

UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA  
DEPARTAMENTO DE ELECTRÓNICA  
VALPARAÍSO - CHILE



**" APLICACIÓN DE TÉCNICAS DE 'DATA AUGMENTATION'  
EN SEÑALES DE VOZ PARA LA DETECCIÓN DE PARKINSON  
UTILIZANDO REDES CONVOLUCIONALES ".**

**MATÍAS IGNACIO VALENZUELA PÉREZ**

MEMORIA PARA OPTAR AL TÍTULO DE  
INGENIERO CIVIL ELECTRÓNICO

PROFESOR GUÍA : MATÍAS ZAÑARTU  
PROFESOR CORREFERENTE : JUAN PABLO CORTÉS

Febrero 2024



*A mi familia . . .*

Por haber estado ahí desde el día 0.

Por inspirarme a cumplir mis metas.

Por demostrarme que siempre se puede salir adelante.

## AGRADECIMIENTOS

En primer lugar, agradezco al profesor Matías Zañartu por su guía y apoyo para encontrar y desarrollar este proyecto. Además, me siento muy agradecido de haber contado con la ayuda, el tiempo y los consejos de Emiro Ibarra en cada una de las etapas necesarias para llevar adelante este trabajo. Rescato enormemente el aprendizaje recibido de parte de ambos.

A mi familia, agradezco haberlos tenido a mi lado cada día en estos seis grandes e importantes años, por haberme ayudado a enfrentar los bajos y por celebrar junto a mi los altos.

A mis amigos, Mireya, Betsabé, Paula, Britney, Diego, Sebastián e Ignacio, quienes han hecho mi paso por la Universidad algo increíble, lleno de buenos recuerdos, viajes, grandes conversaciones y por sobretodo mucha, mucha comida. Sin duda, las risas no faltaron.

Me gustaría agradecer especialmente a Britney y Diego, quienes son las personas con las que más conviví a lo largo de estos años y los llevo siempre conmigo. Gracias por los proyectos que sacamos adelante, por enseñarme y permitirme enseñarles, por los cafecitos y por las largas noches de estudio que han hecho que esté escribiendo estas palabras.

Finalmente, agradecer a todas las personas que han sido parte de este camino, profesores, compañeros, familia y amigos, por haber aportado con distintos granitos de arena que han formado a este Ingeniero Civil Electrónico.

---

## RESUMEN EJECUTIVO

El uso de tecnologías que aplican la inteligencia artificial son cada vez más comunes en nuestro diario vivir y se ha trabajado con ellas en una gran variedad de campos de estudio, donde la medicina y las tecnologías para la salud no quedan ajenas a esta tendencia.

Gracias a la notable habilidad de las redes neuronales de encontrar patrones es que se ha planteado su uso en la detección de distintas patologías que afectan a los seres humanos en base a resultados de exámenes médicos como imágenes de resonancia magnética, electroencefalogramas o tomografías computarizadas. Dado que cada vez se busca el desarrollo de tecnologías menos invasivas y de menor costo para poder entregar soluciones clínicas confiables a una mayor cantidad de pacientes sobretodo en regiones de bajos recursos es que se ha desarrollado trabajos en el ámbito de la detección de enfermedades por medio de datos adquiridos de manera ambulatoria como por ejemplo grabaciones de voz.

Si bien el uso de grabaciones de la voz humana permite una toma de datos menos invasiva que exámenes más complejos como una resonancia magnética, esto no significa que los datos asociados a la voz abunden para realizar modelos capaces de lograr una detección efectiva de manera generalizada para cada patología. De manera específica, en este estudio se trabaja con la detección de la enfermedad de Parkinson a través de la voz y la falta de datos es uno de los grandes problemas que impide que esta tecnología pueda ser utilizada en ambientes no controlados y de manera masiva.

Para afrontar problemas similares, se ha desarrollado distintas técnicas que permiten realizar un aumento de la data utilizada para entrenar dichos modelos. Este estudio tiene por objetivo evaluar el efecto que cinco técnicas de aumento de data distintas tienen sobre un modelo de detección de la enfermedad de Parkinson a través de la voz.

Esta evaluación se realiza utilizando un modelo de redes neuronales convolucionales y la representación en forma de espectrogramas de Mel de las distintas señales de audio tanto originales como aumentadas. Luego de una serie de entrenamientos, se compara las distintas métricas reportadas por cada una de las técnicas utilizadas respecto a un modelo base entrenado sin data sintética.

