas discrepancias pueden darse por el tipo de vehículos que transitan en la ciudad, con Bogotá contando con una edad promedio mayor en sus vehículos que

aquellos en las ciudades estudiadas en el trabajo citado. En concreto, de acuerdo a la Asociación Nacional de Movilidad Sostenible (El Espectador, 2022) el parque automotriz en Colombia tiene, en promedio, 17.5 años para automóviles, 22 años para buses, y 25 años para volquetas; en contraste con los promedios de 11.8, 12.8 y 14.1 años para las mismas categorías (ACEA, 2022). Además, el tráfico en horas pico de Bogotá es presumiblemente de mayor densidad que aquel en ciudades como Milán o Roma, e incluye los ya mencionados vehículos pesados.

De esta manera, se definen unos criterios de inclusión y exclusión iniciales, y el método mediante el cual se llevará a cabo la selección de otros. Vale resaltar que, para efectos del presente trabajo, la inclusión hace referencia a aquellos elementos que se considerarán Eventos Sonoros No Deseados; y exclusión a aquellos que hacen parte del panorama sonoro de Bogotá, necesarios para el correcto análisis del ruido de tráfico rodado.

Como parte de un estudio inicial, se tomaron muestras sonoras sobre dos calles de la ciudad de Bogotá, con el propósito de hacer un conteo de eventos sonoros y delimitar el campo de acción del presente trabajo. En la Figura 1, se puede apreciar el resultado del conteo manual obtenido.

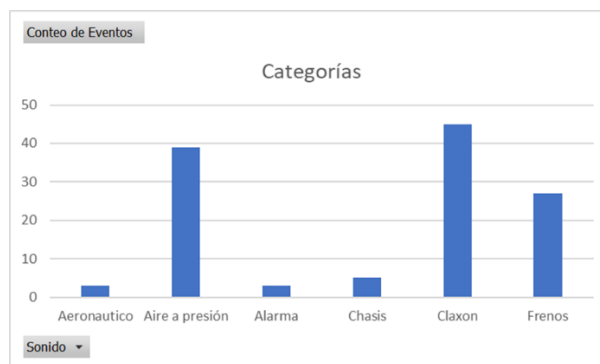


Figura 1. Frecuencia de aparición de eventos sonoros seleccionados en audios estudiados. Fuente Propia.

Se analizaron en total 30 minutos, anotando diferentes tipos de eventos sonoros dentro de los mismos. La categoría de “Aire a presión” incluye frenos de aire, sonido de descompresión de turbinas y turbos. Como se puede apreciar en la gráfica, los eventos de claxon (también conocido como pito), aire a presión y frenos (mayormente dados por el chillido de las pastillas contra el rotor al frenar) son los más frecuentes.

Teniendo en cuenta estos resultados, y el tiempo dispuesto para el desarrollo del proyecto, se decide escoger los tres eventos más recurrentes en el estudio realizado para las muestras, es decir, el Aire a presión, el Claxon y el sonido de Frenos.

Es allí donde se plantea la siguiente pregunta de investigación:

¿Cómo identificar eventos sonoros presentes en el ruido de tráfico rodado tales como claxon, ruido de frenos y salidas de aire a presión en un punto de Bogotá?

1.4. Objetivo General

Clasificar los eventos sonoros no deseados presentes en el ruido de tráfico rodado diurno tales como claxon, ruido de frenos y salidas de aire a presión en un punto de Bogotá por medio de redes neuronales.

1.5. Objetivos Específicos

Identificar los eventos sonoros distintos al tráfico rodado tales como claxon, frenos y salidas de aire a presión, por medio de un análisis basado en duración, nivel máximo promedio y espectro en frecuencia.

Desarrollar un algoritmo para la detección de eventos sonoros distintos al tráfico rodado haciendo uso de una red neuronal artificial.

Evaluar la precisión del algoritmo en la clasificación de Eventos Sonoros No Deseados distintos al tráfico rodado mediante el uso de una matriz de confusión.

1.6. Alcances y Limitaciones

Alcances:

- El algoritmo permitirá la detección y notificación de los ESNDs estipulados en el objetivo general, en grabaciones de audio en un punto de la ciudad de Bogotá.
- Durante la presente investigación, no se hará eliminación de los eventos sonoros clasificados en los audios trabajados.

- El algoritmo que será implementado estará basado en redes neuronales artificiales usado como base, para la extracción de características, los coeficientes cepstrales en las frecuencias de Mel.
- Las grabaciones de audio se realizarán en condiciones de mayor tráfico, el cual depende directamente de la hora del día en el cual se hagan las capturas de muestras. Por ejemplo, si hay presencia de represamientos vehiculares, donde algunos eventos sonoros como la salida de aire a presión, los frenos y el claxon pueden tener mayor frecuencia de ocurrencia.

Limitaciones:

- El tiempo total de registro de audio en campo dependerá directamente de la cantidad y variedad de eventos sonoros que se presenten durante la toma de muestras.
- El desarrollo del proyecto se centra en un punto de una de las avenidas principales de la ciudad de Bogotá, en donde se detectó la presencia significativa de eventos sonoros no deseados. Lo anterior, debido a la extensión geográfica de la ciudad de Bogotá.
- El registro de los audios se realizará en intervalos ininterrumpidos de 15 minutos, duración escogida acorde al Artículo 5 de la Resolución 0627 de 2006, los cuales serán obtenidos de visitas al punto de estudio, en donde las condiciones climáticas tales como lluvia pueden extender el tiempo total de captura de información.

Capítulo 2. Marco Conceptual

2.1. Ruido ambiental

Uno de los principales problemas en la actualidad es el ruido ambiental debido a la gran cantidad de fuentes sonoras que existen actualmente en las calles como lo son las fábricas, las discotecas, las personas y en el presente caso de estudio, los vehículos que son la fuente de contaminación auditiva más común en las ciudades principales esto dada la cantidad de vehículos que hay en las vías todos los días, Según el informe de Andemos publicado por la revista Semana, durante el mes de septiembre de 2019 se registraron 22.904 vehículos nuevos en el país, lo que representa un incremento del 18,3%, lo cual brinda un panorama de la sobrepoblación de automóviles y por ende un aumento en los niveles de ruido por tráfico rodado en las ciudades.

Según el ministerio de ambiente de Bogotá por medio de la publicación en su página web se entiende por ruido, cualquier sonido molesto, lo que le asigna a este contaminante un carácter de percepción. Motivo por el cual, para conocer su verdadero impacto en el ambiente y salud, se debe evaluar de manera metódica y según el marco normativo ambiental vigente. Dicho esto, el ministerio de ambiente de Bogotá agrega que el ruido ambiental evalúa, cómo es el impacto ambiental que las fuentes de emisión sonora afectan a la población urbana inmersa en el ambiente sonoro urbano.

2.1.1. Ruido por tráfico rodado

El ambiente urbano ha tomado gran relevancia a partir del elevado uso de medios de transporte, a raíz de esto se empieza a hablar del ruido por tráfico rodado, el cual se hace más presente en las principales ciudades dado el desarrollo de las ciudades y su crecimiento poblacional, es importante

resaltar que en este tipo de ruido hace presencia cuando los vehículos están en marcha a velocidades de entre 50 y 80km/h, dado que es allí donde es notoria la interacción entre el motor, las llantas y el pavimento. Si las velocidades son menores a 50km/h hace presencia el ruido de propulsión dado por el motor netamente y en caso contrario si las velocidades superan los 80km/h hace mayor presencia el ruido de origen aerodinámico (Sandoval, 2005).

2.1.2. Descriptores de ruido por tráfico rodado

Sound Exposure Level (SEL): El valor SEL se define como el nivel de sonido continuo durante un periodo que contiene la misma energía que el evento sonoro individual completo dado en decibeles (dB). Utiliza tres tipos de constantes de tiempo: Impulse, Fast y Show. En particular, el SEL se obtiene empleando la constante de tiempo impulsiva se denomina IEL. Así, para un evento sonoro que transcurre durante un intervalo de tiempo T, la relación entre el SEL y es: (Sandoval, 2005)

$$SEL = L_{eq}(A) + 10 \log \left(\frac{T}{1s} \right) [dB] \quad (1)$$

2.1.3. Nivel Equivalente Máximo (NEM)

El L_{max} es el máximo valor de nivel de presión sonora registrado durante el tiempo de estudio dado en decibeles (dB), cuando se habla de eventos sonoros individuales, el valor instantáneo máximo del mismo es un parámetro importante. El valor L_{max} debe medirse con la constante de

tiempo *Fast* ya que así se obtiene una correlación con una percepción similar a la del oído humano. (Sandoval, 2005)

2.1.4. Nivel Equivalente Día-Noche:

Debido a la gran variabilidad en los niveles de ruido en el día y en la noche, este indicador L_{dn} da una idea de los niveles de ruido a lo largo de las 24 horas del día. Este indicador viene del $Leq(A)$ para un periodo de 24 horas con una penalización de 10dB para los niveles equivalentes medidos en la noche según el ruido por tráfico vehicular. Para la ciudad de Bogotá según la resolución 0627 de 2006 los horarios son diurno (7:01 am – 21:00 pm) y nocturno (21:01 pm – 7:00 am), este indicador está dado por: (Sandoval, 2005)

$$L_{dn} = 10 \log \left[\frac{1}{24} \left(16 \cdot 10^{\frac{L_d}{10}} + 8 \cdot 10^{\frac{L_n+10}{10}} \right) \right] [dB] \quad (2)$$

2.1.5. Nivel Promedio

Para este indicador solo se calcula la media aritmética de los valores instantáneos de $L_p(A)$ de la siguiente forma: (Sandoval, 2005)

$$L_{Prom}(A) = \frac{1}{n} \cdot \sum_{i=1}^n L_{p_i}(A) [dB] \quad (3)$$

Donde n es el número total de valores de $L_p(A)$ y $L_p(A)$ es cada uno de los niveles a promediar.

2.2. Fundamentos de procesamiento digital de señales

En general, toda señal contiene información que se desea extraer o modificar de acuerdo a los requerimientos de cada aplicación particular. Sismógrafos, por ejemplo, registran señales sísmicas que contienen información sobre intensidad y características espectrales de los sismos, con ayuda de las cuales pueden determinarse entre otras cosas la ubicación de epicentros y la naturaleza de los sismos. Las señales electrocardiográficas permiten al médico determinar el estado del corazón de sus pacientes. Las señales son representadas por funciones matemáticas de una o más variables.

Una señal de voz, por ejemplo, puede representarse como una función de una variable temporal $f(t)$, mientras que imágenes se pueden considerar como funciones de dos variables espaciales $f(x, y)$.

2.2.1. Series y transformadas de Fourier

Las series de Fourier son una herramienta matemática básica empleada para analizar funciones periódicas a través de la descomposición de dicha función en una suma infinita de funciones sinusoidales mucho más simples. Un objetivo básico para las series de Fourier es que toda función periódica de período T puede ser expresada como una suma trigonométrica de senos y cosenos del mismo periodo T .

Se pueden representar como:

$$\frac{a_0}{2} + \sum_{n=1}^{\infty} \left[a_n \cos \frac{2n\pi}{T}t + b_n \sin \frac{2n\pi}{T}t \right] \quad (4)$$

$$f(t) = \frac{a_0}{2} + \sum_{n=1}^{\infty} (a_n \cos \omega_n t + b_n \sin \omega_n t) \quad (5)$$

su forma compleja es:

$$f(t) \sim \sum_{n=-\infty}^{\infty} c_n e^{2\pi i \frac{n}{T}t}. \quad (6)$$

La transformada de Fourier es una herramienta matemática empleada para transformar señales entre el dominio del tiempo (o espacial) y el dominio de la frecuencia, que tiene muchas aplicaciones en la física y la ingeniería. A diferencia de las Series de Fourier, puede ser utilizada para analizar señales no periódicas, en un sentido general, cualquier señal. Es reversible, siendo capaz de transformarse de un dominio a otro. En el caso de una función periódica en el tiempo (por

ejemplo, un sonido musical continuo, pero no necesariamente sinusoidal), la transformada de Fourier se puede simplificar para el cálculo de un conjunto discreto de amplitudes complejas, llamado coeficientes de las series de Fourier. Estos coeficientes representan el espectro de frecuencia de la señal del dominio-tiempo original.

puede ser representada de la siguiente manera:

$$g(\xi) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{+\infty} f(x) e^{-i\xi x} dx \quad (7)$$

2.2.2. Espectrogramas

Los espectrogramas consisten en la representación gráfica del espectro de frecuencias de la emisión sonora. Los espectrogramas pueden revelar rasgos, como altas frecuencias o modulaciones de amplitud, como tal el espectrograma representa el tiempo sobre el eje horizontal, la frecuencia sobre el eje vertical y la amplitud de las señales mediante una escala de grises o de colores. Las representaciones del espectro pueden ayudar a entender mejor de forma visual la información frecuencial de una señal en el dominio temporal dada su representación por colores al modular la amplitud y la frecuencia.

2.2.3. Coeficientes cepstrales en las frecuencias de Mel

Los coeficientes cepstrales se derivan de la transformada de Fourier, pero la particularidad básica es que en los MFCC las bandas de frecuencia están situadas logarítmicamente, según la escala Mel, en la que el punto de referencia se define equiparando un tono de 1000 Hz., 40 dBs por encima del umbral de audición del oyente, con un tono de 1000 mels, tal y como se muestra en la siguiente figura:

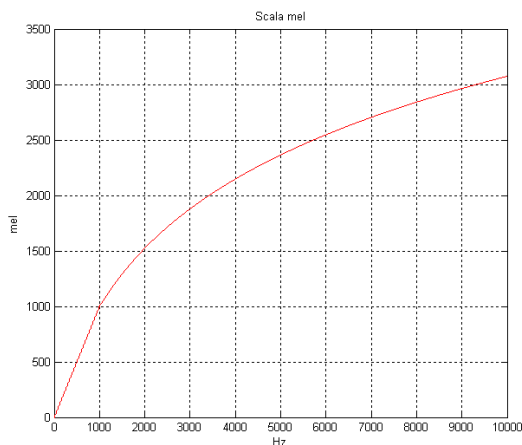


Figura 2. Escala Mel, representando distancia percibida entre tonos. Fuente: (Sahoo et al., 2021).

Los MFCCs son una característica ampliamente usada en el reconocimiento automático del discurso y fueron introducidos por Davis y Mermelstein en los años 80 y han sido el estado del arte desde entonces para todo aquello que tiene que ver con reconocimiento de voz y detección de eventos sonoros.

Los MFCCs se calculan comúnmente al separar la señal en pequeños tramos. A cada tramo aplicarle la Transformada de Fourier discreta y obtener la potencia espectral de la señal. Aplicar el banco de filtros correspondientes a la Escala Mel al espectro obtenido en el paso anterior y sumar las energías en cada uno de ellos. Tomar el logaritmo de todas las energías de cada frecuencia mel. Aplicarle la transformada de coseno discreta a estos logaritmos.

2.3. Fundamentos de Inteligencia Artificial:

Según Lasse Rouhiainen la IA es la capacidad de las máquinas para usar algoritmos, aprender de los datos y utilizar lo aprendido en la toma de decisiones tal y como lo haría un ser humano. Actualmente las tecnologías basadas en la IA ya están siendo utilizadas para ayudar a los humanos a beneficiarse de mejoras significativas y disfrutar de una mayor eficiencia en casi todos los ámbitos

de la vida. Pero el gran crecimiento de la IA también apremia a estar atentos para prevenir y analizar las posibles desventajas directas o indirectas que pueda generar la proliferación de la IA.

En el último tiempo la implementación de la inteligencia artificial ha estado muy presente en el reconocimiento de imágenes estáticas, clasificación y etiquetado de las mismas dado las medidas de seguridad adoptadas en el mundo entero, a su vez las tendencias mundiales han llevado a las grandes industrias a implementar este tipo de algoritmos para hallar las posibles necesidades del futuro y con esto hallando mejoras del desempeño de la estrategia algorítmica comercial, y como hoy día es evidente por medio de los imponentes algoritmos de las redes sociales que detectan y clasifican los gustos y necesidades de las personas según sus búsquedas, también estos algoritmos realizan labores de mantenimiento y protección contra amenazas entre muchas otras actividades desarrolladas a partir de algoritmos basados en IA.

De los algoritmos de inteligencia artificial parten dos grandes ramas llamadas Machine Learning (Aprendizaje Automático) y Deep Learning (Aprendizaje Profundo). El Machine Learning es una forma analítica de resolver problemas mediante la identificación, la clasificación o la predicción. Los algoritmos aprenden de los datos introducidos y luego utilizan este conocimiento para sacar conclusiones de nuevos datos. El Deep Learning como concepto es muy similar al Machine Learning, pero usa algoritmos distintos. Mientras que el Machine Learning trabaja con algoritmos de regresión o con árboles de decisión, el Deep Learning usa redes neuronales que funcionan de forma muy parecida a las conexiones neuronales biológicas del cerebro.

2.3.1. Sistemas supervisados y no supervisados

En el aprendizaje supervisado, los algoritmos trabajan con datos “etiquetados” (labeled data), intentado encontrar una función que, dadas las variables de entrada (input data), les asigne la etiqueta de salida adecuada. El algoritmo se entrena con un “histórico” de datos y así “aprende” a

asignar la etiqueta de salida adecuada a un nuevo valor, es decir, predice el valor de salida. (Simeone, 2018).

El aprendizaje no supervisado tiene lugar cuando no se dispone de datos “etiquetados” para el entrenamiento. Sólo se conocen los datos de entrada, pero no existen datos de salida que correspondan a un determinado input. Por tanto, sólo es posible describir la estructura de los datos, para intentar encontrar algún tipo de organización que simplifique el análisis. Por ello, tienen un carácter exploratorio.

2.3.2. Red Recurrente

La red neuronal recurrente está dada principalmente por la multiplicación de matrices, son una clase de redes para analizar datos de series temporales permitiendo tratar la dimensión de tiempo. Si se tiene en cuenta una sola neurona de esta red puede entenderse que la misma compuesta por una entrada que produce una salida y esa salida es enviada a sí misma como una retroalimentación, siguiendo esta secuencia una capa de neuronas recurrentes se pueden implementar de tal manera que las neuronas reciben dos entradas como lo son la capa anterior y la salida de la misma capa, lo cual va a alimentando la red.

2.3.3. Red Convolutiva

Las redes neuronales convolucionales constituyen actualmente el estado del arte de varios problemas de visión computacional, dado su buen desempeño en problemas de reconocimiento e interpretación en imágenes y video. Su capacidad para actuar adecuadamente en estos contextos está basada en características fundamentales: conexiones locales, pesos compartidos, pooling y el uso de una gran cantidad de capas. El propósito de CNN es extraer todas las características de una imagen y luego usar dichas características para detectar o clasificar los objetos en una imagen. Los parámetros de los filtros que se pueden aprender en estas capas; se ajustarán y optimizarán junto

con los componentes de clasificación para minimizar el error de clasificación total. (detección de equipos de protección)

2.3.4. Feedforward

De acuerdo a Ranschaert, Morozov y Algra (2019), una red neuronal feedforward es aquella en la cual las conexiones entre unidades no forman un ciclo. Una red feedforward tiene una capa de entrada, capas ocultas intermedias, y una capa de salida y la información viaja entre estas capas en una sola dirección – de la capa de entrada a la de salida, jamás devolviéndose.

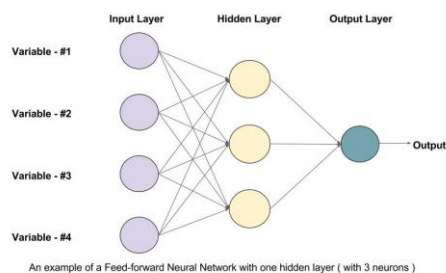


Figura 3. Diagrama de Red Neuronal Feedforward. Fuente: (Gupta, 2017)

2.3.5. Matriz de confusión

En el ámbito de la inteligencia artificial una matriz de confusión es una herramienta que permite evidenciar el desempeño de un algoritmo de aprendizaje supervisado. En la matriz de confusión se cuenta con columnas y filas por cada clase del algoritmo, las columnas indican las clases y las filas indican la instancia real en la clase, es decir, se pueden evidenciar los aciertos y

los errores que está teniendo el algoritmo a la hora de realizar el proceso de aprendizaje con los datos que hayan sido ingresados.

VALORES PREDICCIÓN	Verdaderos positivos	Falsos Positivos
	Falsos Negativos	Verdaderos Negativos
	VALORES REALES	

Figura 4. Matriz de Confusión Binaria. Fuente: (Barrios, 2019)

2.3.6. Batch size

Para hacer eficiente el procesamiento de un dataset, el cual suele contener miles, decenas de miles o, en algunos casos, millones de muestras, es importante dividirlo en secciones más pequeñas que puedan ser estudiadas por el algoritmo de manera individual. El tamaño de dichas secciones se conoce como batch size, definido en un número entero de muestras.

Por ejemplo, si se tiene un dataset con un tamaño de 10.000 muestras, y se define un batch size de 250, el algoritmo dividirá el dataset en 40 secciones de 250 muestras, secciones las cuales analizará una por una, completando un epoch.

2.3.7. Epoch

Epoch, o época, es el número de repeticiones que se indica al algoritmo pasar por todo el set de datos antes de detener el proceso de entrenamiento. Tras cada época, el algoritmo ajusta los pesos asociados a sus neuronas, con el fin de maximizar la precisión y exactitud, y minimizar la

pérdida. Se entiende pues que, al aumentar el número de épocas, el desempeño del algoritmo mejora, al igual que el tiempo de ejecución del mismo.

2.3.8. Iteración

Iteración hace referencia al número de batches necesarios para completar un epoch. En el ejemplo presentado en el apartado 2.3.6, el número de batches necesarios para completar una época es 40, dado el batch size de 250. Dado esto, puede encontrarse el número de iteraciones como la división entre el número total de muestras y el batch size.

2.3.9. Precisión

La Precisión se refiere a la dispersión del conjunto de valores obtenidos a partir de mediciones repetidas de una magnitud. Cuanto menor es la dispersión mayor la precisión. Se representa por la proporción de verdaderos positivos dividido entre todos los resultados positivos (tanto verdaderos positivos, como falsos positivos).

$$\frac{VP}{VP+FP} \quad (8)$$

2.3.10. Exactitud

La Exactitud o Accuracy se refiere a lo cerca que está el resultado de una medición del valor verdadero. En términos estadísticos, la exactitud está relacionada con el sesgo de una estimación. Se representa como la proporción de resultados verdaderos (tanto verdaderos positivos (VP) como verdaderos negativos (VN)) dividido entre el número total de casos examinados (verdaderos positivos, falsos positivos, verdaderos negativos, falsos negativos).

$$\frac{VP+VN}{VP+FP+VN+FN} \quad (9)$$

2.3.11. Pérdida

En términos generales, la pérdida es un concepto de optimización matemática que define un valor numérico asociado a una multitud de parámetros asociados a un modelo a evaluar, mediante el cual puede evaluarse con facilidad el “costo” o imprecisión del mismo dados cambios en sus parámetros. Es derivada de la función objetiva, la cual puede ser usada para maximizar (ganancia) o minimizar (pérdida) el resultado de un modelo.

En el contexto de la Inteligencia Artificial, la pérdida es utilizada como una métrica numérica indicativa del costo computacional de las operaciones erróneas o imprecisas dados los parámetros actuales, en este caso haciendo referencia a los pesos asignados a las neuronas de la red neuronal. Esta pérdida es utilizada por el algoritmo para ajustar dichos pesos, con el fin de reducir el valor a su mínima expresión.

2.3.12. Llanura espectral

De acuerdo a Boashash (2016), la llanura espectral, planicie espectral, o entropía de Wiener es una medida de la distribución energética en el dominio de la frecuencia. Un valor alto representa alta uniformidad del espectro, mientras que valores bajos representan la concentración de la energía en una banda muy estrecha.

En términos generales, la llanura espectral se calcula como la media geométrica de la Transformada de Fourier de una función, normalizada por su media aritmética, de la siguiente manera:

$$SF = M \frac{(\prod_{k=1}^M |Z[k]|)^{1/M}}{(\sum_{k=1}^M |Z[k]|)} \quad (10)$$

Donde SF es la llanura espectral; $Z[k]$ es la Transformada de Fourier de la función $z[n]$, función analítica (i.e., función compleja sin valores negativos de frecuencia) de una señal real $x[n]$; y M es la longitud de $Z[k]$.

El rango de valores obtenido a través de dicha fórmula es entre 0 y 1, con un valor de 0 indicando concentración total de la energía en una frecuencia, y 1 indicando una distribución uniforme en todo el ancho de banda. Los valores resultantes varían bastante en magnitud, por lo cual se acostumbra convertir el valor de SF a dBFS, creando una escala de $-\infty$ a 0.

Capítulo 3. Metodología



Figura 5. Diagrama de bloques de la Metodología Empleada. Fuente propia.

Desde el punto de vista epistemológico el presente proyecto se desarrolla desde el método objetivista y a su vez presenta un enfoque ingenieril cuantitativo correlacionando las variables propuestas entendiendo que se quiere analizar el comportamiento de las variables independientes

según las variables dependientes, dado que se realizan mediciones y evaluaciones del contenido de eventos sonoros anómalos presentes en el ruido de tráfico rodado en un punto de Bogotá.

Las categorías de trabajo halladas para el proyecto salen del análisis audiovisual de grabaciones previas en 2 puntos diferentes de la Carrera 68 con Calle 22 realizadas junto al profesor Óscar Acosta, allí se analizaron videos de una duración promedio de 2 minutos y una duración total de video de 35 minutos, donde se evidenciaron los diferentes eventos sonoros anómalos presentes en el ruido de tráfico rodado que hacen parte de la presente investigación y que se decantaron en el análisis ya mencionado generando las categorías base para el presente proyecto, siendo estos claxon, frenos y aire a presión.

3.1. Recopilación de Datos

- Identificar los eventos sonoros distintos al tráfico rodado tales como claxon, frenos y salidas de aire a presión, por medio de un análisis basado en duración, nivel máximo promedio y espectro en frecuencia.
 - Toma de muestras sonoras en campo, se harán visitas de campo para determinar que la grabación se realice en un punto que cumpla con las condiciones óptimas para obtener la mayor cantidad de muestras por grabación; tal como el flujo vehicular y la presencia de vehículos de carga, así como ausencia de lluvia.
 - Con base en los antecedentes se proyecta tener un total de 2400 muestras en un total de 10 horas grabación divididas en grabaciones de mínimo 15 min.

- Separación de muestras - 800 mínimo por cada categoría (claxon, frenos, salida de aire a presión), notando una duración para cada muestra que refleje el inicio y final del evento correspondiente.
- Desarrollar un algoritmo para la detección de eventos sonoros distintos al tráfico rodado haciendo uso de una red neuronal artificial.
 - Obtención de los Coeficientes Cepstrales en las Frecuencias de Mel, por medio de la librería Librosa, de Python.
 - Separación del 70% de las muestras, con las cuales se realizará el proceso de entrenamiento (para un total mínimo de 1680).
 - Desarrollo, en Python, de un algoritmo basado en TensorFlow y Keras cuyos datos de entrada sean las muestras ya determinadas, clasificadas en las tres categorías ya determinadas. Se incluye, como parte de la información propia de cada muestra, el nivel promedio (en dBFS) de cada una.
 - Ejecución del algoritmo en la plataforma Colab de Google, usando las 1680 muestras separadas para el entrenamiento de la red neuronal.
 - Validación de la red neuronal por medio de las 720 muestras separadas para este propósito anteriormente.

- Evaluar la precisión del algoritmo en la clasificación de Eventos Sonoros No Deseados distintos al tráfico rodado mediante el uso de una matriz de confusión.
 - Toma de nuevas muestras, en un punto diferente de la ciudad, con el fin de evaluar no sólo la capacidad del algoritmo para identificar nuevos sonidos, sino hacer esto en condiciones ambientales diferentes a aquellas con las cuales fue entrenado el programa.
 - Separación de muestras - 100 de cada categoría.
 - Prueba de las muestras seleccionadas en el algoritmo, haciendo cuenta de los resultados acertados y fallidos (Falsos Negativos y Falsos Positivos).
 - Ingresando los valores de aciertos, Falsos Negativos y Falsos Positivoss obtenidos en una matriz de confusión multicategoría, hacer un análisis de precisión y exactitud, encontrando así un valor numérico mediante el cual expresar la efectividad del algoritmo desarrollado para identificar los eventos sonoros de estudio.

3.2. Determinación de Variables

- Dependientes: Métricas de evaluación de redes neuronales (Precisión y Exactitud).
- Independientes: Duración del evento sonoro, nivel promedio del evento, Coeficientes Cepstrales en las Frecuencias de Mel, flujo vehicular, sistema de registro de datos (grabadora, micrófono, preamplificador, frecuencia de muestreo, profundidad en bits).

3.3. Determinación de Variables y Elementos

Como se puede evidenciar en el apartado anterior, se cuenta con gran cantidad de Variables Independientes, lo cual puede explicarse con la naturaleza relativamente aleatoria de la información que este trabajo abarca, así como datos de entrada que será llevada la red neuronal. La obtención de algunas variables de independientes requerirá de un procesamiento y análisis de grabaciones de audio tomadas del mundo exterior. Las variables dependientes presentadas son obtenidas después de la implementación de la red convolucional, por lo cual son mejor explicadas al final.

En este orden de ideas, se empieza por la duración del evento sonoro, la cual es una propiedad intrínseca de los sonidos estudiados, ya que es dada por la longitud observada del evento; la cual puede ser obtenida fácilmente por medio de cualquier software de edición de audio, calculando el tiempo en segundos entre el inicio y el final del evento sonoro estudiado. El nivel promedio, en este caso, ha de medirse en dBFS, ya que para las capturas se hará uso de una grabadora de campo estéreo cuyos micrófonos no tienen un diámetro o disposición estándar que permita una calibración por medio de un pistófono; y el nivel en dBFS puede obtenerse en la pista de audio directamente,

y es independiente de más variables, pues es una indicación de la amplitud de un sonido real. Dicha pista se establece con una frecuencia de muestreo de 44.1 kHz y una profundidad de bits de 16 bit.

Los Coeficientes Cepstrales en las Frecuencias de Mel (MFCCs) hacen parte del procesamiento del audio, y permiten extraer características frecuenciales del sonido de una manera diseñada para ser similar a aquella en que funciona el oído humano. Estos permiten definir el audio por sus características principales en una imagen que contiene mucha menos información que el archivo de audio completo. A pesar de ser, a su vez, dependientes de la señal de audio de la cual son generados, para el propósito del presente trabajo son variables independientes, pues es de allí de donde el algoritmo extrae las características de cada archivo de audio, entrenándose y finalmente, generando valores para las métricas de precisión y exactitud.

Para el proceso de captura se depende de la favorabilidad del clima y las condiciones de tráfico, ya que la lluvia puede no sólo afectar los equipos de grabación, sino también crear ruido indeseado. Por esto, la presencia de lluvia es muy importante, y a pesar de que el método de determinarla es sencillo, haciéndose visualmente, es importante tener en cuenta la posibilidad de lluvia a juzgar por las nubes visibles y el pronóstico del clima, lo cual puede causar un aplazamiento en la fecha de captura. Respecto al tráfico, es importante tener en cuenta la cantidad de vehículos en el punto de medición, y la cantidad de eventos que pueden producirse en cada caso. Si el tráfico es muy lento, puede presentarse mayor cantidad de claxon, por ejemplo, más pueden enmascarse fácilmente inutilizando las muestras. Para asegurar un buen flujo vehicular, se hará un conteo, promediando el número de vehículos que pasen en dada unidad de tiempo.

Una vez hecho el procesamiento de datos y desarrollado (y validado) el algoritmo, se introducirán los números de Verdaderos Positivos, Falsos Positivos, Verdaderos Negativos y Falsos Negativos en una matriz de confusión multiclase, la cual permite obtener cantidad de parámetros

referentes al desempeño del algoritmo desarrollado. Se escogen, de estos, la precisión y la exactitud, las cuales se calculan a través de las siguientes fórmulas:

$$\text{Precisión (Precision)} = \frac{TP}{TP+FP} \quad (11)$$

$$\text{Exactitud (Accuracy)} = \frac{TP+TN}{P+N} \quad (12)$$

Donde TP corresponde a los Verdaderos Positivos, FP a los Falsos Positivos, TN a los Verdaderos Negativos, P a los Positivos Totales y N a los Negativos Totales.

Capítulo 4. Desarrollo de Ingeniería

En el siguiente trabajo se abordan campos de la ingeniería de sonido como lo son la detección y clasificación de eventos sonoros por medio de redes neuronales, esto conlleva a realizar grabaciones de campo, mediciones de niveles de ruido, procesamiento de señales, uso de estaciones de audio para la separación de muestras de audio que va a procesar el algoritmo, utilización de herramientas informáticas para la realización de algoritmos basados en detección y clasificación.

A continuación, se explicará en detalle las actividades realizadas para el desarrollo de cada uno de los objetivos específicos, así como los recursos utilizados y los criterios que se tuvieron en cuenta.

4.1. Equipos Utilizados

Los equipos utilizados en la toma de muestra se describen a continuación:

- **Capturadora de audio Zoom H1n**

La Zoom H1n Es una capturadora de mano de audio con sistema de grabación estéreo por medio de la configuración XY a 90° de sus micrófonos, los cuales son de condensador. La capturadora está alimentada con pilas y ofrece hasta 10 horas de tiempo de grabación, graba directamente a tarjetas microSD y maneja hasta los 120 dB SPL para una grabación sin distorsión. Fue escogida por su bajo costo y buenas reseñas, teniendo en cuenta que la etapa de grabación de sonidos no sería calibrada, y los micrófonos integrados cumplirían la función necesaria adecuadamente.



Figura 4. Zoom H1n. Fuente: (Zoom Corp, n.d.)

- **Trípode**

Trípode metálico de alta resistencia.



Figura 5. Trípode metálico. Fuente: Dixten

- **DAW Reaper**

REAPER es una aplicación completa de producción de audio digital para computadoras, que ofrece un conjunto completo de herramientas de grabación, edición, procesamiento, mezcla y

masterización de audio multipista y MIDI.¹ Se escoge dicho software por su gratuidad, además de tener experiencia en su manejo, facilitando el desarrollo del presente trabajo.



Figura 5. Software Reaper. Fuente: (Reaper, n.d.)

4.2. Captura de audio e identificación de clases

Para hacer la identificación de las clases de los eventos sonoros se hicieron previas grabaciones de video y audio en dos puntos a cercanos a la intersección entre la Avenida Ciudad de Cali con la Calle 68, utilizando un capturadora de audio Zoom H1n, cámara GoPro. Estas grabaciones se analizaron y se segmentaron para la debida extracción de muestras, luego se realizó un conteo para la clasificación de aquellas muestras, dando como resultado la detección de tres grandes clases como lo son “frenos”, “claxon” y “aire”. En la clase “frenos” se encuentran todos los sonidos asociados al “chillido” o sonido agudo en el momento del frenado del vehículo, el cual en materia automotriz se asocia al desgaste de las pastillas de los frenos en las ruedas de los carros y motos. En la clase “claxon” se tienen todos los sonidos del claxon, el pito o las bocinas de los vehículos. Por último, la clase llamada “aire” comprende las muestras de aquellos sonidos que su fuente principal son las salidas de aire a presión en los vehículos como la turbo transmisión, los frenos de

¹ <https://www.reaper.fm/>

ahogo y los multiplicadores de la transmisión en las tractomulas. En esta última clase hace mayor presencia los vehículos de carga pesada.



Figura 6. Punto de vista desde el punto de grabación. Fuente propia.

Una vez identificados los sonidos a analizar, se hace la etapa de grabación correspondiente, haciendo uso de una grabadora de mano Zoom H1n, montada sobre un trípode. La mayor parte de las muestras posteriores fueron tomadas sobre la Avenida Calle 127, en diferentes puntos a lo largo de su extensión al occidente de la Autopista Norte. El motivo para esto es facilidad de desplazamiento a dicho punto para la grabación, dado el número de muestras y, por consiguiente, grabaciones, por hacer. Se hacen capturas de aproximadamente 15 minutos a la vez, de acuerdo al Artículo 5 de la Resolución 0267 de 2006, del Ministerio de Ambiente. Los archivos son guardados en formato PCM, por medio de la extensión .wav, con una frecuencia de muestreo de 44100 Hz y una profundidad de 16 bits.

4.3. Separación de muestras

Para la creación de los dataset, se hizo una escucha y visualización de todas las grabaciones hechas, haciendo uso de herramientas de visualización de espectrograma integradas a los softwares

Reaper y FL Studio 20. Dicha herramienta permite visualizar, en el eje horizontal, el tiempo de reproducción, y en el vertical, las frecuencias que componen cada instante del sonido, obtenidas por medio de una Transformada Rápida de Fourier (FFT). Las imágenes presentadas a continuación muestran ejemplos de muestras ya separadas de Eventos Sonoros de freno, aire, y claxon respectivamente, en el visualizador de espectro de FL Studio 20.



Figura 7. Espectrograma de tres muestras de ruido de chillido de frenos. Fuente propia.

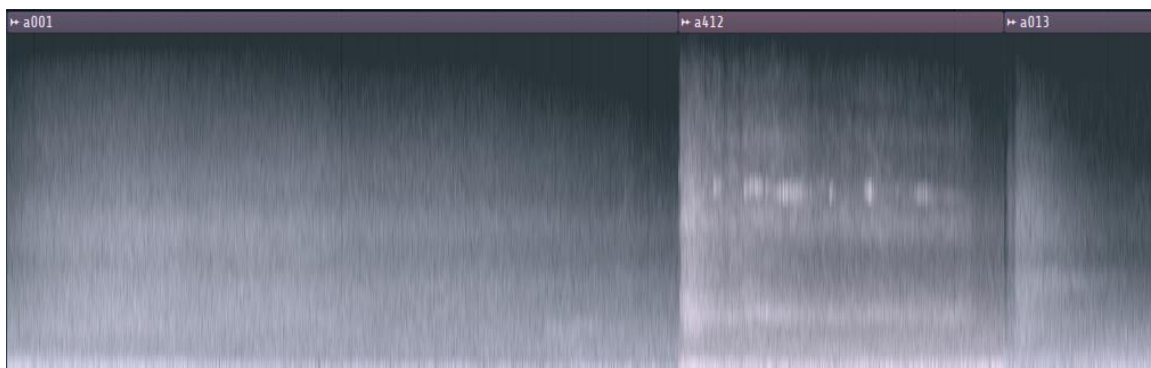


Figura 8. Espectrograma de tres muestras de ruido por salida de aire a presión. Fuente propia.

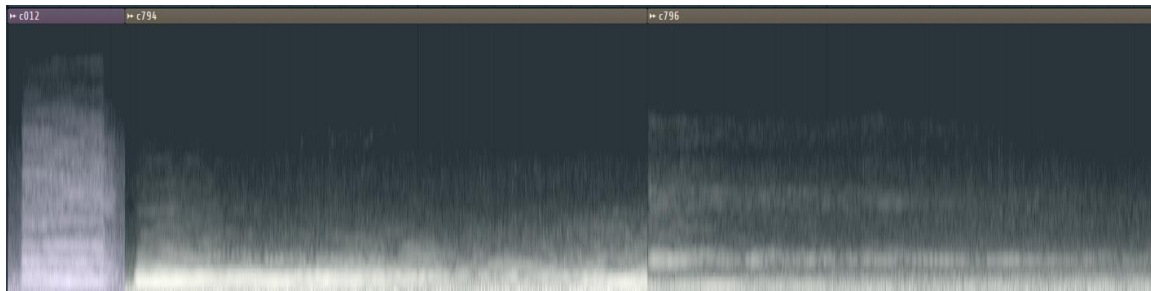


Figura 9. Espectrograma de tres muestras de ruido de claxon. Fuente propia.

Es posible, observando las anteriores imágenes, identificar las características de cada tipo de evento, incluyendo su longitud, nivel sonoro y contenido en frecuencia; sin embargo, un análisis a profundidad sobre dichas características puede encontrarse en la siguiente sección.

Al recortar cada una de las muestras dentro del software utilizado, se exportan a carpetas debidamente etiquetadas para cada uno de los eventos sonoros propuestos. Dentro de estas, se enumeran las muestras con una letra precedente indicativa del tipo de evento sonoro pertinente, con la letra *f* indicando que la muestra pertenece a la familia de ruido de freno, la letra *c* indicando que hace parte de los sonidos de claxon, la letra *a* indicando su pertenencia a salidas de aire a presión, y la letra *s* indicando que es un sonido sin clasificar.

Debido al modo en que opera el algoritmo, es importante dividir las muestras en un dataset de entrenamiento y uno de validación. Se ha definido ya que el porcentaje de las muestras dedicada a cada dataset será de 70% y 30%, respectivamente, con un 70% de las muestras dedicadas al entrenamiento, y el 30% a la validación. Dichos porcentajes se escogen por práctica común, ya que a pesar de no existir fórmula alguna para decidir la división adecuada, la regla 70/30 es de uso común (Stone, 1974). Dado que cada categoría contempla 800 muestras, se divide cada grupo de muestras en estos porcentajes, obteniendo 560 muestras de cada categoría para entrenamiento, y 240 para validación. En total, considerando las 4 categorías a trabajar, se tienen 2240 muestras para

entrenamiento, y 960 para validación. Dicho esto, se copian las muestras correspondientes a dos carpetas llamadas “train” y “test”, con las cuales se alimenta el algoritmo explicado más adelante.

Finalmente, y para correcto funcionamiento del algoritmo, es crucial crear un documento complementario de las muestras en formato .csv (Comma Separated Values), el cual dirá a la red neuronal el nombre de las muestras que analizará, y las categorías correspondientes a cada una. Así, se escriben dos archivos .csv (uno para las muestras de entrenamiento, train.csv, y otro para las muestras de validación, test.csv) con dos columnas, cuyo título corresponde al nombre y categoría de cada muestra, denotado por fname y label. Por ejemplo, la muestra de freno número 529 aparecerá entrada como f529.wav,freno.

4.4. Análisis de características

Una vez obtenidas 800 muestras por cada categoría, cantidad explicada en el apartado 3.1 Recopilación de Datos, se hace un análisis de tres propiedades de las diferentes categorías de Evento Sonoro, con el fin de analizar las características que diferencia a cada una.

Como es expuesto en el primer Objetivo Específico, aquellas propiedades se definen como duración, nivel promedio y espectro en frecuencia; medidas a través de longitud, nivel RMS y llanura espectral, respectivamente.

4.4.1. Análisis de duración

Tomando la longitud, en segundos, de las 800 muestras correspondientes a cada categoría, se hace un análisis estadístico comparando los valores característicos para cada categoría de Evento Sonoro, dando indicación de las propiedades temporales de los mismos.

Al obtener la media de longitud para cada categoría se obtienen valores de 0.55 segundos para el ruido por salida de aire a presión, 0.88 segundos para el ruido por chillido de frenos y 0.66

segundos para ruido de claxon. Un diagrama de caja y bigotes² indicando los cuartiles y rango de la duración de cada categoría puede apreciarse a continuación.

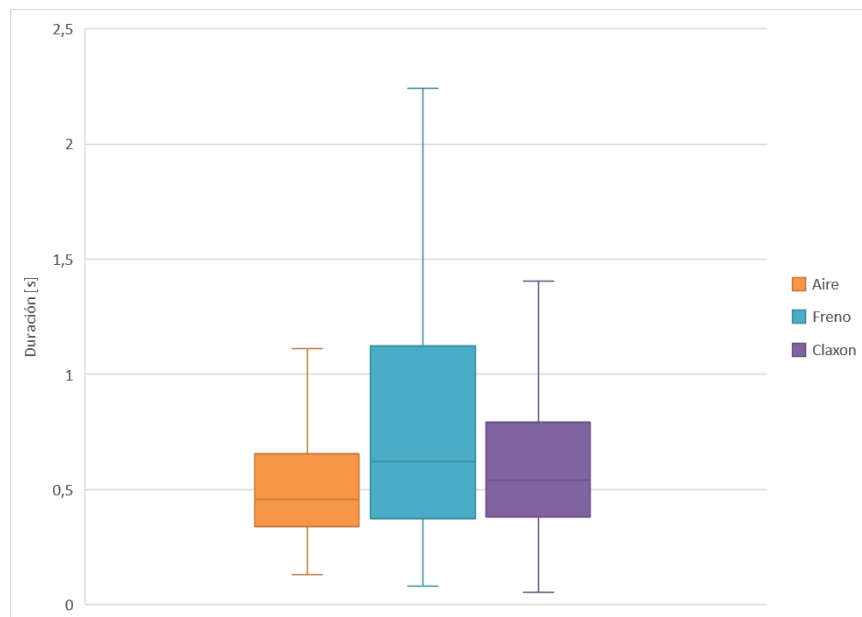


Figura 10. Diagrama de caja y bigotes de distribución de longitudes para cada categoría. Fuente propia.

El ruido por salida de aire a presión presenta un Cuartil 1 en 0.34 segundos, un Cuartil 3 en 0.65 segundos, con un rango intercuartílico de 0.31 segundos, y un rango de 0.98 segundos.

El ruido por chillido de frenos presenta un Q1 en 0.38 segundos, un Q3 en 1.12 segundos, con un rango intercuartílico de 0.75 segundos, y un rango de 2.16 segundos.

El sonido de claxon presenta un Q1 en 0.38 segundos, un Q3 en 0.79 segundos, con un rango intercuartílico de 0.41 segundos, y un rango de 1.35 segundos.

² Con el fin de hacer un análisis enfocado a las características de los sonidos, los valores atípicos son excluidos, pues son comparativamente pocos respecto a los datos presentes dentro de los cuartiles.

Haciendo análisis de los promedios presentados anteriormente y el diagrama de caja y bigotes, se evidencia que el ruido de freno no sólo tiende a ser el de mayor longitud, sino también el de mayor rango, seguido por el sonido de claxon y la salida de aire a presión, lo cual es consistente con la naturaleza de cada tipo de sonido. Por ejemplo, el ruido por salida de aire a presión es un sonido de corta duración por naturaleza, por lo cual tiene poca tendencia a perdurar en el tiempo y, al ser la velocidad a la cual escapa el aire una función de la construcción física de la válvula mediante la cual escapa, su duración tiene poca variación. El 75% inferior de las muestras tienen una longitud entre los 0.13 y los 0.65 segundos, el 25% superior están entre los 0.65 y los 1.11 segundos.

El ruido por chillido de frenos, por el contrario, muestra no sólo una longitud promedio más larga, sino también un rango muy extenso. El tiempo de frenado de un vehículo depende de factores como su velocidad inicial, la velocidad final y el nivel de aplicación de los frenos, los cuales crean variación en el tiempo en el cual el chillido característico se puede evidenciar. No es un sonido impulsivo por naturaleza, por lo cual su longitud puede prolongarse bastante, con el 25% superior encontrándose entre 1.12 y 2.24 segundos.

Finalmente, el sonido de claxon demuestra encontrarse intermedio entre las otras dos categorías, debido a que, aunque es un sonido impulsivo, no tiene una longitud definida por las dimensiones de la fuente, y su longitud es dependiente del uso que el conductor del vehículo le dé. El 75% inferior de los datos corresponden a sonidos entre 0.05 y 0.79 segundos, posiblemente de usos preventivos del claxon, con el 25% superior ubicado entre 0.79 y 1.4 segundos.

4.4.2. Análisis de nivel promedio

Se hace un cálculo del valor dBFS (RMS) de cada uno de los archivos correspondientes a las tres categorías, de acuerdo al cálculo explicado en la sección 4.6.1. Con los resultados, se prosigue con un análisis estadístico comparando los valores característicos para cada categoría de Evento

Sonoro, indicando cómo se comparan unos con otros en lo que respecta a nivel sonoro. Las muestras no fueron calibradas, más al tener todas ellas la misma profundidad de bits (16 bit), su nivel digital medido en dBFS puede ser comparado. La grabadora Zoom H1n, mediante la cual fueron hechas las tomas, se mantuvo en un nivel de preamplificación 6 (en una escala 0-10), y la función “Autolevel” apagada, con el fin de lograr consistencia de nivel sonoro a través de todas las muestras.