Entre los principales resultados del estudio se encuentra que la variabilidad que entregue la técnica de aumento de data utilizada a los datos de entrenamiento tiene una directa relación con la capacidad de generalización que tendrá el modelo. Así, de las técnicas utilizadas se destaca 'Pitch Change' o cambio en el tono de voz, la cual logra generar datos variados que aumentan en aproximadamente un 10 % la exactitud de clasificación en validación cruzada.

Finalmente, se destaca que todas las técnicas utilizadas logran mejorar el comportamiento del modelo frente a una validación con la misma base de datos y una validación cruzada, aunque sea en un bajo porcentaje. Por tanto, se considera que el uso de aumento de data es beneficioso para el problema de la generalización pero son necesarios más estudios ya sea con otras formas de aumento de data para evaluar el desempeño o bien con distintas combinaciones de ellas e incluso otros enfoques que permitan entrenar modelos capaces de realizar una detección efectiva de la enfermedad de Parkinson.

**Palabras Clave.** Enfermedad de Parkinson, Aumento de data, Red neuronal convolucional, Espectrograma de Mel

---

## ABSTRACT

The use of artificial intelligence technologies is becoming more common in our daily lives, and they have been applied in a wide variety of fields of study, with medicine and health technologies being no exception to this trend.

Thanks to the remarkable ability of neural networks to find patterns, their use in detecting various human pathologies based on results from medical tests such as magnetic resonance imaging, electroencephalograms, or computed tomography has been proposed. As the development of less invasive and lower-cost technologies is increasingly sought to provide reliable clinical solutions to a greater number of patients, especially in regions with limited resources, work has been developed in the field of disease detection through ambulatory data acquisition, such as voice recordings.

While the use of human voice recordings allows for less invasive data collection than more complex tests such as magnetic resonance imaging, this does not mean that voice-associated data is abundant enough to build models capable of achieving effective detection in a generalized manner for each disease. Specifically, this study focuses on the detection of Parkinson's disease through voice, and the lack of data is one of the major problems preventing this technology from being used in uncontrolled and mass environments.

To address similar problems, various techniques have been developed to increase the amount of data used to train such models. This study aims to evaluate the effect that five different data augmentation techniques have on a Parkinson's disease detection model through voice.

This evaluation is performed using a convolutional neural network model and Mel spectrogram representations of the various audio signals, both original and augmented. After a series of trainings, the different metrics reported by each of the techniques used are compared to a base model trained without synthetic data.

Among the main results of the study, it is found that the variability provided by the data augmentation technique used on the training data is directly related to the model's generalization ability. Thus, among the techniques used, 'Pitch Change', or changes in voice tone, stands out for generating diverse data that increase classification accuracy in cross-validation by approximately 10 %.

Finally, it is noted that all the techniques used manage to improve the model's behavior in both validation with the same database and cross-validation, albeit to a small extent. Therefore, the use of data augmentation is considered beneficial for the problem of generalization, but further studies are necessary, either with other forms of data augmentation to assess performance or with different combinations of them, and even other approaches that allow training models capable of achieving effective Parkinson's disease detection.

**Keywords.** Parkinson's Disease, Data augmentation, Convolutional neural network, Mel-spectrogram|

---

## GLOSARIO

- **CNN:** Convolutional neural network (red neuronal convolucional).
- **GAN:** Generative adversarial networks o redes generativas antagónicas. Corresponden a una representación de dos redes neuronales que en conjunto son capaces de generar información nueva en base a datos de entrada.
- **VAE:** Variational Autoencoder o Codificador automático. Es un algoritmo de inteligencia artificial generativa que permite la creación de nuevos datos.
- **PD:** Parkinson's Disease (Enfermedad de Parkinson). Se utiliza para referirse a personas que padecen la enfermedad.
- **HC:** Healthy Control (Control Sano o Grupo de control sano). Se utiliza para referirse a personas que no padecen la enfermedad de Parkinson.
- **HNR:** Harmonic to Noise Ratio o razón armónico-ruido en español. Es una medida que indica cuánto ruido aditivo existe sobre una señal de voz.
- **Frecuencia fundamental ( $f_0$ ):** Se define como la frecuencia más baja de los pliegues vocales cuando una persona habla.