Al obtener el nivel RMS promedio para cada categoría se obtienen valores de -30.23 dBFS para el ruido por salida de aire a presión, -33.17 dBFS para el ruido por chillido de frenos y -30.49 dBFS para ruido de claxon. Un diagrama de caja y bigotes indicando los cuartiles y rango del nivel sonoro de cada categoría puede apreciarse a continuación.

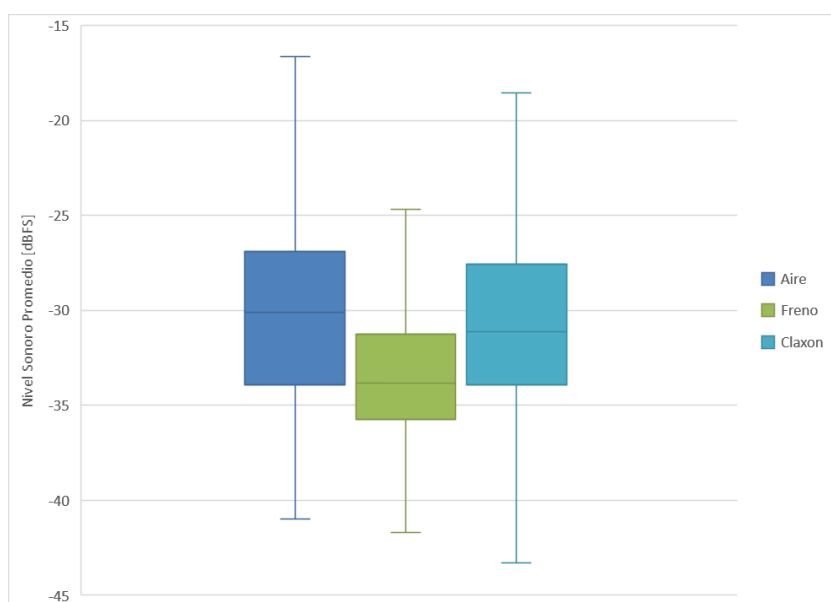


Figura 11. Diagrama de caja y bigotes de distribución de Nivel Sonoro (RMS) para cada categoría. Fuente propia.

El ruido por salida de aire a presión presenta un Cuartil 1 en -33.92 dBFS, un Cuartil 3 en -26.89 dBFS, con un rango intercuartílico de 7.03 dB, y un rango de 24.32 dB.

El ruido por chillido de frenos presenta un Q1 en -35.74 dBFS, un Q3 en -31.26 dBFS, con un rango intercuartílico de 4.49 dB, y un rango de 17.03 dB.

El sonido de claxon presenta un Q1 en -33.94 dBFS, un Q3 en -27.56 dBFS, con un rango intercuartílico de 6.38 dB, y un rango de 24.73 dB.

Haciendo análisis de los promedios presentados anteriormente y el diagrama de caja y bigotes, se evidencia que la salida de aire a presión presenta el mayor nivel promedio, seguido del claxon y el chillido de freno, lo cual es consistente con la naturaleza de cada tipo de sonido. Por ejemplo, el ruido por salida de aire a presión es un sonido que, si bien puede mezclarse con el fondo (como se explorará en el siguiente segmento), es sonoro y definido cuando se produce en cercanía, por lo cual presenta los mayores niveles entre las muestras presentadas. El 75% inferior de las muestras tienen una longitud entre -40.97 y -26.89 dBFS, el 25% superior están entre -40.97 y -16.65 dBFS.

El ruido por chillido de frenos es el más tenue, al tratarse de un ruido por fricción leve entre dos superficies metálicas, y sin características impulsivas. El 75% inferior de las muestras se ubica entre -41.72 y -31.26 dBFS, con el 25% superior entre -34.26 y -24.59 dBFS.

Finalmente, el sonido de claxon se ubica en la mitad de sonoridad entre los demás tipos de muestra. A pesar de ser, por diseño, sonoro; no supera en promedio la sonoridad del sonido por salida de aire a presión. Una razón para esto es que los cláxones son más notorios en comparación con el ruido de fondo, lo cual permite que aquellos registrados y clasificados puedan ser en mayor parte, más lejanos, sin tener que estar en campo directo de la fuente para ser registrado y distinguido. Esto demuestra también el amplio rango presente, mayor a las otras dos categorías. El 75% inferior de los datos se encuentran entre -43.29 y -27.56 dBFS, mientras el 25% superior se encuentra entre -27.56 y -18.56 dBFS.

4.4.3. Análisis de espectro en frecuencia

Como es explicado en el Marco Conceptual, la llanura espectral es una medida mediante la cual se puede determinar la atonalidad de un sonido, determinando qué tan “plana” es una gráfica de la Transformada de Fourier del mismo. El rango de valores de la llanura espectral es de 0 a 1, variando entre diferentes magnitudes, por lo cual se acostumbra la conversión a dBFS, para más fácil lectura de los resultados; en la siguiente gráfica, un valor más inferior representa un sonido más tonal, y uno superior representa un sonido más atonal, similar al ruido aleatorio. Hecha dicha conversión, se prosigue con un análisis estadístico comparando los valores característicos para cada categoría de Evento Sonoro, permitiendo conocer las características espectrales de cada tipo de sonido. El ruido por salida de aire a presión obtuvo un promedio de -54.45 dBFS, el ruido de freno -61.12 dBFS y el sonido de claxon -62.66 dBFS.

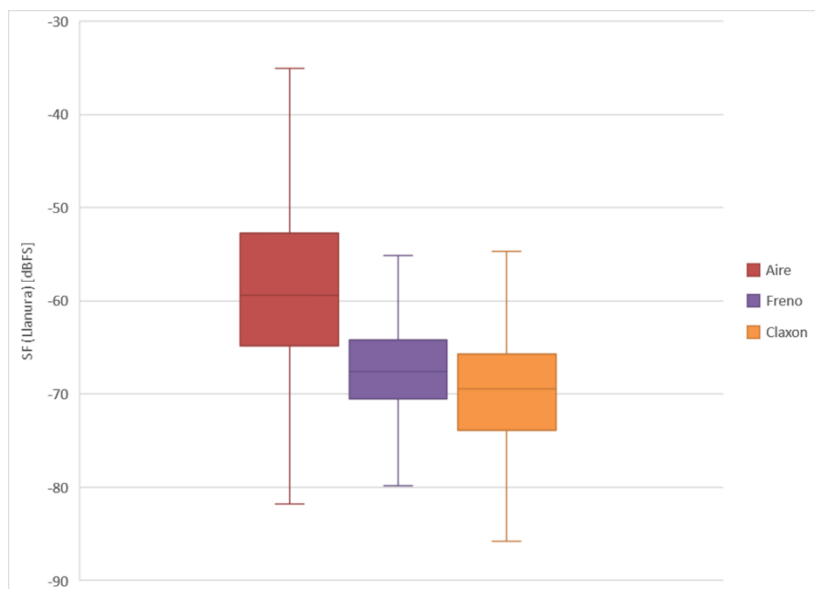


Figura 12. Diagrama de caja y bigotes de distribución de llanura espectral para cada categoría. Fuente propia.

El ruido por salida de aire a presión presenta un Cuartil 1 en -59.97 dBFS, un Cuartil 3 en -49.93 dBFS, con un rango intercuartílico de 10.04 dB, y un rango de 39.52 dB.

El ruido por chillido de frenos presenta un Q1 en -64.12 dBFS, un Q3 en -58.17 dBFS, con un rango intercuartílico de 5.95 dB, y un rango de 23.20 dB.

El sonido de claxon presenta un Q1 en -66.69 dBFS, un Q3 en -59.59 dBFS, con un rango intercuartílico de 7.10 dB, y un rango de 27.36 dB.

El comportamiento observado de los resultados obtenidos refleja el comportamiento visto en las imágenes del apartado 4.3, mostrando una clara distinción en contenido en frecuencia entre los tres tipos de eventos sonoros. Es en esta métrica donde se evidencia la mayor diferencia numérica entre las categorías, indicando que el principal factor diferenciador es el contenido en frecuencia de cada una de las categorías.

Es la categoría de ruido por salida de aire a presión la cual muestra la mayor diferenciación, con un promedio considerablemente superior a las otras dos, indicando que, en efecto, al ser un ruido de banda ancha, genera un espectrograma bastante plano, matizado por su decaimiento en alta frecuencia, que explica también los bajos valores de planitud encontrados en el cuartil inferior del grupo de datos. El 75% superior se encuentra en valores entre -59.97 y -35.22 dBFS, con el 25% inferior encontrándose entre -74.74 y -59.97 dBFS.

El ruido por chillido de frenos se presenta mucho más tonal, como es de esperarse dada su fuente siendo la fricción entre dos piezas metálicas, produciendo un tono. Cabe resaltar que a pesar de ser más tonal que el sonido de claxon, encuentra en general valores de llanura mayores; esto puede explicarse haciendo uso de los resultados encontrados en el apartado anterior, notando su nivel sonoro mucho menor, causando que el aporte del ruido de fondo (i.e., aquel independiente de la presencia del ruido de freno) sea mayor, balanceando el espectrograma resultante y aumentando la llanura del mismo. El 75% superior de los datos de llanura se encuentran entre -64.12 y -49.69 dBFS, mientras que el 25% inferior se encuentra entre -72.89 y -64.12 dBFS.

El sonido de claxon es, por diseño, tonal, ya que genera uno o más tonos por medio de una fuente destinada específicamente para ello, con un timbre específico que permite resaltar frente a otro ruido presente. Es este timbre el que causa que la llanura no sea menor, pues a pesar de ser muy constante en su tonalidad, es un sonido que se expande en el espectro de frecuencia, más aún cuando el claxon está compuesto de múltiples tonos, como es común. De analizarse aislados, el ruido por chillido de frenos sería considerablemente menos llano que el sonido de claxon, no obstante, la realidad es otra, por motivos expuestos en el apartado anterior. El 75% superior de los datos de llanura espectral para el ruido por claxon se encuentran en el rango entre -66.69 y -49.84 dBFS, con el 25% inferior ubicando entre -77.20 y -66.69 dBFS.

Dicho análisis permitió hacer una comprobación y selección de los archivos en sus respectivas categorías, asegurándose que caigan dentro de los rangos establecidos. En este orden de ideas, un archivo que contenga un sonido de ruido por salida de aire a presión deberá tener una duración de entre 0.13 y 1.11 segundos, un nivel promedio entre -40.97 y -16.65 dBFS y una llanura espectral entre -74.74 y -35.22 dBFS; un archivo con el sonido del chillido de pastillas de freno una duración entre 0.08 y 2.24 segundos, un nivel promedio entre -41.72 y -24.59 dBFS, y una llanura espectral entre -72.89 y -49.69 dBFS. Finalmente, un archivo con una grabación del sonido de un claxon, deberá durar entre 0.05 y 1.4 segundos, con un nivel promedio entre -43.29 y -18.56 dBFS, y una llanura espectral entre -77.2 y -49.84 dBFS. Al probar los archivos por este filtro, se identifican datos no pertenecientes, los cuales son movidos a sus correctas categorías, y reemplazados.

4.5. Aleatorización de los datos

A la hora de entrenar un algoritmo, es importante asegurarse que no se formen grupos de datos similares consecutivos, ya que esto puede sesgar la clasificación realizada por la red neuronal hacia aquellos elementos repetidos. Para evitarlo, se decide aleatorizar los datos a utilizar, evitando en lo

más posible consecución de numerales de una misma clase, y buscando una distribución equitativa de probabilidad entre las cuatro clases, i.e., evitar aglomeraciones de varias muestras de la misma clase.

Para dicho proceso se hace uso de Microsoft Excel, por medio de la función =ALEATORIO(), la cual genera un valor decimal aleatorio entre 0 y 1 cada vez que se actualiza la celda. El proceso sigue la siguiente estructura:

1. Se genera la lista de datos correspondiente al nombre de los archivos utilizados para cada categoría, empezando con a001, hasta a800. Se organizan verticalmente en la columna A de Excel.
2. Se copian y pegan las 800 celdas a Microsoft Word, donde por medio de la función “Reemplazar” se cambia el salto de línea (representado por el símbolo ^p), por la extensión del archivo y su clase. Es decir, en el caso del valor a549, después del reemplazo se registra como a549.wav,aire. Esto para cumplir con los requerimientos de fname y label descritos en la sección anterior. Una vez hecho esto, se trasladan los nuevos datos a Excel.
3. Se escribe la fórmula =ALEATORIO() en la celda B1 (directamente al frente del primer dato, en este caso, a001).
4. Se extiende la fórmula hasta coincidir con el número de datos directamente a la izquierda, es decir, hasta la fila 800. Esto genera 800 datos aleatorios, correspondiente cada uno a un nombre de archivo.
5. Se selecciona de nuevo la celda B1, y bajo la pestaña Datos, sección Ordenar y Filtrar, se selecciona la herramienta para organizar de mayor a menor los datos (no es

importante cuál). Esto organiza los datos aleatorios generados consecutivamente, y así mismo los nombres de archivo correspondiente a cada uno, pues esta herramienta detecta datos adyacentes como parte de una tabla.

6. El listado resultante en la columna A está ya aleatorizado, pero dado que la función `=ALEATORIO()` genera un nuevo valor cada vez que la celda es actualizada, el proceso de organización genera datos completamente nuevos, por lo cual se puede repetir el paso 5 múltiples veces.
7. Se repiten los pasos 1 a 6 para cada una de las diferentes categorías (cada una en una hoja de Excel diferente). Al trabajar cada categoría por aparte, se asegura que habrá una selección proporcional de cada tipo de archivo.
8. Una vez hecho este proceso con las cuatro categorías, se escogen las primeras 560 celdas de cada categoría, las cuales la están aleatorizadas, y se insertan en una nueva hoja de Excel una después de la otra, para un total de 2240 entradas. Se repiten los pasos 3 y 4, extendiendo la fórmula hasta cubrir todas las muestras de las cuatro categorías.
9. Se repite el paso 7 escogiendo las últimas 240 muestras de cada categoría, es decir, aquellas que no fueron seleccionadas en el paso anterior. De igual manera, se ponen en una hoja aparte completando 960 celdas. Igualmente se repiten los pasos 3 y 4 en esta selección.
10. Los resultados de los pasos 8 y 9 corresponden a los datos que se utilizarán para `train.csv` y `test.csv`, respectivamente. Para esto, se copia la lista de cada archivo y se pega en el Bloc de Notas de Windows, poniendo antes de la primera línea `"fname,label"`, para denominar el cabecero del archivo. Los respectivos archivos son guardados como `"train.csv"` y `"test.csv"`.

4.6. Obtención fórmula para cálculo de dBSPL

Con el fin de demostrar los efectos de los eventos sonoros no deseados en el nivel de presión sonora de una grabación de tráfico rodado, es importante determinar un método para obtener dichos valores a partir de los archivos existentes. Para esto, se aprovecha la función *splmeter* de MatLab, la cual toma un archivo de calibración y utiliza este valor para determinar el nivel de presión sonora de otro archivo de audio que se introduzca. Es importante notar que, dado que el presente proyecto fue desarrollado en el lenguaje de programación Python, esta función no puede ser implementada directamente, teniendo que recurrir a encontrar la fórmula mediante la cual MatLab consigue dicho proceso.

Para esto, se generan 12 archivos de audio por medio del Software libre Audacity, el cual expresa la amplitud de una señal como un valor de 0 a 1. Los archivos generados contienen cinco segundos de una onda sinusoidal a 1000 Hz, con amplitudes de 0.01, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9 y 1. El propósito de esto es generar archivos controlados para calcular la fórmula base de la función *splmeter* a partir de los resultados obtenidos. A continuación, se describirá como, probando diferentes archivos como archivos de calibración, y comparando los otros once se puede extrapolar una función a través de Microsoft Excel.

Se desarrolla pues un sencillo código de MatLab, estableciendo un archivo como de calibración y extrayendo el nivel correspondiente de los demás archivos. Se muestra aquí una sección del código, evaluando la señal de amplitud 0.01 como señal de calibración (definida como 94 dBSPL):

```

clc

[testTone,FS] = audioread('punto01.wav');

splMeterReading = 94;
calib = calibrateMicrophone(testTone,FS,splMeterReading,"FrequencyWeighting","C-weighting");

[x,FS] = audioread('punto01.wav');
max(x);

spl = splMeter("CalibrationFactor",calib, ...
    "FrequencyWeighting","A-weighting", ...
    "TimeWeighting","Fast", ...
    "TimeInterval",0.2, ...
    "SampleRate",FS);

[LCF,~,LCpeak] = spl(x);
maximo(1)=max(LCF);

```

Este proceso se repite para los 12 archivos a analizar y se tabulan los resultados. Para el caso del ejemplo, en donde el archivo de calibración corresponde a aquel de amplitud 0.01, se obtienen los siguientes datos:

Tabla 1. Valores de SPL y Pascales con valor de calibración 0,01. Fuente propia.

Amplitud	dB SPL	Pa
0,01	94	1,0
0,05	108	5,0
0,1	114	10,0
0,2	120	20,0
0,3	123,5	29,9
0,4	126	39,9
0,5	128	50,2
0,6	129,6	60,4
0,7	130,9	70,2
0,8	132,1	80,5
0,9	133,1	90,4
1	134	100,2

Al graficar los datos obtenidos, y encontrado una línea de tendencia que se ajustase a los datos dados, se obtiene la siguiente gráfica:

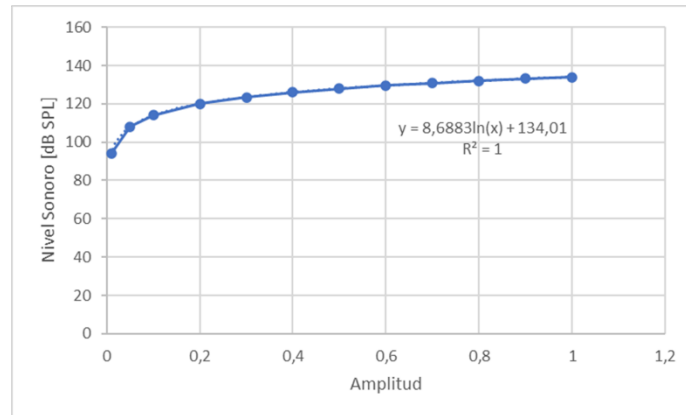


Figura 13. Gráfica cálculo de MatLab con valor de calibración 0,01. Fuente propia.

Así mismo, se obtiene la siguiente gráfica para la expresión en Pascales:

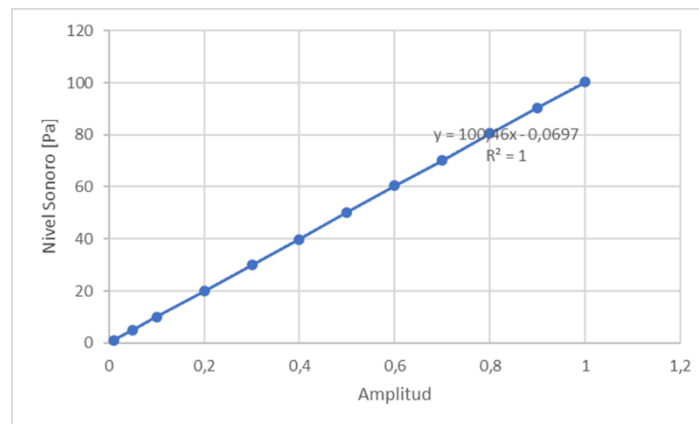


Figura 14. Gráfica cálculo de MatLab con valor de calibración 0.01 (lineal). Fuente propia.

La fórmula encontrada para dBSPL es la siguiente:

$$dBSPL = 8.7 * \ln(Amplitud) + 134 \text{ [dBSPL]} \quad (12)$$

La fórmula encontrada para Pascales es la siguiente:

$$Pa = 100.2 * Amplitud - 0.02 \text{ [Pa]} \quad (13)$$

Para el caso de calibración ajustada en amplitud 0.05, se obtienen los siguientes valores:

Tabla 2. Valores de SPL y Pascales con valor de calibración 0,05. Fuente propia.

Amplitud	dB SPL	Pa
0,01	80	0,2
0,05	94	1,0
0,1	100	2,0
0,2	106	4,0
0,3	109,6	6,0
0,4	112,1	8,1
0,5	114	10,0
0,6	115,6	12,1
0,7	116,9	14,0
0,8	118,1	16,1
0,9	119,1	18,0
1	120	20,0

Los cuales, al ser graficados, se obtienen de la siguiente manera:

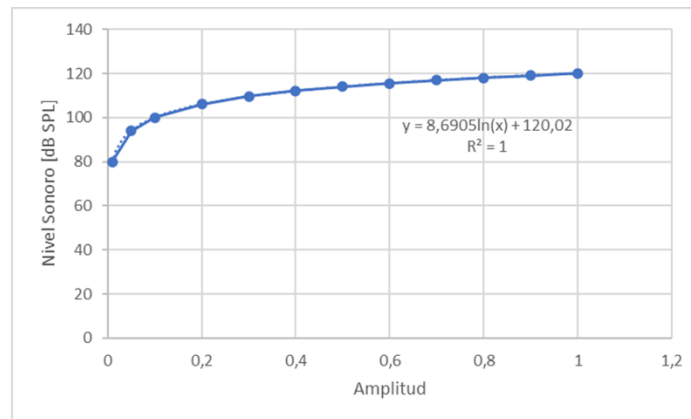


Figura 15. Gráfica cálculo de MatLab con valor de calibración 0,05. Fuente propia.

Así mismo, se obtiene la siguiente gráfica para la expresión en Pascales:

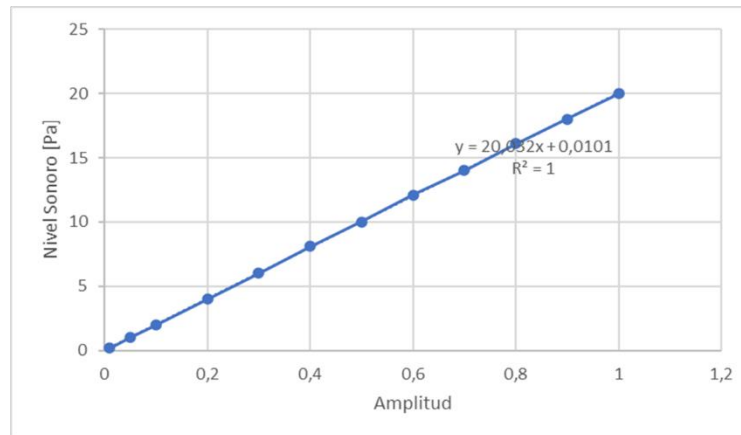


Figura 16. Gráfica cálculo de MatLab con valor de calibración 0.05 (lineal). Fuente propia.

La fórmula encontrada en dBSPL es:

$$dBSPL = 8.7 * \ln(Amplitud) + 120 [dBSPL] \quad (14)$$

La fórmula encontrada en Pascales es:

$$Pa = 20 * Amplitud + 0.006 [Pa] \quad (15)$$

Al desarrollar este procedimiento, estableciendo cada una de las 12 muestras como valor de calibración, se encuentra un patrón, en el cual la fórmula obtenida corresponde siempre a un logaritmo natural, con coeficiente 8.7, y con un complemento inversamente proporcional a la amplitud de la señal de calibración. Se toma nota de dichos complementos, y se obtiene la siguiente tabla:

Tabla 3. Complemento de ecuación respecto a valor de calibración. Fuente propia.

Calibración	Complemento
1	94
0,9	95
0,8	96
0,7	97
0,6	98,4
0,5	100
0,4	102
0,3	104,5
0,2	108
0,1	114
0,05	120
0,01	134

Estos valores son también graficados, con el fin de encontrar una función que defina el comportamiento de estos complementos, hallando pues la siguiente gráfica:

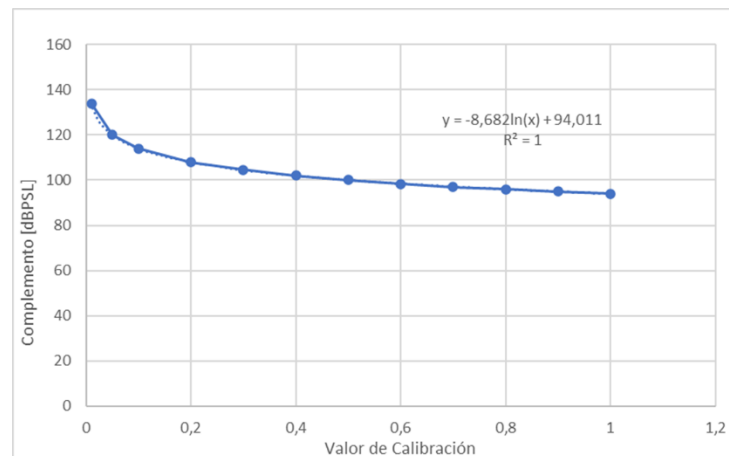


Figura 17. Gráfica cálculo de complemento fórmula. Fuente propia.

De igual manera, se observa que, al desarrollar las fórmulas correspondientes al cálculo en Pascales, el complemento varía entre -0,02 y 0,01, sin tener una clara tendencia a subida o bajada,

por lo cual se descarta como variación natural, y se observa el coeficiente asociado. Dicho coeficiente es tabulado mostrando el siguiente comportamiento:

Tabla 4. Coeficiente asociado a diferentes valores de calibración (Pascales). Fuente propia.

Calibración	Coeficiente
0,01	100,20
0,05	20,00
0,1	10,02
0,2	5,00
0,3	3,34
0,4	2,51
0,5	2,00
0,6	1,66
0,7	1,44
0,8	1,25
0,9	1,11
1	1,00

Teniendo los valores dados por la Tabla 4, se grafican con el fin de obtener una fórmula que describa el comportamiento de los coeficientes:

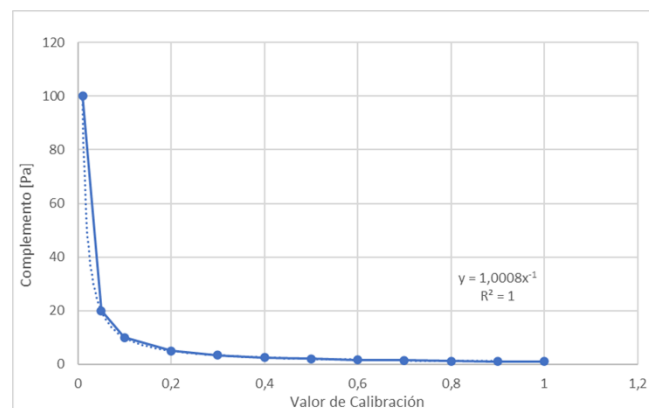


Figura 18. Gráfica de coeficientes (Pascales). Fuente propia.

La fórmula obtenida se puede expresar de la siguiente manera:

$$\text{Coeficiente} = \frac{1}{\text{Calibración}} \quad (16)$$

Añadiendo la fórmula para coeficientes recién obtenida a aquella obtenida para el cálculo de pascales, se obtiene:

$$Pascales = \frac{1}{Calibración} * Amplitud$$

$$Pascales = \frac{Amplitud}{Calibración} [dBSPL] \text{ (17)}$$

Con los valores obtenidos en la Tabla 3 y la Figura 17, asociadas al complemento de la fórmula para decibeles SPL, se obtiene pues una fórmula logarítmica negativa, descrita de la siguiente manera:

$$Complemento = -8.7 * \ln(Calibración) + 94 \text{ (18)}$$

Uniendo la fórmula encontrada anteriormente y la recién determinada, se obtiene la siguiente expresión:

$$dBSPL = 8.7 * \ln(Amplitud) + (-8.7 * \ln(Calibración) + 94) [dBSPL] \text{ (19)}$$

Simplificando dicha expresión, se obtiene pues una ecuación final mediante la cual puede calcularse el Nivel de Presión Sonora en dBSPL a partir de un archivo de audio y una señal de calibración:

$$dBSPL = 8.7 * \ln(Amplitud) - 8.7 * \ln(Calibración) + 94 [dBSPL] \text{ (20)}$$

Donde *Amplitud* corresponde al valor momentáneo de la señal de audio, en una amplitud de 0 a 1, y *Calibración* corresponde al valor máximo del archivo de calibración dado para la grabación.

Teniendo dos archivos de audio calibrados a 94 y 114 dBSPL, se procede a validar la fórmula encontrada, evaluando dichos archivos y esperando obtener los niveles correspondientes. Se nota aquí que dicho cálculo está alejado en 4 decibeles, pues los valores obtenidos corresponden a 90 y 110 dBSPL, respectivamente.

Este resultado hace necesaria una observación y análisis del material original, observando las condiciones en las cuales fueron tomadas las muestras. Se descubre que, al hacerse uso de un micrófono USB, la ganancia no se aplica a nivel físico, sino en software; revisando los pasos que se tomaron para la grabación original se descubre que se movió un slider digital de nivel puesto en -4 dB, causando que los archivos de audio, incluido aquellos de calibración, fueran exportados con un nivel 4 dB inferior al correcto. Por esto, para el Capítulo 5.2, en donde se analizará el desempeño de la fórmula aquí hallada, se hará una compensación por estos 4 dB, sin afectar el resultado de la fórmula hallada.

Simplificando la Ecuación 20, por propiedades de logaritmos:

$$dBSPL = 8.7 * \ln\left(\frac{Amplitud}{Calibración}\right) + 94 \text{ [dBSPL]} \quad (21)$$

$$dBSPL = 8.7 * \frac{\log_{10}(Amplitud/Calibración)}{\log_{10} e} + 94$$

$$dBSPL = 8.7 * \frac{\log_{10}(Amplitud/Calibración)}{0.434} + 94$$

$$dBSPL = 20 * \log\left(\frac{Amplitud}{Calibración}\right) + 94 \text{ [dBSPL]} \quad (22)$$

Por ejemplo, si quiere hallarse el valor SPL de una señal con amplitud 0.81, dado un valor de calibración establecido en 0.4, se obtiene el siguiente resultado:

$$20 * \log\left(\frac{0.81}{0.4}\right) + 94 = 100.13 \text{ dB SPL}$$

Es de notar de la fórmula presentada:

- Es dada sin ponderación de ningún tipo, pues la frecuencia o longitud de la señal no forman parte del cálculo. Se define entonces como un cálculo de dB SPL momentáneo ponderado Z, o lineal.
- Es independiente de la profundidad en bits de la señal de audio, pues en todo caso se normaliza a una amplitud de 0 a 1, sin importar el número de valores intermedios entre dichas cantidades.
- Tanto Python como MatLab trabajan amplitudes de -1 a 1, indicando el *zero crossing* en 0. Es de notar pues que, si se intenta calcular un valor de dB SPL por medio de la fórmula obtenida a partir de un valor negativo, habrá un error de matemáticas, pues no es posible calcular el logaritmo de una cantidad negativa.

Para solucionar este último punto presentado, se realiza primero un cálculo de Raíz Cuadrada Media (RMS) a los datos sin procesar, proceso descrito en el apartado siguiente.

4.6.1. Cálculo valor RMS

Para obtener un nivel equivalente (o L_{eq}) de cada segmento de audio, se procede a calcular el valor RMS (Valor Cuadrático Medio) de cada uno de estos, lo cual elimina además los posibles problemas causados al tratar de calcular un logaritmo de una cantidad negativa.

La ecuación utilizada para el cálculo del nivel RMS de una señal se define de la siguiente manera:

$$RMS = \sqrt{\frac{1}{T} \int_0^T x^2 dx} \quad (23)$$

Donde T representa el tiempo de integración, y x representa el valor de amplitud instantáneo³ en cada momento de integración, desde el instante 0 hasta el tiempo T .

Dado que en el dominio digital los datos son representados de forma discreta, y cada uno de los segmentos definidos se define como un vector, una definición más adecuada del valor RMS se contempla así:

$$RMS = \sqrt{\frac{1}{n} \sum_i x_i^2} \quad (24)$$

Donde n corresponde al número de valores dentro del arreglo, i representa un índice de iteraciones, y x el valor de amplitud correspondiente a cada posición dentro del arreglo definido.

³ Se toma dicho valor como un porcentaje, entre 0 y 1, siendo 1 el valor máximo digital (0 dBFS).

Una vez obtenido dicho valor, se aplica la fórmula encontrada en el apartado anterior para determinar la equivalencia del valor RMS en dB SPL.

4.7. Desarrollo del algoritmo

A continuación, se muestra el proceso para la elaboración del entrenamiento y procesamiento del algoritmo para la detección y clasificación de eventos sonoros no deseados presentes en el ruido por tráfico rodado a través de redes neuronales, el cual se realiza a través de Google Colab mediante la sintaxis de Python. El código aquí descrito fue adaptado ampliamente de aquel encontrado en la página de repositorios *GitHub*, como parte del proyecto “Comparative Analysis of Classifiers on Audio, Image, and Text Datasets”, de Akshay Singh Rana, Harmanpreet Singh, y Himanshu Arora, presentado ante la Universidad de Montréal en el año 2019.

El presente capítulo es una vista total del código implementado, para una explicación visual de la lógica del mismo, véase el Anexo II: Diagrama de Flujo del Algoritmo.

En primera medida se sube el contenido de las muestras obtenidas en el proceso de segmentación mediante la página de almacenamiento en línea Drive de Google en la cual van a ir todas las muestras tanto las muestras para el entrenamiento como las muestras para la validación.

En primera instancia se importa *gdrive*, la librería de Google Colab que permite que los servidores de Google tengan acceso al almacenamiento personal en el cual están almacenados los dataset, declarando además el camino base en el cual se podrán encontrar:

```

from google.colab import drive
drive.mount('/content/drive', force_remount=True)

# Montar datasets
import sys
IN_COLAB = "google.colab" in sys.modules
if IN_COLAB:
    base_path = '/content/drive/My Drive/Colab Notebooks/Tesis'
else:
    base_path = ""

# Definir paths de datasets
import os
train_path = os.path.join(base_path, 'train')
train_labels_path = os.path.join(base_path, 'csvs', 'train.csv')
test_path = os.path.join(base_path, 'test')
test_labels_path = os.path.join(base_path, 'csvs', 'test.csv')

Mounted at /content/drive

```

Figura 19. Creación de rutas de entrenamiento y validación. Fuente propia

Se importan todas las librerías a utilizar, asegurándose de instalar aquellas que no estén inmediatamente disponibles:

```

!pip install xlswriter
!pip install visualkeras
import xlswriter
import datetime
import pandas as pd
import seaborn as sns
import wave
import librosa
from matplotlib import cm
import numpy as np
import scipy
from tqdm import tqdm_notebook
import matplotlib.pyplot as plt
import IPython.display as ipd
import tensorflow as tf
from tensorflow import keras
tf.keras.optimizers.Adam
from keras import Sequential
from tensorflow.keras.utils import Sequence, to_categorical
from sklearn.preprocessing import LabelEncoder
from keras import losses, models, optimizers
from keras.activations import relu, softmax
from keras.layers import Dense, Dropout, Input, Convolution2D, BatchNormalization, Activation, MaxPool2D, Flatten
from sklearn.metrics import confusion_matrix
%matplotlib inline

```

Figura 20. Importación de librerías. Fuente propia.

Se imprimen algunas muestras del dataset de entrenamiento, evidenciando que los nombres de archivo y labels son correctamente leídos por el algoritmo:

```
train_df = pd.read_csv(train_labels_path)
test_df = pd.read_csv(test_labels_path)
train_df.head()
```

	fname	label
0	a437.wav	aire
1	s339.wav	otro
2	c681.wav	claxon
3	f792.wav	freno
4	s584.wav	otro

Figura 21. Verificación de cabeceros de archivo. Fuente propia.

Se imprimen datos de duración y sample rate de un audio del dataset, además de generar reproductor para escucha del mismo.

```
fname = os.path.join(train_path, 'a212.wav')
print(fname)

wav = wave.open(fname)
print("Frecuencia de muestreo = ", wav.getframerate())
print("Samples totales de archivo = ", wav.getnframes())
print("Duración = ", wav.getnframes()/wav.getframerate())

ipd.Audio(fname)

/content/drive/My Drive/Colab Notebooks/Tesis/train/a212.wav
Frecuencia de muestreo = 44100
Samples totales de archivo = 15749
Duración = 0.3571201814058957
```



Figura 22. Verificación lectura de archivos. Fuente propia.

Se declaran las clases en las cuales están clasificadas las muestras, correspondientes a la columna “label” de los archivos CSV correspondiente. Las clases ‘freno’, ‘claxon’ y ‘aire’ hacen referencia a los correspondientes eventos sonoros antes definidos. La clase ‘otro’ hace referencia a una selección de otro tipo de sonidos no correspondientes a las tres clases ya definidas, incluyendo tráfico rodado, alarmas, sirenas, perifoneo, entre otros. Esto es declarado también:

```
filtered_classes = ['freno', 'claxon', 'aire', 'otro']
train_df_filtered = train_df[train_df["label"].isin(filtered_classes)]
test_df_filtered = test_df[test_df["label"].isin(filtered_classes)]

print("Número de muestras de entrenamiento: %d"%(train_df_filtered.shape[0]))
print("Número de muestras de validación: %d"%(test_df_filtered.shape[0]))
print("Cantidad de Clases: %d"%(train_df_filtered.label.nunique()))
print("\nClases: ", train_df_filtered.label.unique())

Número de muestras de entrenamiento: 2240
Número de muestras de validación: 960
Cantidad de Clases: 4

Clases: ['aire' 'otro' 'claxon' 'freno']
```

Figura 23. Declaración de clases. Fuente propia.

Se definen las funciones para preparación y procesamiento de los archivos de audio:

```
class Config(object):
    def __init__(self,
                 sampling_rate=16000, audio_duration=2, n_classes=3,
                 learning_rate=0.0001, max_epochs=20, n_mfcc=40):
        self.sampling_rate = sampling_rate
        self.audio_duration = audio_duration
        self.n_classes = n_classes
        self.n_mfcc = n_mfcc
        self.learning_rate = learning_rate
        self.max_epochs = max_epochs
        self.audio_length = self.sampling_rate * self.audio_duration
        self.dim = (self.n_mfcc, 1 + int(np.floor(self.audio_length/512)), 1)

def prepare_data(fnames, config, data_dir):
    X = np.empty(shape=(len(fnames), config.dim[0], config.dim[1], 1))
    input_length = config.audio_length
    for i, fname in tqdm_notebook(enumerate(fnames), total=len(fnames)):
        file_path = os.path.join(data_dir, fname)
        data, _ = librosa.core.load(file_path, sr=config.sampling_rate, res_type="kaiser_fast")
```

Figura 24. Definición de funciones de procesamiento. Fuente propia.

Se define la función para el análisis del archivo y se realiza el ajuste del tamaño del archivo audio agregando ceros a los extremos de los MFCC para igualar duraciones en todos los archivos.

```
# Random offset / Padding
if len(data) > input_length:
    max_offset = len(data) - input_length
    offset = np.random.randint(max_offset)
    data = data[offset:(input_length+offset)]
else:
    if input_length > len(data):
        max_offset = input_length - len(data)
        offset = np.random.randint(max_offset)
    else:
        offset = 0
    data = np.pad(data, (offset, input_length - len(data) - offset), "constant")

data = librosa.feature.mfcc(data, sr=config.sampling_rate, n_mfcc=config.n_mfcc)
data = np.expand_dims(data, axis=-1)
X[i,] = data
return X
```

Figura 25. Definición de padding. Fuente propia.

Se definen las propiedades de los archivos de audio a partir de las librerías importadas, a través de las mismas se encuentran propiedades como nombre del archivo, clase, duración y valores MFCC entre otros.

```
class DataGenerator(Sequence):
    def __init__(self, fnames, labels, base_path, batch_size=32, n_mfcc=40,
                 n_classes=3, audio_duration=2, sampling_rate=44100, shuffle=True):
        self.data_ids = fnames
        self.labels = labels
        self.base_path = base_path
        self.batch_size = batch_size
        self.n_mfcc = n_mfcc
        self.n_classes = n_classes
        self.audio_duration = audio_duration
        self.sampling_rate = sampling_rate
        self.audio_length = audio_duration * self.sampling_rate
        self.dim = (self.n_mfcc, 1 + int(np.floor(self.audio_length/512)), 1)
        self.shuffle = shuffle
        self.on_epoch_end()
```

Figura 26. Definición de propiedades de archivos. Fuente propia.

Se define una función para contabilizar el número de epochs. Con esto, el número de epochs es actualizado después de cada repetición, permitiendo visualizar el progreso y detener el entrenamiento tras transcurrir el número de épocas definido.

Se define además una función para calcular el número de batches por epoch, dado el tamaño del dataset y el batch size determinado, permitiendo al algoritmo determinar el final de un epoch. Este número describe el número de iteraciones por epoch.

Adicional, se crea la función que divide el dataset en el número de batches calculados en la función declarada anterior. Con estas funciones, el algoritmo tiene las herramientas necesarias para dividir el dataset en las porciones que le permitirán procesarlo más adelante, por el número de épocas que se han de definir.

```
def on_epoch_end(self):
    'Updates indices after each epoch'
    self.indices = np.arange(len(self.data_ids))
    if self.shuffle:
        np.random.shuffle(self.indices)

def __len__(self):
    'Denotes the number of batches per epoch'
    return int(np.floor(len(self.data_ids) / self.batch_size))

def __getitem__(self, index):
    'Generate one batch of data'
    # Generate indices of the batch
    list_IDS_temp = self.indices[index*self.batch_size:(index+1)*self.batch_size]
    # Generate data
    X, y = self.__data_generation(list_IDS_temp)

    return X, y
```

Figura 27. Definición de funciones para actualización de índices. Fuente propia.

Se definen funciones para crear los arreglos que contendrán los batches, conservando las categorías de las muestras que cada batch contendrá. Se crean primero arreglos vacíos, de longitud correspondiente al batch size, para luego ser poblados por las muestras del dataset ya cargado.

Además, se crea una función que, por medio de la librería librosa, abre, carga e interpreta el dataset como archivos de audio, declarando además las funciones mediante las cuales se genera el MFCC de cada uno de dichos archivos.

```
def __data_generation(self, list_IDs_temp):
    'Generates data containing batch_size samples'
    # Initialization
    X = np.empty((self.batch_size, *self.dim))
    # Store class
    y = self.labels[list_IDs_temp]
    # Generate data
    for i, index in enumerate(list_IDs_temp):
        X[i,] = self.load_audio_dataset(index)
    X = X.reshape((X.shape, 1))
    return X, to_categorical(y, num_classes=self.n_classes)

def load_audio_dataset(self, index):
    'Load audio dataset'
    f = self.data_ids[index]
    file_path = os.path.join(self.base_path, f)
    data, _ = librosa.core.load(file_path, sr=self.sampling_rate, res_type='kaiser_fast')
    data = self.pad_dataset(data, self.audio_length)
    data = librosa.feature.mfcc(data, sr=self.sampling_rate, n_mfcc=self.n_mfcc)
    data = np.expand_dims(data, axis=-1)
    print("Data shape: ", data.shape)
    return data
```

Figura 28. Funciones generadoras de datos. Fuente propia.

Se define función para ajustar la longitud del tamaño de los datos que son menores a la longitud del audio.

```
def pad_dataset(self, data, audio_length):  
    'Pad dataset if length less than audio_length'  
    # random padding / offset  
    if len(data) > audio_length:  
        max_offset = len(data) - audio_length  
        offset = np.random.randint(max_offset)  
        data = data[offset:(audio_length+offset)]  
    # pad if data is smaller  
    else:  
        if audio_length > len(data):  
            max_offset = audio_length - len(data)  
            offset = np.random.randint(max_offset)  
        else:  
            offset = 0  
        data = np.pad(data, (offset, audio_length - len(data) - offset), "constant")  
    return data
```

Figura 29. Función para aplicar padding. Fuente propia.

Se definen el modelo de convolución y las neuronas del algoritmo.

```
def get_2d_conv_model(config):

    nclass = config.n_classes

    inp = Input(shape=(config.dim[0],config.dim[1],1))

    x = Convolution2D(32, (4,4), padding="same")(inp)
    x = BatchNormalization()(x)
    x = Activation("relu")(x)
    x = MaxPool2D()(x)

    x = Convolution2D(32*2, (5,10), padding="same")(x)
    x = BatchNormalization()(x)
    x = Activation("relu")(x)
    x = MaxPool2D()(x)

    x = Convolution2D(32*3, (4,10), padding="same")(x)
    x = BatchNormalization()(x)
    x = Activation("relu")(x)
    x = MaxPool2D()(x)

    x = Convolution2D(32*3, (4,10), padding="same")(x)
    x = BatchNormalization()(x)
    x = Activation("relu")(x)
    x = MaxPool2D()(x)

    x = Convolution2D(32*3, (4,10), padding="same")(x)
    x = BatchNormalization()(x)
    x = Activation("relu")(x)
    x = MaxPool2D()(x)

    x = Flatten()(x)
    x = Dense(64)(x)
    x = BatchNormalization()(x)
    x = Activation("relu")(x)

    out = Dense(nclass, activation=softmax)(x)

    model = models.Model(inputs=inp, outputs=out)
    opt = tf.keras.optimizers.Adam(config.learning_rate)

    model.compile(optimizer=opt, loss=losses.categorical_crossentropy, metrics=['acc'])
    return model
```

Figura 30. Definición red neuronal, neuronas. Fuente propia.

Se definen las funciones que permiten evaluar el algoritmo como lo son la matriz de confusión, la pérdida y la exactitud, las cuales se grafican para evidenciar los resultados.

```
def plot_confusion_matrix(y_true, y_pred, labels):
    matrix = confusion_matrix(y_true, y_pred)
    fig, ax = plt.subplots(figsize=(12,10))
    plt.imshow(matrix)
    ax.set_xticks(range(len(labels)));
    ax.set_xticklabels(labels, rotation=0)
    ax.set_yticks(range(len(labels)));
    ax.set_yticklabels(labels)
    max_confusions = 0
    confused_classes = (-1, -1)
    for i, true_label in enumerate(matrix):
        for j, predicted_label in enumerate(true_label):
            text = ax.text(j, i, matrix[i, j],
                           ha="center", va="center", color="w");
    plt.tick_params(axis=u'both', which=u'both',length=0)
    plt.title("Matriz de Confusión");

def plot_loss(history):
    from matplotlib import pyplot as plt
    plt.figure(figsize=(10,7))
    plt.plot(history.history['loss'], c="darkblue")
    plt.plot(history.history['val_loss'], c="crimson")
    plt.legend(["Train", "Validation"])
    plt.title("Pérdida")
    plt.xlabel("Iteración")
    plt.ylabel("Pérdida")
    plt.grid(True, alpha = 0.2)
    plt.show()

def plot_acc(history):
    from matplotlib import pyplot as plt
    plt.figure(figsize=(10,7))
    plt.plot(history.history['acc'], c="darkblue")
    plt.plot(history.history['val_acc'], c="crimson")
    plt.legend(["Train", "Validation"])
    plt.title("Exactitud")
    plt.xlabel("Iteración")
    plt.ylabel("Accuracy")
    plt.grid(True, alpha = 0.2)
    plt.show()
```

Figura 31. Funciones para matriz de confusión, y gráficas de pérdida y exactitud. Fuente propia.

Una vez definidas las funciones mediante las cuales se procesarán los datos y se entrenará el algoritmo, se empieza dicho proceso con el dataset existente:

```

train_base_path = os.path.join(base_path, 'train')
test_base_path = os.path.join(base_path, 'test')

X_train_labels = np.array(train_df['label'])
encoder = LabelEncoder()
encoder.fit(X_train_labels)
X_train_fnames = np.array(train_df['fname'])
X_train_labels = encoder.transform(X_train_labels)

X_test_fnames = np.array(test_df['fname'])
X_test_labels = np.array(test_df['label'])
X_test_labels = encoder.transform(X_test_labels)

config = Config(sampling_rate=44100, audio_duration=2, learning_rate=0.0001, n_mfcc=40, n_classes=4)
X_train = prepare_data(X_train_fnames, config, train_base_path)
y_train = to_categorical(X_train_labels, num_classes=config.n_classes)

X_test = prepare_data(X_test_fnames, config, test_base_path)
y_test = to_categorical(X_test_labels, num_classes=config.n_classes)

mean = np.mean(X_train, axis=0)
std = np.std(X_train, axis=0)

X_train = (X_train - mean)/std
X_test = (X_test - mean)/std

np.save(base_path + "/X_train_all.npy", X_train)
np.save(base_path + "/y_train_all.npy", y_train)
np.save(base_path + "/X_test_all.npy", X_test)
np.save(base_path + "/y_test_all.npy", y_test)

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:17: TqdmDeprecationWarning: This function will be removed in tqdm==5.0.0
Please use `tqdm.notebook.tqdm` instead of `tqdm.tqdm_notebook`
100% ██████████ 2240/2240 [08:46<00:00, 5.06it/s]

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:17: TqdmDeprecationWarning: This function will be removed in tqdm==5.0.0
Please use `tqdm.notebook.tqdm` instead of `tqdm.tqdm_notebook`
100% ██████████ 960/960 [04:04<00:00, 4.43it/s]

```

Figura 32. Procesamiento de datasets establecidos. Fuente propia.

Se muestra luego la estructura del modelo, mostrando las formas, pooling y funciones de activación correspondientes. Este modelo, con estas especificaciones, es el que se usará para el entrenamiento. Se genera además una representación gráfica de la arquitectura de la red neuronal, visible en el Anexo III.

```
config = Config(sampling_rate=44100, audio_duration=2, learning_rate=0.08, n_mfcc=40, n_classes=4)
model = get_2d_conv_model(config)
model.summary()
import visualkeras
visualkeras.layered_view(model)
```

Model: "model"

Layer (type)	Output Shape	Param #
=====		
input_1 (InputLayer)	[(None, 40, 173, 1)]	0
conv2d (Conv2D)	(None, 40, 173, 32)	544
batch_normalization (Batch Normalization)	(None, 40, 173, 32)	128
activation (Activation)	(None, 40, 173, 32)	0
max_pooling2d (MaxPooling2D)	(None, 20, 86, 32)	0
conv2d_1 (Conv2D)	(None, 20, 86, 64)	81984
batch_normalization_1 (Batch Normalization)	(None, 20, 86, 64)	256
activation_1 (Activation)	(None, 20, 86, 64)	0
max_pooling2d_1 (MaxPooling2D)	(None, 10, 43, 64)	0
conv2d_2 (Conv2D)	(None, 10, 43, 96)	245856
batch_normalization_2 (Batch Normalization)	(None, 10, 43, 96)	384
activation_2 (Activation)	(None, 10, 43, 96)	0
max_pooling2d_2 (MaxPooling2D)	(None, 5, 21, 96)	0
conv2d_3 (Conv2D)	(None, 5, 21, 96)	368736
batch_normalization_3 (Batch Normalization)	(None, 5, 21, 96)	384
activation_3 (Activation)	(None, 5, 21, 96)	0
max_pooling2d_3 (MaxPooling2D)	(None, 2, 10, 96)	0
flatten (Flatten)	(None, 1920)	0
dense (Dense)	(None, 64)	122944

Figura 33. Impresión forma red neuronal, parte 1. Fuente propia.

batch_normalization_4 (Batch Normalization)	(None, 64)	256
activation_4 (Activation)	(None, 64)	0
dense_1 (Dense)	(None, 3)	195
=====		
Total params: 821,667		
Trainable params: 820,963		
Non-trainable params: 704		

Figura 34. Impresión forma red neuronal, parte 2. Fuente propia.

Se escribe un archivo log para registrar las características de ejecución y el momento en que se hizo el entrenamiento, para posteriormente empezar con el fitting (o entrenamiento) del sistema. Se escoge un batch size de 250 elementos, y de igual manera 250 epochs (véase *Marco Teórico*), basados en las gráficas después presentadas:

```
!rm -rf '/content/drive/My Drive/Colab Notebooks/ML_sound_classification_project/logs/'
log_dir = os.path.join(base_path, 'logs', datetime.datetime.now().strftime("%Y%m%d-%H%M%S"))
from keras.callbacks import TensorBoard
tensorboard_callback = TensorBoard(log_dir=log_dir, write_graph=True)

history = model.fit(X_train, y_train,
                    validation_data=(X_test, y_test),
                    batch_size=250,
                    callbacks=[tensorboard_callback],
                    epochs=250)

Epoch 1/250
9/9 [=====] - 40s 748ms/step - loss: 1.4181 - acc: 0.4304 - val_loss: 38007.2734 - val_acc: 0.2500
Epoch 2/250
9/9 [=====] - 3s 329ms/step - loss: 1.0612 - acc: 0.5304 - val_loss: 1255.9995 - val_acc: 0.3396
Epoch 3/250
9/9 [=====] - 3s 323ms/step - loss: 0.9400 - acc: 0.5964 - val_loss: 386.2606 - val_acc: 0.2500
Epoch 4/250
9/9 [=====] - 3s 332ms/step - loss: 0.8992 - acc: 0.6027 - val_loss: 186.4669 - val_acc: 0.2500
Epoch 5/250
9/9 [=====] - 3s 330ms/step - loss: 0.8087 - acc: 0.6621 - val_loss: 15.0559 - val_acc: 0.4427
Epoch 6/250
9/9 [=====] - 3s 330ms/step - loss: 0.7625 - acc: 0.6893 - val_loss: 26.9818 - val_acc: 0.4990
Epoch 7/250
9/9 [=====] - 3s 329ms/step - loss: 0.7107 - acc: 0.7214 - val_loss: 21.1958 - val_acc: 0.4458
Epoch 8/250
```

Figura 35. Entrenamiento de modelo. Fuente propia.

Por medio de las funciones ya definidas arriba, se grafica la pérdida del sistema durante el entrenamiento, resaltando en Azul la pérdida de entrenamiento y en Rojo la pérdida de validación:

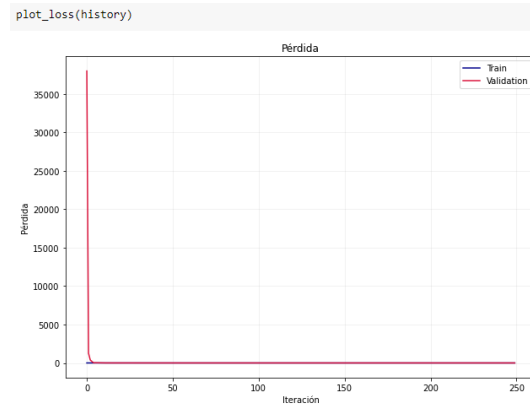


Figura 36. Gráfica de pérdida a lo largo de entrenamiento. Fuente propia.

Se repite éste mismo proceso para la exactitud, conservando el patrón de colores:

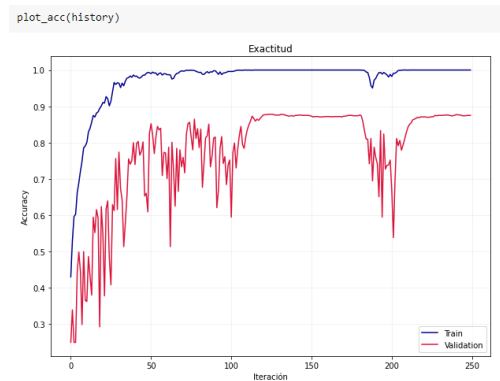


Figura 37. Gráfica de exactitud a lo largo de entrenamiento. Fuente propia.

Por medio del dataset de validación, se hace una predicción de elementos para generar una matriz de confusión mediante la cual pueda ser evaluado objetivamente el algoritmo. Esto se hace a través de la función `model.predict()`, encontrando los porcentajes de predicción más altos por

medio de `np.argmax()`. Se imprime de una vez la matriz de confusión correspondiente, la cual se analizará en la sección *Análisis de Resultados*.

```
y_pred_ = model.predict(X_test)

y_pred = np.argmax(y_pred_, axis=1)
y_true = np.argmax(y_test, axis=1)
labels = encoder.classes_
plot_confusion_matrix(y_true, y_pred, labels)
```

Figura 38. Generación matriz de confusión. Fuente propia.

Indicada la respuesta con mayor probabilidad para cada predicción, se almacena en un arreglo la lista de eventos encontrados en el proceso, y este arreglo es impreso:

```
res=[]
for i in range (0,len(y_pred)):
    if y_pred[i]==0:
        resi="aire"
        res.append(resi)
    elif y_pred[i]==1:
        resi="claxon"
        res.append(resi)
    elif y_pred[i]==2:
        resi="freno"
        res.append(resi)
    elif y_pred[i]==3:
        resi="otro"
        res.append(resi)
print(res)

['claxon', 'aire', 'claxon', 'claxon', 'aire', 'aire', 'freno', 'aire', 'claxon',
```

Figura 39. Creación arreglo con predicciones. Fuente propia.

Para evaluar nuevos archivos, independientes a aquellos mediante los cuales fue entrenado y validado el sistema, se permite el procesamiento de una grabación determinada por el usuario, la cual recomendablemente es de larga duración, de tráfico rodado diurno en condiciones climáticas favorables. También se solicita un archivo de calibración para el cálculo del nivel de presión sonora

RMS (raíz cuadrada media) de cada segmento de audio definido por el usuario. La predicción de eventos sonoros no deseados se hará también en los intervalos definidos por el usuario, en segundos enteros. Se crea además una carpeta, de nombre *chopped*, en la cual se almacenarán los segmentos.

```
!mkdir chopped

calib = '/content/drive/My Drive/Colab Notebooks/SPLs/Zoom_94dB.wav' #Nombre del archivo de calibración.
sr1=44100
x2,sr2 = librosa.load(calib,sr=44100)
# prueba = '/content/drive/My Drive/Colab Notebooks/SPLs/Hayuelos_Primer_Punto_EQ.wav' #Nombre de la grabación.
prueba = '/content/drive/My Drive/Colab Notebooks/SPLs/Hayuelos 2 EQ.wav'
segmento=0.999 #Longitud, en segundos, de los segmentos a analizar.
sr1=44100
chunk=segmento*sr1
chunk=round(chunk,1)
chunk=int(chunk)
x1,sr1 = librosa.load(prueba,sr=44100)
x1=np.array(x1)
mod=len(x1) % sr1*segmento
```

Figura 40. Lectura y preprocesamiento de archivo nuevo. Fuente propia.

Se divide el archivo de audio en los segmentos correspondientes y, hallando el valor de calibración a partir del archivo provisto, se calcula el valor RMS de presión sonora para cada segmento. En el presente código se añade la corrección de +4 dB dada por un error en la fuente de datos, como se explica en la sección 4.6, Obtención de fórmula para cálculo de dBSPL. Para cualquier otra aplicación, el valor visualizado como 98 debe ser 94, como indica la Fórmula 22:

```
output=[x1[i:i + chunk] for i in range(0, len(x1), chunk)]
output=np.array(output)
calibr=max(x2)
SPL=[]
outputs=[]
for i in range (0,len(output)):
    rms=np.square(output[i])
    rms=np.sum(rms)
    rms=rms/(len(output[i]))
    rms=np.sqrt(rms)
    spli=8.7*np.log(rms)-8.7*np.log(calibr)+98
    spli=round(spli,1)
    SPL.append(spli)
SPL=np.array(SPL)

rms=np.square(x1)
rms=np.sum(rms)
rms=rms/(len(x1))
rms=np.sqrt(rms)
Leq=8.7*np.log(rms)-8.7*np.log(calibr)+98
Leq=round(Leq,4)
SPLs=str(SPL)
```

Figura 41. Fragmentación y cálculo de nivel RMS de segmentos. Fuente propia.

La preparación de los datos para la predicción requiere que estos tengan un nombre, una configuración inicial, y un *path*, por lo cual es necesario crear archivos en formato .wav dentro de una carpeta para este fin, propósito por el cual fue creada la carpeta *chopped* anteriormente. Estos archivos generados son nombrados en su orden de generación, y para ser guardados en la carpeta especificada es necesario cambiar de directorio mientras se produce dicho proceso:

```
from scipy.io.wavfile import write
nombres=[]
archivo=[]
%cd /content/chopped
for i in range (0,len(output)):
    a=i
    a=str(a)
    b=".wav"
    c=a+b
    nombres.append(c)
    scipy.io.wavfile.write(nombres[i], 44100, output[i])
%cd /content


/content/chopped
/content
```

Figura 42. Guardado de fragmentos en carpeta temporal. Fuente propia.

Teniendo el archivo de audio original dividido en secciones de igual longitud y almacenados como archivos de audio en una carpeta temporal, se procede con la preparación de los datos y la predicción de los mismos:

```
path_chop="/content/chopped"
config = Config(sampling_rate=44100, audio_duration=2, learning_rate=0.008, n_classes=4)
mfccs = prepare_data(nombres, config, path_chop)
mfccs = (mfccs - mean)/std
y_pred_ = model.predict(mfccs, use_multiprocessing=True, workers=6, verbose=1)
```

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:17: TqdmDeprecationWarning: Please use `tqdm.notebook.tqdm` instead of `tqdm.tqdm_notebook`

100%  893/893 [00:16<00:00, 54.21it/s]

28/28 [=====] - 0s 16ms/step

Figura 43. Procesamiento y predicción de datos. Fuente propia.

Una vez hecha la predicción, se utiliza un método similar al utilizado para la generación de la matriz de confusión anterior, detectando la clase más probable para cada muestra analizada y almacenando el valor correspondiente para luego ser impreso. Se hace además un conteo de cada tipo de evento:

```
y_pred = np.argmax(y_pred_, axis=1)
res=[]
aires=0
claxons=0
frenos=0
otross=0
for i in range (0,len(y_pred)):
    if y_pred[i]==0:
        resi="aire"
        res.append(resi)
        aires=aires+1
    elif y_pred[i]==1:
        resi="claxon"
        res.append(resi)
        claxons=claxons+1
    elif y_pred[i]==2:
        resi="freno"
        res.append(resi)
        frenos=frenos+1
    elif y_pred[i]==3:
        resi="otro"
        res.append(resi)
        otross=otross+1

print("Salidas de Aire a Presión: ",aires)
print("Claxons: ",claxons)
print("Chillidos de pastillas de Freno: ",frenos)
print("Otros: ",otross)
```

Salidas de Aire a Presión: 12
 Claxons: 53
 Chillidos de pastillas de Freno: 115
 Otros: 713

Figura 44. Creación de arreglo y conteo de eventos sonoros. Fuente propia.

Para obtener el resultado de la eliminación de los eventos sonoros no deseados de la grabación original, se genera un archivo de audio con todos los eventos sonoros detectados como “otro”, es decir, aquellos no pertenecientes a las categorías de “aire”, “freno” o “claxon”. Dicho archivo recibe el nombre de “Audio Limpio.wav”. Se generan también archivos de audio con la suma de cada categoría de Evento Sonoro No Deseado, y uno con la suma de todos los Eventos Sonoros No Deseados detectados, con el nombre de “Audio ESND.wav”:

```
AudioLimpio=[]
AudioClaxon=[]
AudioFreno=[]
AudioAire=[]
AudioESND=[]
for i in range(0,len(res)-1):
    if y_pred[i]==3:
        a=output[i]
        AudioLimpio.append(a)
    if y_pred[i]==0:
        b=output[i]
        AudioAire.append(b)
    if y_pred[i]==1:
        c=output[i]
        AudioClaxon.append(c)
    if y_pred[i]==2:
        d=output[i]
        AudioFreno.append(d)
    if y_pred[i]!=3:
        e=output[i]
        AudioESND.append(e)

AudioLimpio=np.array(AudioLimpio)
AudioLimpio = AudioLimpio.flatten()
AudioClaxon=np.array(AudioClaxon)
AudioClaxon = AudioClaxon.flatten()
AudioFreno=np.array(AudioFreno)
AudioFreno = AudioFreno.flatten()
AudioAire=np.array(AudioAire)
AudioAire = AudioAire.flatten()
AudioESND=np.array(AudioESND)
AudioESND = AudioESND.flatten()

scipy.io.wavfile.write('Audio Limpio.wav', 44100, AudioLimpio)
scipy.io.wavfile.write('Audio Aire.wav', 44100, AudioAire)
scipy.io.wavfile.write('Audio Freno.wav', 44100, AudioFreno)
scipy.io.wavfile.write('Audio Claxon.wav', 44100, AudioClaxon)
scipy.io.wavfile.write('Audio ESNDs.wav', 44100, AudioESND)
```

Figura 45. Generación de archivos derivados. Fuente propia.

Se calcula en la siguiente sección el Nivel de Presión Sonora Equivalente (L_{eq}) para el audio proveído, en los segmentos indicados por el usuario. Aquí también se calcula un L_{eq} total, un L_{eq} para el audio dada la eliminación de ESNDs (correspondiente al archivo “Audio Limpio.wav”), y un L_{eq} correspondiente a la suma de todos los ESNDs consecutivos. Los resultados de este proceso son estudiados en la sección 5.3 del presente documento:

```

rms=np.square(AudioLimpio)
rms=np.sum(rms)
rms=rms/(len(AudioLimpio))
rms=np.sqrt(rms)
LeqL=8.7*np.log(rms)-8.7*np.log(calibr)+98
LeqL=round(LeqL,4)

rms=np.square(AudioESND)
rms=np.sum(rms)
rms=rms/(len(AudioESND))
rms=np.sqrt(rms)
LeqE=8.7*np.log(rms)-8.7*np.log(calibr)+98
LeqE=round(LeqE,4)

print(Leq)
print(LeqL)
print(LeqE)

outputl=[AudioLimpio[i:i + chunk] for i in range(0, len(AudioLimpio), chunk)]
outputl=np.array(outputl)

SPLL=[]

for i in range(0,len(res)):
    if res[i]=='otro':
        spli=SPL[i]
        SPLL.append(spli)

```

Figura 46. Cálculo Nivel Sonoro Equivalente para audios. Fuente propia.

Con el fin de poder conservar los datos encontrados en cada ejecución y hacer el análisis adecuado a los resultados allí encontrados, se genera un archivo de Excel (en formato .xlsx) con cuatro columnas: Marca de inicio del segmento, marca de final del segmento, evento sonoro hallado en el segmento y nivel de presión sonora (RMS) del segmento. Este se genera a la carpeta base de la ejecución y puede ser accedido a través del panel lateral izquierdo, con el nombre de “Resultados.xlsx”:

```

inicio=[]
final=[]
for i in range(0,len(y_pred)):
    ini=i*chunk
    ini=ini/sr1
    ini=str(datetime.timedelta(seconds=ini))
    fin=(i+1)*chunk
    fin=fin/sr1
    fin=str(datetime.timedelta(seconds=fin))
    inicio.append(ini)
    final.append(fin)

workbook = xlswriter.Workbook('Resultados.xlsx')
worksheet = workbook.add_worksheet(name='Audio Original')
worksheet.write('A1', 'Inicio')
worksheet.write('B1', 'Final')
worksheet.write('C1', 'Sonido')
worksheet.write('D1', 'dBSPL (RMS)')

for i in range(0,len(res)):
    j=i+2
    indice=str(j)
    celdaA='A'+indice
    worksheet.write(celdaA, inicio[i])
    celdaB='B'+indice
    worksheet.write(celdaB, final[i])
    celdaC='C'+indice
    worksheet.write(celdaC, res[i])
    celdaD='D'+indice
    worksheet.write(celdaD, SPL[i])

```

Figura 47. Generación archivo de Excel con predicciones y niveles sonoros. Fuente propia.

Dentro del mismo archivo .xlsx generado, se guarda otra Hoja de nombre “Audio sin ESNDs”, mostrando los valores correspondientes al archivo de audio cuyos eventos sonoros no deseados

fueron eliminados. La lista resultante, al igual que el archivo de audio, son de menor longitud que el archivo original:

```

worksheet = workbook.add_worksheet(name='Audio sin ESNDs')
worksheet.write('A1', 'Inicio')
worksheet.write('B1', 'Final')
worksheet.write('C1', 'Sonido')
worksheet.write('D1', 'dBSPL (RMS)')

for i in range(0, len(output1)):
    j=i+2
    indice=str(j)
    celdaA='A'+indice
    worksheet.write(celdaA, inicio[i])
    celdaB='B'+indice
    worksheet.write(celdaB, final[i])
    celdaC='C'+indice
    worksheet.write(celdaC, 'otro')
    celdaD='D'+indice
    worksheet.write(celdaD, SPL[i])

workbook.close()

```

Figura 48. Inclusión audio depurado a archivo de Excel. Fuente propia.

Capítulo 5. Presentación y Análisis de Resultados

5.1. Métricas de evaluación

Como parte del desarrollo del algoritmo, se produce una matriz de confusión compuesta por las predicciones hechas por el mismo en el dataset de validación, compuesto por 960 muestras, divididas en las cuatro categorías definidas. A partir de esto se calculan y presentan las métricas estadísticas propuestas en el tercer objetivo específico.

$$Precisión (Precision) = \frac{TP}{TP+FP} \quad (11)$$

$$Exactitud (Accuracy) = \frac{TP+TN}{P+N} \quad (12)$$

Teniendo en cuenta las ecuaciones 10 y 11, ya descritas para la obtención de Exactitud y Precisión, cabe aclarar los valores que se tendrán en cuenta para dicho cálculo, ya que el presente es un ejemplo de una matriz de confusión multiclase. Se da pues la métrica de precisión especificada para cada clase, tomando como Verdaderos Positivos (TP) aquellas predicciones correctas, y Falsos Positivos (FP) aquellas predicciones que se indicaron como de la categoría estudiada, más eran de otra categoría.

Se obtiene, por medio del algoritmo desarrollado, la siguiente matriz de confusión, la cual indica el resultado de predicción a través del dataset de validación.

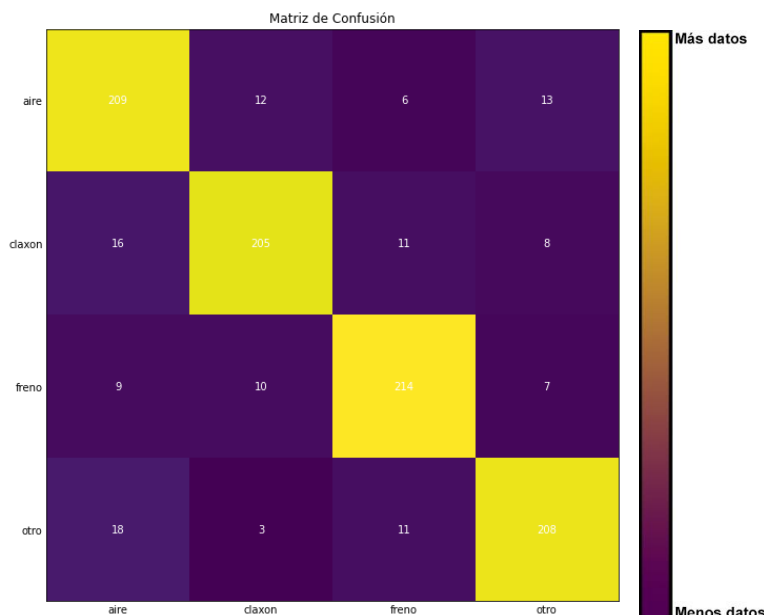


Figura 49. Matriz de confusión (con convención de colores) para el algoritmo generado. Fuente Propia

5.1.1. Análisis de precisión por categoría

Para la categoría de “Aire”, conteniendo salidas de aire a presión, se observan 209 muestras clasificadas correctamente, resaltadas en color amarillo. Por otra parte, 16 muestras que debían ser clasificadas como “Claxon”, fueron clasificadas como Aire, al igual que 9 que debían ser “Freno”, y 18 que debían ser “Otro”, o sin clasificar. Se tiene pues un valor de Verdaderos Positivos (TP) de 209, y Falsos Positivos (FP), de 43 (suma de las cantidades restantes). Se aplica pues la fórmula 11, arriba descrita, para hallar la precisión de la categoría “Aire”.

$$Precisión = \frac{TP}{TP + FP} = \frac{209}{209 + 43} = 0.829 \approx 83\%$$

Se observa pues una precisión del 83% para la clasificación de muestras sonoras asociadas a ruido por salida de aire a presión.

Para la categoría “Claxon”, correspondiente a los “pitidos” de vehículos de todo tamaño, se obtienen valores de 205 para Verdaderos Positivos (TP), y 25 Falsos Positivos (FP).

Aplicando la fórmula ya descrita, se obtiene el valor de precisión:

$$Precisión = \frac{TP}{TP + FP} = \frac{205}{205 + 25} = 0.891 \approx 89\%$$

Se obtiene así una precisión del 89% en la clasificación de Eventos Sonoros asociados a claxon de vehículos de todo tamaño.

Para la categoría “Freno”, correspondiente al “chillido” de las pastillas de freno con el rotor, se obtienen valores de 214 para Verdaderos Positivos (TP), y 28 Falsos Positivos (FP).

Aplicando la fórmula ya descrita, se obtiene el valor de precisión:

$$Precisión = \frac{TP}{TP + FP} = \frac{214}{214 + 28} = 0.884 \approx 88\%$$

Se obtiene así una precisión del 88% en la clasificación de Eventos Sonoros asociados a chillido de pastillas de freno.

Finalmente, para la categoría “Otro”, incluyendo ruido de motor y rodamiento, además de otros sonidos fuera del alcance del presente trabajo, se obtienen valores de 208 para Verdaderos Positivos (TP), y 28 Falsos Positivos (FP).

Aplicando la fórmula ya descrita, se obtiene el valor de precisión:

$$Precisión = \frac{TP}{TP + FP} = \frac{208}{208 + 28} = 0.881 \approx 88\%$$

Se obtiene así una precisión del 88% en la clasificación de Eventos Sonoros no clasificados o correspondientes a ruido de motor, exhosto, o rodamiento.

5.1.2. Evaluación de exactitud de algoritmo

La exactitud es una métrica dada para el desempeño general del algoritmo, por lo cual los valores a considerar se toman de la matriz de confusión entera, no por categoría.

$$Exactitud (Accuracy) = \frac{TP + TN}{P + N} = \frac{209 + 205 + 214 + 208}{960} = 0.871 \approx 87\%$$

Se determina pues, finalmente, una exactitud del 87% para el algoritmo desarrollado en el presente trabajo. Cabe resaltar que P+N hace referencia al total de los datos registrados.

5.2. Evaluación con muestras externas

Las muestras externas son tomadas en colaboración con el profesor Óscar Esneider Acosta Agudelo, y como parte de las capturas realizadas para la materia de Acústica Ambiental en la Universidad de San Buenaventura sede Bogotá durante el semestre 2021-2, se obtienen grabaciones de ruido por tráfico rodado capturadas a través de equipos calibrables, en un proceso descrito en la siguiente sección.

5.2.1. Instrumentos utilizados

La toma de muestras se realizó con los siguientes equipos:



Figura 50. Micrófono de medición miniDSP UMIK-1. Fuente: (miniDSP, n.d.)

- Pistófono SVANTEK SV30A



Figura 51. Pistófono SVANTEK SV30A. Fuente: Dr. Jordan Design, n.d.)

- Trípode 4 metros



Figura 52. Trípode metálico. Fuente: (Dixten, n.d.)

- Cámara Apeman A80



Figura 53. Apeman A80. Fuente: (Memorycow, n.d.)

- HandyRec Zoom



Figura 54. Captura de software HandyRec. Fuente: (Apkpure, n.d.)

- Software SvanPC++ SVANTEK



Figura 55. Logo SVANTEK SVANPC++. Fuente: (Svantek, n.d.)

- Vantage Vue Precision Weather Station



Figura 56. Estación meteorológica Vantage Vue. Fuente: (Davis Instruments, n.d.)

5.2.2. Proceso de recolección de datos

Para esta recolección de datos se ubicaron dos puntos localizados sobre la Av. Ciudad de Cali, entre la calle 19A y la calle 22. El primer punto se ubicó frente al parqueadero del almacén Jumbo en el sentido sur-norte, el segundo punto se ubicó en un separador vial en el sentido norte-sur. Al iniciar se aplicaron configuraciones para ajustar la medición del sonómetro en dBA y lineal, dado el Artículo 20 de la Resolución 0627 de 2006 se registran los datos meteorológicos para cumplir con los requerimientos climatológicos de la medición los cuales especifican que la medición debe realizarse en tiempo seco, sin humedad, lluvia o truenos además de una velocidad del viento que no exceda los 3m/s, por tal motivo y haciendo uso del Vantage Vue se registró una velocidad del viento máxima en 2m/s y una temperatura promedio de 19°C. El procedimiento a continuación descrito se realizó de igual forma para los dos puntos.

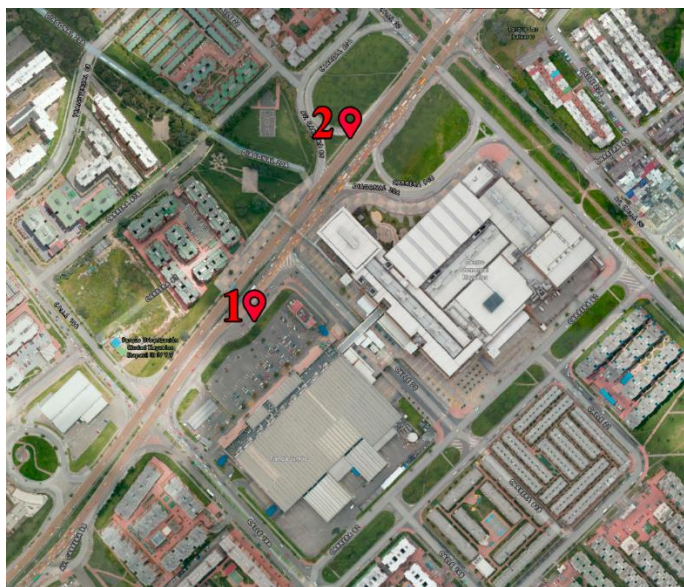


Figura 57. Puntos de grabación. Fuentes: (IDECA, 2015), (Flaticon, n.d.)

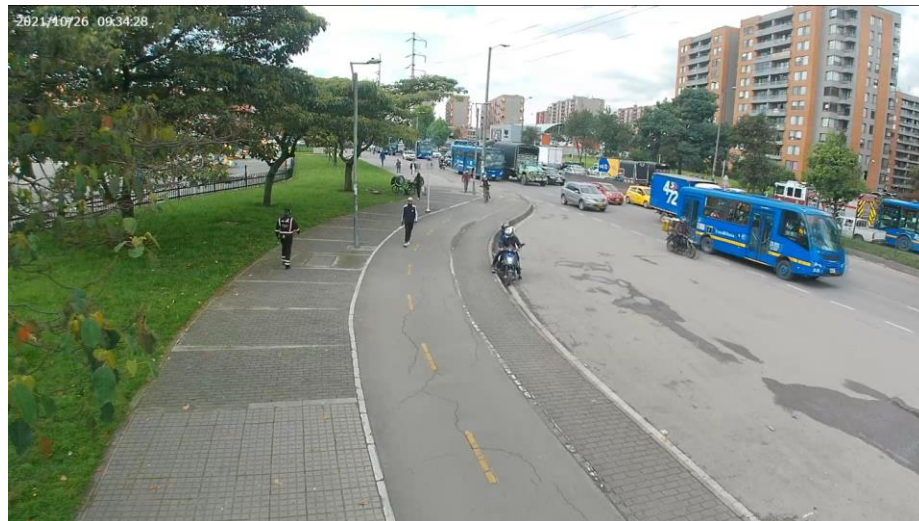


Figura 58. Vista desde primer punto de grabación. Fuente propia.



Figura 59. Vista desde segundo punto de grabación. Fuente propia.

Inicialmente se instaló el micrófono al igual que la cámara en el trípode a cuatro metros del suelo, apuntando a la Avenida Ciudad de Cali, previamente al micrófono se le añadió el filtro anti viento. Luego de tener todas las configuraciones adecuadas para la toma de muestras se realiza la medición haciendo uso de la App Handy Recorder Zoom con un tiempo de medición total de 22 minutos en intervalos de aproximados 3 minutos, habiendo realizado la toma de las muestras se

procedió a extraer y digitalizar los datos con el fin de obtener los valores del espectrograma y data logger similares a los obtenidos con el sonómetro en SvanPC++.

5.2.3. Comparación con datos obtenidos

El procesamiento posterior de los datos, dados los archivos de calibración, fue hecho por medio del software dBFA Suite⁴, el cual permite procesamiento de los datos en segmentos definibles, con un valor escogido de 999 milisegundos para la presente aplicación. El software escogido divide el audio provisto en secciones de la duración definida, haciendo cálculo del Nivel Sonoro Equivalente L_{eq} para cada uno de los segmentos. El archivo de calibración es utilizado para determinar el nivel digital asociado al Nivel de Presión Sonora establecido en la calibración, usualmente 94 dB, caso tal para la aplicación aquí presentada. Especificando la no aplicación de ponderación, temporal o frecuencial, dBFA Suite genera un archivo de texto con los valores correspondientes al L_{eq} de cada segmento, en el presente caso correspondiente a 999 milisegundos.

Para evaluar dicho material en el algoritmo desarrollado, se hace inclusión del mismo, junto con el archivo de calibración correspondiente, en el entorno de ejecución virtual dentro del cual se ejecuta el código, y utilizando las herramientas ya descritas en el apartado 2.5, Desarrollo del Algoritmo, para su procesamiento y cálculo.

Tras crear un análisis por medio del software descrito, se compara con los resultados obtenidos por medio de la función de cálculo de nivel de presión sonora incluido en el algoritmo desarrollado. Dado que el algoritmo desarrollado utiliza índices enteros para la segmentación del audio

⁴ En el presente párrafo se describe el proceso hecho por un tercero, el Ingeniero Óscar Acosta, para la obtención de los datos. Dichos datos fueron entregados en un archivo .txt, generado a través del software dBFA Suite. Se conserva la implementación de dicho software con el fin de no añadir alteraciones al material fuente.

ingresado, es necesario redondear valores no enteros correspondientes a tamaños de segmento diferentes a aquellos múltiplos de 1. Es decir, mientras que el software utilizado para el análisis puede generar una tabla de datos con un tamaño de ventana de exactamente 999 milisegundos, en el caso de una grabación con una frecuencia de muestreo de 44100 Hz correspondiente a 44055.9 muestras por segundo, el algoritmo generado redondea dicho valor hacia 44056 Hz, desplazando el valor definido de 999 milisegundos a 999.0023 milisegundos, por lo cual a lo largo de la duración del archivo utilizado estas diferencias se van acumulando, causando un aumento en la inexactitud del cálculo, dado que se están analizando segmentos totalmente diferentes. No obstante, haciendo un análisis de los primeros 30 segundos de audio dados los parámetros establecidos, hallando la diferencia entre los valores obtenidos por medio del software dBFA Suite y el algoritmo aquí descrito, se obtuvo el comportamiento mostrado en la Tabla 5. Es importante resaltar aquí que, dado el error en la obtención de datos encontrado en la sección 4.6, Obtención fórmula para cálculo de dBSPL, en donde debe compensarse con +4 dB los datos obtenidos, esa corrección es hecha para hallar los siguientes resultados. No obstante, dados datos correctamente medidos y calibrados, esta corrección no será necesaria y la Fórmula 22 será aplicable sin constantes o modificaciones adicionales.

Tabla 5. Error entre predicción y datos. Fuente propia.

dB SPL (Algoritmo)	dB SPL (dBFA Suite)	Error
82,4	82,4	0
83,3	83,4	0,1
82,5	82,6	0,1
80,2	80,2	0
80,1	80,1	0
80,3	80,4	0,1
78,7	78,6	-0,1
78,3	78,5	0,2
79,5	79,5	0
78,7	78,8	0,1
79,7	79,3	-0,4
79,1	79,7	0,6
79,2	79,1	-0,1
79,5	79,5	0
78,5	78,6	0,1
79	79,1	0,1
79,3	79,4	0,1
78	78	0
77,6	77,6	0
77,7	78	0,3
78,7	78,7	0
77,8	78	0,2
78,4	78,3	-0,1
77,8	77,5	-0,3
78,8	78,8	0
77,7	77,9	0,2
77,9	77,9	0
79	78,9	-0,1
78,3	78,5	0,2
79,4	79,5	0,1

Dichos errores se obtienen de la siguiente manera, método también utilizado para el análisis estadístico posterior:

$$\Delta dB = dB SPL_{dBFA Suite} - dB SPL_{dBFA Suite} [dB] \quad (25)$$

De los resultados así obtenidos se hace un diagrama de caja y bigotes, el cual proporciona una representación visual de la distribución de errores y su tendencia; para esto se hace uso de los

primeros 10 minutos de grabación, correspondiente a 601 datos⁵. El diagrama generado puede observarse en la Figura 60.

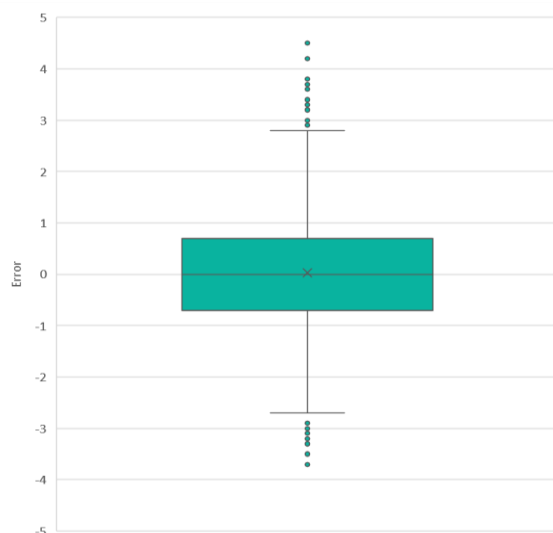


Figura 60. Diagrama de caja y bigotes de error entre predicción y datos. Fuente Propia

El diagrama generado indica una gran precisión del método de cálculo, ubicando el Cuartil 1 en -0.7, el Cuartil 3 en 0.7, y la Mediana, o Cuartil 2, en exactamente 0. De allí se puede hallar una distancia intercuartil de 1.4, y en comparación con una media calculada en 0.026, se puede notar un marginal sesgo positivo, indicando una muy ligera tendencia a calcular valores menores a aquellos generados por dBFA Suite. Dados estos valores, puede esperarse un desempeño confiable, teniendo en cuenta que el 50% de los datos se ubican dentro de un error de 1.4 dB⁶.

⁵ Para los datos completos registrados para esta sección, véase el Anexo I: Tabla de datos.

⁶ Todos los números en este párrafo son dados en dB.

Comprendiendo el desempeño del algoritmo, se procede a utilizar sus resultados para calcular el aporte brindado por los Eventos Sonoros No Deseados a las grabaciones digitales de ruido por tráfico rodado.

5.3. Análisis aporte de ESNDs a cálculo de Nivel de Presión Sonora

Por medio de dos grabaciones del mismo lugar, obtenidas mediante el proceso descrito en el apartado 5.2, se determina el Nivel de Presión Sonora equivalente (L_{eq}) para la grabación original, con los Eventos Sonoros incluidos, con el fin de ser comparados con el audio generado al eliminar todos aquellos Eventos Sonoros No Deseados encontrados durante la ejecución del algoritmo. El proceso para dicha eliminación involucra la unión de todos aquellos segmentos que, al ser analizados, fueran marcados con la etiqueta “otros”, indicando la ausencia de Eventos Sonoros No Deseados, y la generación de un archivo de audio producto de esta unión. Al hacer dicha comparación, se analiza pues el efecto de los eventos sonoros no deseados en una medición de ruido por tráfico rodado, en este caso, para la obtención del Nivel Equivalente.

5.3.1. Primer punto de grabación

Para el primer punto de grabación, se obtiene en primera instancia, por medio de la fórmula 23, un valor de Nivel de Presión Sonora equivalente para la grabación de ruido de tráfico original, obteniendo un valor de 81.16 dBSPL.

Posteriormente, habiendo separado aquellas muestras identificadas como alguna de las tres categorías de Eventos Sonoros No Deseados, y generando un archivo de audio depurado, con dichas muestras eliminadas, se calcula el Nivel de Presión Sonora equivalente para el mismo, obteniendo un valor de 81.19 dBPSL.

Finalmente, se calcula el Nivel de Presión Sonora equivalente para la suma de todos los Eventos Sonoros No Deseados hallados en el archivo de audio, encontrando un valor de 81.11 dBSPL.

Para este caso, es de notar que el audio resultante al eliminar los Eventos Sonoros No Deseados es de mayor amplitud que aquel original, sin embargo, esto puede indicar que, aunque el efecto no es necesariamente de aumento, existe, y afecta el resultado final. En este caso, se nota una diferencia de 0.03 dB, lo cual puede parecer poco, pero demuestra que, si el propósito es la captura de ruido por tráfico rodado específicamente (ruido de motor/exhosto, y ruido de contacto neumático-pavimento), la presencia de eventos sonoros distintos a aquellos hace mella y sesga el resultado.

5.3.2. Segundo punto de grabación

Para la grabación obtenida en el segundo punto de grabación se siguió el mismo protocolo que en el primero. Se obtuvo así un valor para la toma original de 82.27 dBSPL.

Para el archivo de audio depurado, sin los Eventos Sonoros No Deseados, se obtuvo entonces un valor de 82.18 dBSPL.

Finalmente, para la suma de todos los eventos sonoros no deseados, comprendiendo aquellos sonidos que se buscan retirar de la grabación original, se obtiene un valor de 82.86 dBSPL.

En este caso es de notar que el aporte de los Eventos Sonoros No Deseados causa un aumento en el resultado final, con una diferencia de casi 0.1 dB entre ambos escenarios. Es de notar además el considerablemente mayor Nivel Equivalente obtenido a partir del compendio de los Eventos Sonoros No Deseados detectados, superando casi en 0.6 dB la grabación original. Se puede determinar pues que la eliminación de dichos eventos sonoros es esencial para una precisa determinación del Nivel de Presión Sonora asociado a una grabación de ruido por tráfico rodado.

Capítulo 6. Conclusiones

Dados los resultados expuestos en el capítulo anterior, se determinan los siguientes puntos clave:

- Las categorías escogidas, aun siendo únicamente tres, demostraron ser suficientemente distintivas entre sí para su clasificación, en la etapa de desarrollo de los dataset específicos.
- El proceso de toma y clasificación de muestras implementado produjo datasets apropiados, permitiendo al algoritmo desarrollado extraer características como la duración y los MFCC, entre las 800 muestras de cada categoría, mejorando su precisión al identificar cada una, y mejorando su exactitud para diferenciar entre las mismas.
- Una exactitud del 87% indica que el algoritmo desarrollado es competente en la distinción entre las categorías determinadas.
- Para cada categoría de Evento Sonoro, se obtuvo una precisión del 83% o más, demostrando que el algoritmo desarrollado logró un muy buen entendimiento de cada categoría mediante la cual fue entrenada, implicando una correcta extracción de características.
- Teniendo en cuenta la corrección de +4 dB necesaria hecha para compensar por un error en la generación de los datos de prueba, la fórmula implementada en código para obtener el Nivel Sonoro Equivalente de los segmentos de audio es apropiada, con error promedio de menos de 0.03 dB.

- Los Eventos Sonoros No Deseados presentan un aporte no insignificante al cálculo de Nivel Equivalente de una grabación digital de ruido por tráfico rodado en condiciones favorables, en horario diurno. En los casos estudiados, se evidencian diferencias de 0.03 dB y 0.1 dB entre el Nivel Equivalente de la grabación original, y aquella en la cual los Eventos Sonoros No Deseados fueron removidos. Dichas cantidades, aunque pequeñas, dan indicio del sesgo que causa la presencia de ESNDs en las grabaciones de ruido por tráfico rodado.

RECOMENDACIONES

- En primera instancia, se recomienda la creación de datasets para más Eventos Sonoros No Deseados con el fin de entrenar el algoritmo para identificar más tipos de eventos sonoros encontrados en grabaciones, por ejemplo, alarmas de carro, sirenas o perifoneo, entre otros. Esto puede además aumentar la precisión en cada categoría, pues permitiría al algoritmo aprender una mejor distinción entre Eventos Sonoros.
- Se recomienda adaptar el sistema de detección para funcionar, si no en tiempo real, con un promediado temporal reciente, para aplicaciones de monitoreo constante.
- Las métricas de evaluación mejoraron constantemente a lo largo de la inclusión de más muestras en los dataset. Se recomienda pues aumentar el tamaño de los existentes y acorde a la primera recomendación, crear nuevos datasets también de mayor tamaño.
- Si se lleva a cabo la realización de datasets correspondientes a Eventos Sonoros No Deseados externos a aquellos analizados en el presente trabajo, tener en cuenta la realización de un nuevo dataset de “otros”, pues en su estado actual, dicho dataset incluye gran variedad de eventos sonoros, incluidos los sugeridos en la primera recomendación.

Capítulo 7. Referencias

AEC - Contaminación acústica. ASOCIACIÓN ESPAÑOLA PARA LA CALIDAD. (2013).

Retrieved 21 June 2022, from <https://www.aec.es/web/guest/centro-conocimiento/contaminacion-acustica>.

Acosta, Ó., Montenegro, C., & Gaona, P. (2020). Condiciones de tránsito vehicular y uso de un modelo para la predicción de ruido por tráfico rodado en un entorno local de la ciudad de Bogotá-Colombia. *Revista Ibérica De Sistemas E Tecnologías De Informação*, E27, 605-614. Retrieved 23 May 2021, from.

Zambon, G., Benocci, R., Bisceglie, A., Roman, H., & Bellucci, P. (2017). The LIFE DYNAMAP project: Towards a procedure for dynamic noise mapping in urban areas. *Applied Acoustics*, 124, 52-60. doi: 10.1016/j.apacoust.2016.10.022

Alsina-Pagès, R. M., Alías, F., Socoró, J. C., & Orga, F. (2018). Detection of anomalous noise events on low-capacity acoustic nodes for dynamic road traffic noise mapping within an hybrid WASN. *Sensors (Switzerland)*, 18(4). <https://doi.org/10.3390/s18041272>

Alías, F., Alsina-Pagès, R. M., Orga, F., & Socoró, J. C. (2018). Detection of anomalous noise events for real-time road-traffic noise mapping: The DYNAMAP's project case study. *Noise Mapping*, 5(1), 71–85. <https://doi.org/10.1515/noise-2018-0006>

Sandoval, A. M. (2005). Ruido por tráfico urbano: conceptos, medidas descriptivas y valoración económica. *Revista de Economía y Administración Universidad Autónoma de Occidente*.

- Su, T. W., Liu, J. Y., & Yang, Y. H. (2017). Weakly-supervised audio event detection using event-specific Gaussian filters and fully convolutional networks. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 791–795. <https://doi.org/10.1109/ICASSP.2017.7952264>
- Brown, A., & De Coensel, B. (2018). A study of the performance of a generalized exceedance algorithm for detecting noise events caused by road traffic. *Applied Acoustics*, 138, 101–114. <https://doi.org/10.1016/j.apacoust.2018.03.031>
- The Sonic Environment of Cities. (1969). *Environment and Behavior*, 1(1), 49–70. <https://doi.org/10.1177/001391656900100104>
- European Environment Agency. (2014). *Noise in Europe 2014* (p. 19). Luxembourg: Publications Office of the European Union.
- European Automobile Manufacturers Association. (2022). Average age of the EU motor vehicle fleet, by vehicle type. Retrieved from <https://www.acea.auto/figure/average-age-of-eu-motor-vehicle-fleet-by-vehicle-type/>
- Pérez, Ó. (2022). ¿Cuáles son los vehículos con mayor edad del parque automotor colombiano? *El Espectador*. Retrieved 3 July 2022, from <https://www.elespectador.com/autos/cuales-son-los-vehiculos-con-mayor-edad-en-el-parque-automotor-colombiano/>.
- Ortega, C., & Sepúlveda, R. (2019). El 80 % del ruido en Bogotá lo producen los automotores. *El Tiempo*. Retrieved 21 June 2022, from <https://www.eltiempo.com/bogota/cuanto-ruido-se-genera-y-que-lo-produce-en-bogota-327728>.

- Ipsos Napoleón Franco. (2013). Medio ambiente y gestión del riesgo (pp. 59-66). Medellín: Medellín Cómo Vamos. Retrieved from https://www.medellincomovamos.org/sites/default/files/2020-01/documentos/1461076335wpdm_Informe%20de%20percepción%20ciudadana%20-%20Medio%20ambiente%20y%20gestión%20del%20riesgo%2C%202013.pdf
- Stowell, D., Giannoulis, D., Benetos, E., Lagrange, M., & Plumbley, M. D. (2015). Detection and Classification of Acoustic Scenes and Events. *IEEE Transactions on Multimedia*, 17(10), 1733–1746. <https://doi.org/10.1109/TMM.2015.2428998>
- Mioduszewski, P., Ejsmont, J. A., Grabowski, J., & Karpiński, D. (2011). Noise map validation by continuous noise monitoring. *Applied Acoustics*, 72(8), 582–589. <https://doi.org/10.1016/j.apacoust.2011.01.012>
- Hermida Cadena, L. F. (2019). *Desarrollo de un modelo de evaluación de paisajes sonoros según aspectos espaciales, temporales, subjetivo y de contexto*.
- Corporación Autónoma Regional de Cundinamarca CAR. (2007). *Mapa de Ruido Municipio de Girardot Cundinamarca*. 107.
- Cities, S., & Ambiental, R. (2020). *Diseño e implementación de una plataforma digital bajo el concepto de Smart Cities para visualizar el impacto por ruido en los entornos locales de cinco universidades de la ciudad de Bogotá Iván Felipe Pérez Naranjo*. 1–139.
- Secretaría de Ambiente de Bogotá (SDA). (2022). Secretaría de Ambiente de Bogotá. Obtenido de Secretaría de Ambiente de Bogotá: <https://ambientebogota.gov.co/ruido>

- Singh Rana, A., Singh, H., & Arora, H. (2019). Comparative Analysis of Classifiers on Audio, Image, and Text Datasets. *Université De Montréal*.
- Ranschaert, E., Morozov, S., & Algra, P. (2019). Artificial intelligence in medical imaging (pp. 347-373). Springer.
- Gupta, V. (2017). Understanding Feedforward Neural Networks [Image]. <https://learnopencv.com/wp-content/uploads/2017/10/mlp-diagram.jpg>.
- Barrios, J. (2019). *MATRIZ DE CONFUSIÓN BINARIA* [Image]. From <https://www.juanbarrios.com/wp-content/uploads/2019/07/MATRIZ-CONFUSION-400x358.png>.
- Sahoo, K., Dutta, I., Ijaz, M., Wozniak, M., & Singh, P. (2021). TLEFuzzyNet: Fuzzy Rank-Based Ensemble of Transfer Learning Models for Emotion Recognition From Human Speeches. *IEEE Access*, 9, 166518-166530. <https://doi.org/10.1109/access.2021.3135658>
- Imoto, K. (2018). Introduction to acoustic event and scene analysis. *Acoustical Science and Technology*, 39(3), 182–188. <https://doi.org/10.1250/ast.39.182>
- Boashash, B. (2016). Time-Frequency Signal Analysis and Processing (2nd ed.).
- Hibbert, D. (2007). Systematic errors in analytical measurement results. *Journal Of Chromatography A*, 1158(1-2), 25-32. <https://doi.org/10.1016/j.chroma.2007.03.021>
- International Organization for Standardization. (2006). *Statistics — Vocabulary and symbols — Part 2: Applied statistics* (ISO 3534-2:2006).

Stone, M. (1974). Cross-Validatory Choice and Assessment of Statistical Predictions. Journal Of The Royal Statistical Society: Series B (Methodological), 36(2), 111-133.
<https://doi.org/10.1111/j.2517-6161.1974.tb00994.x>

Zoom Corp. Zoom H1n [Image]. <https://zoomcorp.com/es/es/grabadoras-de-mano/handheld-recorders/h1n-handly-recorder/>.

Reaper. Reaper [Image]. <https://www.reaper.fm/>.

miniDSP. UMIK-1 [Image].
<https://www.minidsp.com/images/stories/virtuemart/product/282A1844.jpg>.

Dr. Jordan Design. Svantek SV30A class 1 [Image].
https://winaudiomls.de/joomla/components/com_jshopping/files/img_products/full_35b9639b708a86e315daf40acf0b16d2.jpg.

Dixten. Стойка FALCON EYES ST-804B [Image].
<https://dixten.com/catalog/oborudovanie-dlya-foto-i-videosemki/stoyki-dlya-foto-i-videosemki/stoyka-falcon-eyes-st-804b/>.

SVANTEK. SvanPC++ Software [Image]. https://p2k5a5w7.stackpathcdn.com/wp-content/uploads/2020/09/svanpc_icon_500x340.jpg.

Apkpure. Handy Recorder (Beta) [Image].
<https://image.winudf.com/v2/image1/anAuY28uem9vbS5oYW5keXJlY29yZGVyX3NjcmVlbl8wXzE2MDY2MDE4ODJfMDA2/screen-0.jpg?fakeurl=1&type=.jpg>.

Memorycow. Apeman A80 [Image].

<https://www.memorycow.co.uk/image/cache/data/wizard/external/apeman-a80-action-camera-250x250.jpg>.

Davis Instruments. Vantage Vue Weather Station [Image].

https://cdn.shopify.com/s/files/1/0515/5992/3873/products/06250_11_3cf646b0-0196-4239-9d43-9098df9b6972_750x518.jpg?v=1618465704.

IDECA. (2015). Vista satelital híbrida, Centro Comercial Hayuelos y alrededores. [Image].

<https://mapas.bogota.gov.co/#>.

Flaticon. Placeholder free icon [Image].

<https://cdn-icons-png.flaticon.com/512/684/684908.png>.

ANEXO I: TABLA DE DATOS

Inicio	Final	Sonido	dB SPL (Algoritmo)	dB SPL (dBFA Suite)	Error
0:00:00	0:00:01	aire	82,4	82,4	0
0:00:01	0:00:02	otro	83,3	83,4	0,1
0:00:02	0:00:03	claxon	82,5	82,6	0,1
0:00:03	0:00:04	otro	80,2	80,2	0
0:00:04	0:00:05	freno	80,1	80,1	0
0:00:05	0:00:06	freno	80,3	80,4	0,1
0:00:06	0:00:07	otro	78,7	78,6	-0,1
0:00:07	0:00:08	otro	78,3	78,5	0,2
0:00:08	0:00:09	otro	79,5	79,5	0
0:00:09	0:00:10	claxon	78,7	78,8	0,1
0:00:10	0:00:11	otro	79,7	79,3	-0,4
0:00:11	0:00:12	otro	79,1	79,7	0,6
0:00:12	0:00:13	freno	79,2	79,1	-0,1
0:00:13	0:00:14	otro	79,5	79,5	0
0:00:14	0:00:15	otro	78,5	78,6	0,1
0:00:15	0:00:16	otro	79	79,1	0,1
0:00:16	0:00:17	otro	79,3	79,4	0,1
0:00:17	0:00:18	otro	78	78	0
0:00:18	0:00:19	freno	77,6	77,6	0
0:00:19	0:00:20	freno	77,7	78	0,3
0:00:20	0:00:21	freno	78,7	78,7	0
0:00:21	0:00:22	freno	77,8	78	0,2
0:00:22	0:00:23	freno	78,4	78,3	-0,1
0:00:23	0:00:24	otro	77,8	77,5	-0,3
0:00:24	0:00:25	freno	78,8	78,8	0
0:00:25	0:00:26	freno	77,7	77,9	0,2
0:00:26	0:00:27	freno	77,9	77,9	0
0:00:27	0:00:28	freno	79	78,9	-0,1
0:00:28	0:00:29	freno	78,3	78,5	0,2
0:00:29	0:00:30	freno	79,4	79,5	0,1
0:00:30	0:00:31	freno	78,7	78,7	0
0:00:31	0:00:32	freno	78,4	78,7	0,3
0:00:32	0:00:33	otro	78,6	78,1	-0,5
0:00:33	0:00:34	freno	78,7	78,9	0,2
0:00:34	0:00:35	otro	79	79,2	0,2
0:00:35	0:00:36	otro	78,7	79	0,3
0:00:36	0:00:37	aire	79,1	78,7	-0,4
0:00:37	0:00:38	freno	78,9	78,6	-0,3
0:00:38	0:00:39	otro	79,4	79,3	-0,1
0:00:39	0:00:40	otro	78,2	79,2	1
0:00:40	0:00:41	freno	79,4	78,4	-1
0:00:41	0:00:42	freno	80,3	80,4	0,1
0:00:42	0:00:43	otro	80,8	80,8	0
0:00:43	0:00:44	freno	81	81	0
0:00:44	0:00:45	otro	81,1	80,8	-0,3
0:00:45	0:00:46	claxon	82,2	81,8	-0,4

0:00:46	0:00:47	aire	83,2	83,3	0,1
0:00:47	0:00:48	freno	81,8	81,6	-0,2
0:00:48	0:00:49	otro	84,3	84,4	0,1
0:00:49	0:00:50	freno	81,6	81,5	-0,1
0:00:50	0:00:51	freno	80,3	81,9	1,6
0:00:51	0:00:52	freno	80,1	80	-0,1
0:00:52	0:00:53	freno	78,9	78,9	0
0:00:53	0:00:54	claxon	80,5	79,8	-0,7
0:00:54	0:00:55	otro	79	79,6	0,6
0:00:55	0:00:56	otro	78,6	79,5	0,9
0:00:56	0:00:57	otro	79,9	79	-0,9
0:00:57	0:00:58	claxon	79,4	79,8	0,4
0:00:58	0:00:59	otro	79,7	79,8	0,1
0:00:59	0:01:00	otro	80,7	80	-0,7
0:01:00	0:01:01	otro	80,8	80,7	-0,1
0:01:01	0:01:02	otro	80,4	80,7	0,3
0:01:02	0:01:03	otro	80,9	80,9	0
0:01:03	0:01:04	otro	81,3	81,3	0
0:01:04	0:01:05	freno	80,3	80,4	0,1
0:01:05	0:01:06	freno	81	80,9	-0,1
0:01:06	0:01:07	otro	80,9	80,9	0
0:01:07	0:01:08	otro	81,8	81,6	-0,2
0:01:08	0:01:09	otro	80,5	81,1	0,6
0:01:09	0:01:10	otro	80,5	80,4	-0,1
0:01:10	0:01:11	otro	79,2	80,3	1,1
0:01:11	0:01:12	otro	79,6	79,1	-0,5
0:01:12	0:01:13	otro	80,2	79,5	-0,7
0:01:13	0:01:14	otro	79,5	80,3	0,8
0:01:14	0:01:15	otro	80,4	79,7	-0,7
0:01:15	0:01:16	otro	81,3	80,7	-0,6
0:01:16	0:01:17	otro	81,2	81,5	0,3
0:01:17	0:01:18	freno	80,3	81,1	0,8
0:01:18	0:01:19	otro	80,6	80,3	-0,3
0:01:19	0:01:20	otro	80,9	81,1	0,2
0:01:20	0:01:21	claxon	80,3	80,4	0,1
0:01:21	0:01:22	otro	79,4	79,7	0,3
0:01:22	0:01:23	otro	79,8	79,8	0
0:01:23	0:01:24	otro	79,7	79,9	0,2
0:01:24	0:01:25	otro	81	80,3	-0,7
0:01:25	0:01:26	otro	80,7	80,9	0,2
0:01:26	0:01:27	freno	81,3	80,8	-0,5
0:01:27	0:01:28	aire	80,6	81,2	0,6
0:01:28	0:01:29	aire	82,7	81,2	-1,5
0:01:29	0:01:30	aire	82,5	82,9	0,4
0:01:30	0:01:31	otro	82,2	82,2	0
0:01:31	0:01:32	otro	81,9	81,9	0
0:01:32	0:01:33	otro	81,9	82	0,1
0:01:33	0:01:34	otro	81,9	82,2	0,3
0:01:34	0:01:35	otro	82,6	82,1	-0,5
0:01:35	0:01:36	otro	82,5	82,4	-0,1

0:01:36	0:01:37	otro	85,5	82,8	-2,7
0:01:37	0:01:38	otro	85,2	85,9	0,7
0:01:38	0:01:39	aire	84,1	84,9	0,8
0:01:39	0:01:40	otro	83,5	83,9	0,4
0:01:40	0:01:41	freno	82,7	83,7	1
0:01:41	0:01:42	otro	84,4	83,2	-1,2
0:01:42	0:01:43	otro	83,9	84,6	0,7
0:01:43	0:01:44	otro	83	83,6	0,6
0:01:44	0:01:45	otro	82,3	82,3	0
0:01:45	0:01:46	otro	82,2	82,5	0,3
0:01:46	0:01:47	otro	81,5	82,1	0,6
0:01:47	0:01:48	otro	83,6	81,7	-1,9
0:01:48	0:01:49	otro	81,5	83,5	2
0:01:49	0:01:50	otro	81,9	81,3	-0,6
0:01:50	0:01:51	aire	82,2	82,4	0,2
0:01:51	0:01:52	otro	82,7	82	-0,7
0:01:52	0:01:53	aire	82,3	83	0,7
0:01:53	0:01:54	otro	85,1	82,6	-2,5
0:01:54	0:01:55	otro	84,9	85,3	0,4
0:01:55	0:01:56	otro	84,6	84,8	0,2
0:01:56	0:01:57	claxon	83,2	84,6	1,4
0:01:57	0:01:58	otro	83	83,1	0,1
0:01:58	0:01:59	aire	83,4	83	-0,4
0:01:59	0:02:00	otro	83,5	83,7	0,2
0:02:00	0:02:01	otro	83,7	83,1	-0,6
0:02:01	0:02:02	claxon	82	83,9	1,9
0:02:02	0:02:03	freno	82	81,9	-0,1
0:02:03	0:02:04	otro	82,8	82,1	-0,7
0:02:04	0:02:05	otro	83,6	82,9	-0,7
0:02:05	0:02:06	otro	79,9	83,6	3,7
0:02:06	0:02:07	otro	80,3	79,7	-0,6
0:02:07	0:02:08	otro	82,9	80,4	-2,5
0:02:08	0:02:09	otro	82,5	82,9	0,4
0:02:09	0:02:10	otro	82,3	82,8	0,5
0:02:10	0:02:11	otro	81,7	82	0,3
0:02:11	0:02:12	otro	83,7	81,9	-1,8
0:02:12	0:02:13	otro	82	83,8	1,8
0:02:13	0:02:14	aire	83,6	81,8	-1,8
0:02:14	0:02:15	otro	83,8	83,8	0
0:02:15	0:02:16	freno	84,1	83,7	-0,4
0:02:16	0:02:17	freno	81,5	84,2	2,7
0:02:17	0:02:18	freno	81,9	81,5	-0,4
0:02:18	0:02:19	otro	82,2	81,9	-0,3
0:02:19	0:02:20	otro	82,8	82,3	-0,5
0:02:20	0:02:21	otro	82,2	82,9	0,7
0:02:21	0:02:22	otro	81,8	82,3	0,5
0:02:22	0:02:23	otro	80,8	81,8	1
0:02:23	0:02:24	otro	80,6	80,9	0,3
0:02:24	0:02:25	otro	79,5	80,7	1,2
0:02:25	0:02:26	otro	79,9	79,6	-0,3

0:02:26	0:02:27	otro	79,4	80	0,6
0:02:27	0:02:28	otro	79,8	79,4	-0,4
0:02:28	0:02:29	otro	79,1	79,6	0,5
0:02:29	0:02:30	otro	79	79,4	0,4
0:02:30	0:02:31	otro	77,9	79	1,1
0:02:31	0:02:32	otro	79,4	78	-1,4
0:02:32	0:02:33	otro	80,2	79,5	-0,7
0:02:33	0:02:34	otro	80,3	79,9	-0,4
0:02:34	0:02:35	otro	78,4	80,6	2,2
0:02:35	0:02:36	otro	79,5	78,5	-1
0:02:36	0:02:37	otro	78,9	79,4	0,5
0:02:37	0:02:38	otro	79,9	78,9	-1
0:02:38	0:02:39	otro	80,4	79,6	-0,8
0:02:39	0:02:40	otro	80,1	80,6	0,5
0:02:40	0:02:41	otro	80	80,2	0,2
0:02:41	0:02:42	otro	80,5	79,9	-0,6
0:02:42	0:02:43	aire	79,5	80,7	1,2
0:02:43	0:02:44	otro	83	79,3	-3,7
0:02:44	0:02:45	otro	85,5	82,4	-3,1
0:02:45	0:02:46	otro	84,7	85,1	0,4
0:02:46	0:02:47	otro	84,6	85	0,4
0:02:47	0:02:48	otro	82,5	84,9	2,4
0:02:48	0:02:49	otro	79,5	82,9	3,4
0:02:49	0:02:50	otro	81,4	79,8	-1,6
0:02:50	0:02:51	otro	80,4	81,1	0,7
0:02:51	0:02:52	otro	81	80,3	-0,7
0:02:52	0:02:53	otro	81,7	81,2	-0,5
0:02:53	0:02:54	otro	80,5	82	1,5
0:02:54	0:02:55	otro	80,7	80,4	-0,3
0:02:55	0:02:56	otro	81,5	80,7	-0,8
0:02:56	0:02:57	otro	80,7	80,8	0,1
0:02:57	0:02:58	otro	81,1	81,4	0,3
0:02:58	0:02:59	otro	80,1	81,2	1,1
0:02:59	0:03:00	otro	79,4	80,5	1,1
0:03:00	0:03:01	otro	80,9	79,7	-1,2
0:03:01	0:03:02	otro	80	80,3	0,3
0:03:02	0:03:03	otro	80,8	80,4	-0,4
0:03:03	0:03:04	otro	80,9	80,7	-0,2
0:03:04	0:03:05	otro	80,9	80,8	-0,1
0:03:05	0:03:06	otro	80,6	80,8	0,2
0:03:06	0:03:07	freno	80,4	81,2	0,8
0:03:07	0:03:08	otro	80,6	80,5	-0,1
0:03:08	0:03:09	otro	81,9	80,7	-1,2
0:03:09	0:03:10	aire	80,5	81	0,5
0:03:10	0:03:11	otro	81,3	81,1	-0,2
0:03:11	0:03:12	otro	80,8	81,5	0,7
0:03:12	0:03:13	otro	80,9	80,4	-0,5
0:03:13	0:03:14	otro	79,5	81,2	1,7
0:03:14	0:03:15	otro	82,2	80,2	-2
0:03:15	0:03:16	otro	81,7	81	-0,7

0:03:16	0:03:17	otro	81,2	82,1	0,9
0:03:17	0:03:18	otro	81,8	81,1	-0,7
0:03:18	0:03:19	otro	81,5	81,8	0,3
0:03:19	0:03:20	otro	81,3	81,7	0,4
0:03:20	0:03:21	otro	81,5	81,5	0
0:03:21	0:03:22	otro	80,8	81,3	0,5
0:03:22	0:03:23	otro	80,9	81,8	0,9
0:03:23	0:03:24	otro	80,6	80,4	-0,2
0:03:24	0:03:25	otro	80,5	80,6	0,1
0:03:25	0:03:26	otro	81,8	80,4	-1,4
0:03:26	0:03:27	otro	81,7	81,5	-0,2
0:03:27	0:03:28	otro	81	81,8	0,8
0:03:28	0:03:29	otro	82,4	81,1	-1,3
0:03:29	0:03:30	otro	84,3	82,1	-2,2
0:03:30	0:03:31	otro	81,7	83,2	1,5
0:03:31	0:03:32	otro	80,4	83,7	3,3
0:03:32	0:03:33	otro	80,5	80,9	0,4
0:03:33	0:03:34	otro	78,3	80,2	1,9
0:03:34	0:03:35	aire	81,4	80	-1,4
0:03:35	0:03:36	otro	80,8	79,2	-1,6
0:03:36	0:03:37	otro	79,2	82,1	2,9
0:03:37	0:03:38	otro	82,5	79,6	-2,9
0:03:38	0:03:39	otro	84	80,7	-3,3
0:03:39	0:03:40	otro	80,8	83,7	2,9
0:03:40	0:03:41	otro	81,1	82,2	1,1
0:03:41	0:03:42	otro	79,1	81,9	2,8
0:03:42	0:03:43	otro	79,5	80	0,5
0:03:43	0:03:44	otro	80,3	79,1	-1,2
0:03:44	0:03:45	otro	78,3	80	1,7
0:03:45	0:03:46	otro	79,1	80	0,9
0:03:46	0:03:47	otro	78,7	78,4	-0,3
0:03:47	0:03:48	otro	77,4	79,1	1,7
0:03:48	0:03:49	otro	77,6	77,7	0,1
0:03:49	0:03:50	claxon	78,9	77,8	-1,1
0:03:50	0:03:51	otro	79	78	-1
0:03:51	0:03:52	freno	78,8	78,8	0
0:03:52	0:03:53	otro	80,8	79,1	-1,7
0:03:53	0:03:54	otro	79,9	79,4	-0,5
0:03:54	0:03:55	otro	79,7	81	1,3
0:03:55	0:03:56	otro	79,4	79,8	0,4
0:03:56	0:03:57	otro	82,6	79,4	-3,2
0:03:57	0:03:58	otro	82,5	80,7	-1,8
0:03:58	0:03:59	otro	80,5	82,5	2
0:03:59	0:04:00	otro	80,6	82,2	1,6
0:04:00	0:04:01	otro	79,8	80,3	0,5
0:04:01	0:04:02	otro	81,3	80,5	-0,8
0:04:02	0:04:03	otro	80,8	80,1	-0,7
0:04:03	0:04:04	freno	80,3	81,3	1
0:04:04	0:04:05	otro	81,9	81,1	-0,8
0:04:05	0:04:06	otro	81,9	80,2	-1,7

0:04:06	0:04:07	otro	82,2	82,1	-0,1
0:04:07	0:04:08	otro	83,2	81,6	-1,6
0:04:08	0:04:09	freno	81,3	82,7	1,4
0:04:09	0:04:10	otro	80,8	83,1	2,3
0:04:10	0:04:11	otro	81,3	81	-0,3
0:04:11	0:04:12	otro	81,1	80,8	-0,3
0:04:12	0:04:13	otro	81,9	81,6	-0,3
0:04:13	0:04:14	otro	80,2	81,2	1
0:04:14	0:04:15	otro	80,3	81,7	1,4
0:04:15	0:04:16	otro	80,1	80	-0,1
0:04:16	0:04:17	otro	79,9	80,4	0,5
0:04:17	0:04:18	otro	80,6	80	-0,6
0:04:18	0:04:19	aire	79,8	80	0,2
0:04:19	0:04:20	otro	80,1	80,7	0,6
0:04:20	0:04:21	otro	79,9	80	0,1
0:04:21	0:04:22	aire	80,3	80,2	-0,1
0:04:22	0:04:23	otro	79,8	79,6	-0,2
0:04:23	0:04:24	otro	79,7	80,4	0,7
0:04:24	0:04:25	otro	80,2	79,9	-0,3
0:04:25	0:04:26	freno	78,5	79,8	1,3
0:04:26	0:04:27	freno	78,7	80,2	1,5
0:04:27	0:04:28	otro	78,5	78,4	-0,1
0:04:28	0:04:29	otro	78,3	78,7	0,4
0:04:29	0:04:30	freno	79,3	78,5	-0,8
0:04:30	0:04:31	freno	79,8	78,5	-1,3
0:04:31	0:04:32	aire	81,1	79,3	-1,8
0:04:32	0:04:33	otro	80,1	79,8	-0,3
0:04:33	0:04:34	otro	80,1	81,2	1,1
0:04:34	0:04:35	otro	80,4	80	-0,4
0:04:35	0:04:36	aire	81,4	80,1	-1,3
0:04:36	0:04:37	otro	81,6	80,4	-1,2
0:04:37	0:04:38	claxon	82,7	81,4	-1,3
0:04:38	0:04:39	claxon	83,9	81,6	-2,3
0:04:39	0:04:40	claxon	85	82,7	-2,3
0:04:40	0:04:41	claxon	86,4	83,9	-2,5
0:04:41	0:04:42	claxon	87	85,1	-1,9
0:04:42	0:04:43	freno	85,5	86,3	0,8
0:04:43	0:04:44	freno	82,5	87	4,5
0:04:44	0:04:45	freno	82,1	85,7	3,6
0:04:45	0:04:46	otro	80,5	82,6	2,1
0:04:46	0:04:47	claxon	82,9	82,3	-0,6
0:04:47	0:04:48	otro	83,5	80,5	-3
0:04:48	0:04:49	otro	82,8	82,7	-0,1
0:04:49	0:04:50	otro	82,6	83,6	1
0:04:50	0:04:51	aire	82,9	82,6	-0,3
0:04:51	0:04:52	otro	82,2	82,6	0,4
0:04:52	0:04:53	otro	80,7	83	2,3
0:04:53	0:04:54	otro	80,8	82,4	1,6
0:04:54	0:04:55	otro	79,9	81	1,1
0:04:55	0:04:56	otro	82,4	80,9	-1,5

0:04:56	0:04:57	otro	82	79,7	-2,3
0:04:57	0:04:58	otro	83,1	82,2	-0,9
0:04:58	0:04:59	otro	81,8	82	0,2
0:04:59	0:05:00	otro	79,7	83	3,3
0:05:00	0:05:01	otro	81	82,4	1,4
0:05:01	0:05:02	otro	80	79,8	-0,2
0:05:02	0:05:03	otro	81,4	80,7	-0,7
0:05:03	0:05:04	otro	80,9	80,3	-0,6
0:05:04	0:05:05	otro	81,4	81,1	-0,3
0:05:05	0:05:06	otro	82,4	81,3	-1,1
0:05:06	0:05:07	otro	80,7	81	0,3
0:05:07	0:05:08	otro	79,4	82,6	3,2
0:05:08	0:05:09	otro	78,7	81,1	2,4
0:05:09	0:05:10	otro	79,7	79,1	-0,6
0:05:10	0:05:11	otro	79,9	79,2	-0,7
0:05:11	0:05:12	otro	81,4	79,7	-1,7
0:05:12	0:05:13	otro	81,6	79,9	-1,7
0:05:13	0:05:14	otro	80,3	80,7	0,4
0:05:14	0:05:15	otro	81,3	82,1	0,8
0:05:15	0:05:16	otro	83,2	80,2	-3
0:05:16	0:05:17	otro	81,2	80,7	-0,5
0:05:17	0:05:18	otro	81,1	82,9	1,8
0:05:18	0:05:19	otro	81,1	82	0,9
0:05:19	0:05:20	otro	82,4	81,5	-0,9
0:05:20	0:05:21	otro	82,2	80,4	-1,8
0:05:21	0:05:22	otro	81,2	82,3	1,1
0:05:22	0:05:23	otro	80	82,4	2,4
0:05:23	0:05:24	otro	81,6	81,5	-0,1
0:05:24	0:05:25	otro	80,2	80,7	0,5
0:05:25	0:05:26	otro	80	81	1
0:05:26	0:05:27	otro	80,8	80,6	-0,2
0:05:27	0:05:28	otro	80,9	80,5	-0,4
0:05:28	0:05:29	otro	81	80,3	-0,7
0:05:29	0:05:30	otro	82	80,9	-1,1
0:05:30	0:05:31	otro	80,1	81,1	1
0:05:31	0:05:32	otro	81,3	81,4	0,1
0:05:32	0:05:33	otro	81,3	81,2	-0,1
0:05:33	0:05:34	otro	81,1	80,8	-0,3
0:05:34	0:05:35	freno	80,8	81,6	0,8
0:05:35	0:05:36	otro	82,1	80,6	-1,5
0:05:36	0:05:37	freno	82,4	81	-1,4
0:05:37	0:05:38	freno	80,7	82,1	1,4
0:05:38	0:05:39	otro	81,4	82,6	1,2
0:05:39	0:05:40	freno	80,9	80,8	-0,1
0:05:40	0:05:41	otro	80,4	81,3	0,9
0:05:41	0:05:42	otro	79,5	81,2	1,7
0:05:42	0:05:43	otro	81,4	80,5	-0,9
0:05:43	0:05:44	otro	80,1	80,1	0
0:05:44	0:05:45	otro	81,1	80,5	-0,6
0:05:45	0:05:46	freno	80,6	81	0,4

0:05:46	0:05:47	otro	81,1	80,3	-0,8
0:05:47	0:05:48	freno	80,9	81,1	0,2
0:05:48	0:05:49	freno	80,1	81,2	1,1
0:05:49	0:05:50	otro	80,9	80,9	0
0:05:50	0:05:51	otro	79,7	80,8	1,1
0:05:51	0:05:52	aire	81,2	79,8	-1,4
0:05:52	0:05:53	otro	80,6	80,8	0,2
0:05:53	0:05:54	otro	82,9	80	-2,9
0:05:54	0:05:55	otro	82,7	81,5	-1,2
0:05:55	0:05:56	otro	82,5	81,8	-0,7
0:05:56	0:05:57	otro	82,3	82,3	0
0:05:57	0:05:58	otro	80,6	83,2	2,6
0:05:58	0:05:59	otro	83,9	81,9	-2
0:05:59	0:06:00	otro	82,1	82	-0,1
0:06:00	0:06:01	freno	81,6	81,4	-0,2
0:06:01	0:06:02	otro	80,8	83,5	2,7
0:06:02	0:06:03	otro	80,1	82,1	2
0:06:03	0:06:04	otro	78,8	81,8	3
0:06:04	0:06:05	otro	79,2	80	0,8
0:06:05	0:06:06	otro	79,7	80,2	0,5
0:06:06	0:06:07	otro	79,5	78,6	-0,9
0:06:07	0:06:08	aire	82,9	79,2	-3,7
0:06:08	0:06:09	aire	81,2	80,6	-0,6
0:06:09	0:06:10	otro	80,4	79	-1,4
0:06:10	0:06:11	otro	80,8	83,7	2,9
0:06:11	0:06:12	otro	79,8	79,7	-0,1
0:06:12	0:06:13	otro	79,2	80,9	1,7
0:06:13	0:06:14	otro	77,8	80,5	2,7
0:06:14	0:06:15	otro	79	79,8	0,8
0:06:15	0:06:16	otro	79,5	78,7	-0,8
0:06:16	0:06:17	otro	80	78,2	-1,8
0:06:17	0:06:18	claxon	81,6	79,2	-2,4
0:06:18	0:06:19	otro	80	79,3	-0,7
0:06:19	0:06:20	otro	79,3	81,6	2,3
0:06:20	0:06:21	otro	79,5	80,1	0,6
0:06:21	0:06:22	otro	79,3	80,1	0,8
0:06:22	0:06:23	freno	78,8	79,4	0,6
0:06:23	0:06:24	freno	77,3	79,4	2,1
0:06:24	0:06:25	freno	77,6	79,3	1,7
0:06:25	0:06:26	otro	78,2	78,5	0,3
0:06:26	0:06:27	otro	77,4	77,3	-0,1
0:06:27	0:06:28	otro	78,4	77,7	-0,7
0:06:28	0:06:29	otro	79,2	78	-1,2
0:06:29	0:06:30	otro	78,8	77,8	-1
0:06:30	0:06:31	otro	78,2	79	0,8
0:06:31	0:06:32	freno	79,1	78,8	-0,3
0:06:32	0:06:33	freno	78,9	78,7	-0,2
0:06:33	0:06:34	freno	78,3	78,7	0,4
0:06:34	0:06:35	otro	79,8	78,9	-0,9
0:06:35	0:06:36	otro	80	78,8	-1,2

0:06:36	0:06:37	otro	81,1	78,5	-2,6
0:06:37	0:06:38	otro	80	79,9	-0,1
0:06:38	0:06:39	aire	79,4	80,1	0,7
0:06:39	0:06:40	otro	80,9	81,2	0,3
0:06:40	0:06:41	freno	80,4	80	-0,4
0:06:41	0:06:42	otro	80,3	79,5	-0,8
0:06:42	0:06:43	otro	80,4	80,9	0,5
0:06:43	0:06:44	otro	79,4	80,4	1
0:06:44	0:06:45	freno	79,6	80,2	0,6
0:06:45	0:06:46	otro	79,2	80,5	1,3
0:06:46	0:06:47	otro	79,3	79,4	0,1
0:06:47	0:06:48	otro	80,3	79,6	-0,7
0:06:48	0:06:49	otro	79,6	79,1	-0,5
0:06:49	0:06:50	otro	80,4	79,4	-1
0:06:50	0:06:51	otro	80,3	80,5	0,2
0:06:51	0:06:52	otro	79,9	79,5	-0,4
0:06:52	0:06:53	otro	79,8	80,4	0,6
0:06:53	0:06:54	otro	81	80,4	-0,6
0:06:54	0:06:55	otro	80	79,9	-0,1
0:06:55	0:06:56	freno	80,6	79,8	-0,8
0:06:56	0:06:57	freno	80,2	81,1	0,9
0:06:57	0:06:58	otro	80,4	80,1	-0,3
0:06:58	0:06:59	aire	81,6	80,5	-1,1
0:06:59	0:07:00	otro	81,4	80,3	-1,1
0:07:00	0:07:01	otro	79,6	80,5	0,9
0:07:01	0:07:02	otro	80,7	81,3	0,6
0:07:02	0:07:03	otro	80,5	81,7	1,2
0:07:03	0:07:04	otro	82,3	79,8	-2,5
0:07:04	0:07:05	otro	84,1	80,6	-3,5
0:07:05	0:07:06	otro	81,9	80,5	-1,4
0:07:06	0:07:07	otro	82,2	81,8	-0,4
0:07:07	0:07:08	otro	80	84,2	4,2
0:07:08	0:07:09	otro	80,3	82,5	2,2
0:07:09	0:07:10	freno	79,4	82,2	2,8
0:07:10	0:07:11	otro	79,4	80,2	0,8
0:07:11	0:07:12	otro	79,7	79,7	0
0:07:12	0:07:13	otro	79,9	80,1	0,2
0:07:13	0:07:14	otro	79,1	79,6	0,5
0:07:14	0:07:15	otro	80,2	79,4	-0,8
0:07:15	0:07:16	otro	80,3	80	-0,3
0:07:16	0:07:17	otro	79,7	79	-0,7
0:07:17	0:07:18	otro	79,6	80,4	0,8
0:07:18	0:07:19	aire	79,8	80,4	0,6
0:07:19	0:07:20	otro	81,4	79,6	-1,8
0:07:20	0:07:21	freno	79,8	79,7	-0,1
0:07:21	0:07:22	freno	79,6	79,9	0,3
0:07:22	0:07:23	otro	80,3	81,1	0,8
0:07:23	0:07:24	aire	79,5	80,2	0,7
0:07:24	0:07:25	otro	80,6	79,1	-1,5
0:07:25	0:07:26	otro	80,1	80,5	0,4

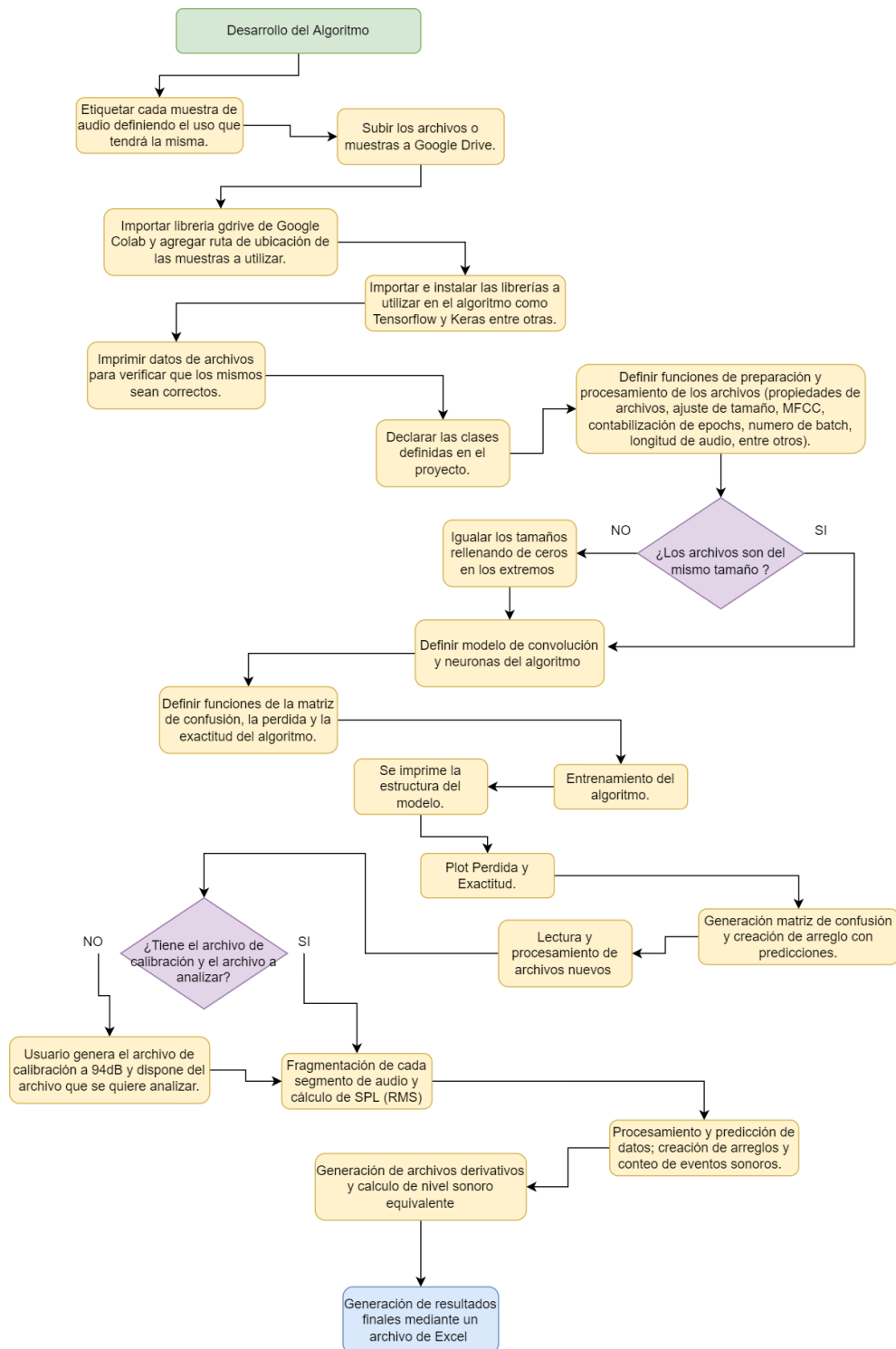
0:07:26	0:07:27	otro	80,2	80	-0,2
0:07:27	0:07:28	otro	80,6	80,2	-0,4
0:07:28	0:07:29	otro	80	80,3	0,3
0:07:29	0:07:30	claxon	79,8	80,3	0,5
0:07:30	0:07:31	otro	80	80,3	0,3
0:07:31	0:07:32	otro	80,7	80,6	-0,1
0:07:32	0:07:33	otro	81,4	79,5	-1,9
0:07:33	0:07:34	otro	81,9	80,1	-1,8
0:07:34	0:07:35	otro	81,8	80,4	-1,4
0:07:35	0:07:36	otro	80,3	81,3	1
0:07:36	0:07:37	otro	80,7	81,8	1,1
0:07:37	0:07:38	otro	81,1	81,7	0,6
0:07:38	0:07:39	otro	81,6	81,3	-0,3
0:07:39	0:07:40	otro	80,8	80,2	-0,6
0:07:40	0:07:41	otro	79,1	80,9	1,8
0:07:41	0:07:42	otro	78,2	81,9	3,7
0:07:42	0:07:43	otro	79,1	80,7	1,6
0:07:43	0:07:44	otro	78,8	79,9	1,1
0:07:44	0:07:45	otro	78,3	78,7	0,4
0:07:45	0:07:46	otro	78,1	78,4	0,3
0:07:46	0:07:47	otro	79,5	78,5	-1
0:07:47	0:07:48	freno	80,1	79,2	-0,9
0:07:48	0:07:49	otro	80,8	77,5	-3,3
0:07:49	0:07:50	otro	80,8	79,4	-1,4
0:07:50	0:07:51	otro	80	80,1	0,1
0:07:51	0:07:52	otro	80,2	80,3	0,1
0:07:52	0:07:53	otro	82	80,7	-1,3
0:07:53	0:07:54	otro	83,7	80,2	-3,5
0:07:54	0:07:55	freno	82,2	81	-1,2
0:07:55	0:07:56	otro	83,1	79,6	-3,5
0:07:56	0:07:57	claxon	83,4	83,8	0,4
0:07:57	0:07:58	otro	85	83,5	-1,5
0:07:58	0:07:59	otro	83,1	81,5	-1,6
0:07:59	0:08:00	freno	83,9	83,8	-0,1
0:08:00	0:08:01	freno	84,6	84,2	-0,4
0:08:01	0:08:02	otro	83,4	84,1	0,7
0:08:02	0:08:03	otro	83,6	82,9	-0,7
0:08:03	0:08:04	otro	82,7	84,8	2,1
0:08:04	0:08:05	otro	83,1	83,6	0,5
0:08:05	0:08:06	otro	81,2	84	2,8
0:08:06	0:08:07	otro	81,9	83,5	1,6
0:08:07	0:08:08	aire	83,3	82,8	-0,5
0:08:08	0:08:09	otro	82	82,5	0,5
0:08:09	0:08:10	otro	82	80,9	-1,1
0:08:10	0:08:11	otro	81,1	83	1,9
0:08:11	0:08:12	otro	81,1	82,6	1,5
0:08:12	0:08:13	otro	80,7	82,2	1,5
0:08:13	0:08:14	freno	80,8	81,2	0,4
0:08:14	0:08:15	aire	80,8	81,6	0,8
0:08:15	0:08:16	otro	82,2	80,8	-1,4

0:08:16	0:08:17	freno	80,1	80,6	0,5
0:08:17	0:08:18	freno	79,6	81	1,4
0:08:18	0:08:19	freno	80,8	81,2	0,4
0:08:19	0:08:20	otro	81	81,9	0,9
0:08:20	0:08:21	freno	79,3	79,3	0
0:08:21	0:08:22	aire	78,5	79,5	1
0:08:22	0:08:23	otro	78,9	81,4	2,5
0:08:23	0:08:24	freno	78,2	80,3	2,1
0:08:24	0:08:25	aire	78	79,3	1,3
0:08:25	0:08:26	freno	78,4	78,6	0,2
0:08:26	0:08:27	otro	79,2	79	-0,2
0:08:27	0:08:28	freno	79,5	78,3	-1,2
0:08:28	0:08:29	claxon	80,4	78	-2,4
0:08:29	0:08:30	otro	80,4	79,1	-1,3
0:08:30	0:08:31	otro	79,1	78,6	-0,5
0:08:31	0:08:32	otro	79,5	80,1	0,6
0:08:32	0:08:33	otro	78,5	80,3	1,8
0:08:33	0:08:34	freno	78,3	80,1	1,8
0:08:34	0:08:35	freno	81,8	79,1	-2,7
0:08:35	0:08:36	otro	80	79,6	-0,4
0:08:36	0:08:37	otro	80,8	78,5	-2,3
0:08:37	0:08:38	otro	80,2	78,9	-1,3
0:08:38	0:08:39	freno	80,4	81,6	1,2
0:08:39	0:08:40	freno	82,4	80,4	-2
0:08:40	0:08:41	otro	79,9	81	1,1
0:08:41	0:08:42	otro	79,4	80,2	0,8
0:08:42	0:08:43	otro	79,5	80,7	1,2
0:08:43	0:08:44	otro	80	82,5	2,5
0:08:44	0:08:45	claxon	81	79,1	-1,9
0:08:45	0:08:46	otro	80,8	79,8	-1
0:08:46	0:08:47	otro	80,2	79,5	-0,7
0:08:47	0:08:48	freno	79,5	79,6	0,1
0:08:48	0:08:49	freno	80,3	81,4	1,1
0:08:49	0:08:50	otro	80,5	80,6	0,1
0:08:50	0:08:51	otro	79,9	80,5	0,6
0:08:51	0:08:52	otro	79,7	79,4	-0,3
0:08:52	0:08:53	otro	78,6	80,6	2
0:08:53	0:08:54	otro	79,2	80,3	1,1
0:08:54	0:08:55	otro	79,6	79,9	0,3
0:08:55	0:08:56	aire	79,5	79,6	0,1
0:08:56	0:08:57	otro	79,3	79,1	-0,2
0:08:57	0:08:58	otro	79,4	78,6	-0,8
0:08:58	0:08:59	otro	79,2	80,2	1
0:08:59	0:09:00	freno	79,4	79,1	-0,3
0:09:00	0:09:01	freno	79,8	79,6	-0,2
0:09:01	0:09:02	freno	79,6	79,1	-0,5
0:09:02	0:09:03	freno	80,1	79,2	-0,9
0:09:03	0:09:04	freno	79,9	79,5	-0,4
0:09:04	0:09:05	otro	80,1	79,8	-0,3
0:09:05	0:09:06	otro	79,7	79,6	-0,1

0:09:06	0:09:07	otro	80,4	80,2	-0,2
0:09:07	0:09:08	otro	80,7	79,9	-0,8
0:09:08	0:09:09	otro	79,7	80,1	0,4
0:09:09	0:09:10	otro	81,5	79,8	-1,7
0:09:10	0:09:11	freno	80	80,4	0,4
0:09:11	0:09:12	freno	79,3	80,7	1,4
0:09:12	0:09:13	freno	81,5	79,7	-1,8
0:09:13	0:09:14	aire	77,7	81,5	3,8
0:09:14	0:09:15	aire	79,1	80	0,9
0:09:15	0:09:16	aire	80,2	79,6	-0,6
0:09:16	0:09:17	freno	80	81,5	1,5
0:09:17	0:09:18	freno	79,4	77,8	-1,6
0:09:18	0:09:19	otro	80,8	79,1	-1,7
0:09:19	0:09:20	otro	81,1	80	-1,1
0:09:20	0:09:21	otro	80,5	79,8	-0,7
0:09:21	0:09:22	freno	80,5	79,6	-0,9
0:09:22	0:09:23	otro	80,5	80,9	0,4
0:09:23	0:09:24	otro	80	81,2	1,2
0:09:24	0:09:25	freno	80,3	80,4	0,1
0:09:25	0:09:26	freno	79,7	80,7	1
0:09:26	0:09:27	otro	80,5	80,3	-0,2
0:09:27	0:09:28	otro	79,8	80,3	0,5
0:09:28	0:09:29	otro	80,9	79,8	-1,1
0:09:29	0:09:30	freno	81,5	80,1	-1,4
0:09:30	0:09:31	otro	82,1	80,5	-1,6
0:09:31	0:09:32	freno	81,9	79,8	-2,1
0:09:32	0:09:33	freno	81,7	80,8	-0,9
0:09:33	0:09:34	freno	80,8	81,4	0,6
0:09:34	0:09:35	freno	79,5	82	2,5
0:09:35	0:09:36	otro	81,3	82,1	0,8
0:09:36	0:09:37	otro	80,5	81,6	1,1
0:09:37	0:09:38	otro	81,1	81	-0,1
0:09:38	0:09:39	otro	80,4	79,5	-0,9
0:09:39	0:09:40	freno	80	80,8	0,8
0:09:40	0:09:41	freno	79,5	81,3	1,8
0:09:41	0:09:42	aire	82,2	80,8	-1,4
0:09:42	0:09:43	otro	81,9	80,6	-1,3
0:09:43	0:09:44	otro	80,9	80	-0,9
0:09:44	0:09:45	otro	82,3	79,6	-2,7
0:09:45	0:09:46	freno	81,6	81,8	0,2
0:09:46	0:09:47	freno	80,1	82,4	2,3
0:09:47	0:09:48	freno	81	80,7	-0,3
0:09:48	0:09:49	claxon	82,5	81,9	-0,6
0:09:49	0:09:50	otro	81,5	81,9	0,4
0:09:50	0:09:51	otro	80,8	80,9	0,1
0:09:51	0:09:52	freno	80,3	80,8	0,5
0:09:52	0:09:53	freno	78,7	82,4	3,7
0:09:53	0:09:54	freno	80,5	81,9	1,4
0:09:54	0:09:55	freno	81,5	80,7	-0,8
0:09:55	0:09:56	freno	80,4	80,4	0

0:09:56	0:09:57	otro	81,2	79,5	-1,7
0:09:57	0:09:58	otro	82,6	79,6	-3
0:09:58	0:09:59	otro	81,2	81,5	0,3
0:09:59	0:10:00	otro	81,9	80,8	-1,1
0:10:00	0:10:01	freno	81	80,9	-0,1

ANEXO II: DIAGRAMA DE FLUJO DEL ALGORITMO



ANEXO III: ARQUITECTURA DE LA RED NEURONAL