# Índice de Contenidos

|                                                                            |           |
|----------------------------------------------------------------------------|-----------|
| <b>1. Introducción</b>                                                     | <b>1</b>  |
| 1.1. Motivación y contexto . . . . .                                       | 1         |
| 1.2. Planteamiento del problema . . . . .                                  | 2         |
| 1.3. Objetivos . . . . .                                                   | 2         |
| 1.4. Alcances y resultados esperados . . . . .                             | 2         |
| <b>2. Estado del Arte</b>                                                  | <b>3</b>  |
| 2.1. Enfermedad de Parkinson y su afección a la voz . . . . .              | 3         |
| 2.2. Redes neuronales convolucionales . . . . .                            | 4         |
| 2.2.1. Uso de la voz en la detección de patologías . . . . .               | 5         |
| 2.3. Aumento de data . . . . .                                             | 8         |
| 2.3.1. Aumento de data en señales de audio . . . . .                       | 9         |
| <b>3. Revisión de técnicas de aumento de data</b>                          | <b>10</b> |
| 3.1. Técnicas aplicadas en señal de audio en el dominio temporal . . . . . | 10        |
| 3.2. Técnicas escogidas para aumentar datos . . . . .                      | 11        |
| 3.2.1. Cambios de velocidad ( <i>speed change</i> ) . . . . .              | 11        |
| 3.2.2. Cambio del tono de voz ( <i>pitch change</i> ) . . . . .            | 11        |
| 3.2.3. Enmascaramiento . . . . .                                           | 12        |
| 3.2.4. Añadir ruido blanco gaussiano . . . . .                             | 13        |
| 3.2.5. Filtrado . . . . .                                                  | 13        |
| <b>4. Implementación técnica</b>                                           | <b>14</b> |
| 4.1. Herramientas . . . . .                                                | 14        |
| 4.2. Modelo . . . . .                                                      | 14        |
| 4.3. Bases de datos . . . . .                                              | 18        |
| 4.4. Implementación de aumento de data . . . . .                           | 19        |
| 4.4.1. Cambios de velocidad ( <i>speed change</i> ) . . . . .              | 20        |
| 4.4.2. Cambio del tono de voz ( <i>pitch change</i> ) . . . . .            | 20        |
| 4.4.3. Enmascaramiento . . . . .                                           | 21        |
| 4.4.4. Añadir ruido blanco gaussiano . . . . .                             | 22        |
| 4.4.5. Filtrado . . . . .                                                  | 22        |
| <b>5. Experimentos y Resultados</b>                                        | <b>24</b> |
| 5.1. Cambios de velocidad ( <i>speed change</i> ) . . . . .                | 24        |
| 5.2. Cambio del tono de voz ( <i>pitch change</i> ) . . . . .              | 26        |
| 5.2.1. Prueba 1 . . . . .                                                  | 27        |
| 5.2.2. Prueba 2 . . . . .                                                  | 28        |
| 5.3. Enmascaramiento . . . . .                                             | 29        |
| 5.4. Añadir ruido blanco gaussiano . . . . .                               | 31        |
| 5.5. Filtrado . . . . .                                                    | 33        |

|                                           |           |
|-------------------------------------------|-----------|
| 5.6. Resumen de resultados . . . . .      | 35        |
| 5.6.1. Reporte 1 . . . . .                | 35        |
| 5.6.2. Reporte 2 . . . . .                | 36        |
| 5.6.3. Reporte 3 . . . . .                | 38        |
| 5.7. Pruebas adicionales . . . . .        | 40        |
| 5.7.1. Mezcla de técnicas . . . . .       | 40        |
| 5.7.2. Mezcla de bases de datos . . . . . | 41        |
| <b>6. Conclusiones y trabajo futuro</b>   | <b>44</b> |



# Índice de Tablas

|                                                                                                                                                            |    |
|------------------------------------------------------------------------------------------------------------------------------------------------------------|----|
| 5.1. Resumen de métricas obtenidas en [20]. Entrenado con PD-GITA . . . . .                                                                                | 24 |
| 5.2. Resumen de métricas para speed change. Entrenado con PD-GITA . . . . .                                                                                | 25 |
| 5.3. Resumen de métricas para pitch change. Entrenado con PD-GITA . . . . .                                                                                | 27 |
| 5.4. Resumen de métricas para pitch change. Entrenado con PD-GITA y menor cantidad de datos aumentados.                                                    | 29 |
| 5.5. Resumen de métricas en prueba de parámetros para enmascaramiento. Entrenado con PD-GITA . . .                                                         | 29 |
| 5.6. Resumen de métricas en prueba de validación cruzada para enmascaramiento. Entrenado con PD-GITA                                                       | 29 |
| 5.7. Resumen de métricas en prueba de parámetro para ruido. Entrenado con PD-GITA . . . . .                                                                | 31 |
| 5.8. Resumen de métricas en prueba de validación cruzada para ruido. Entrenado con PD-GITA . . . . .                                                       | 32 |
| 5.9. Resumen de métricas en prueba de parámetro para filtrado. Entrenado con PD-GITA . . . . .                                                             | 33 |
| 5.10. Resumen de métricas en prueba de validación cruzada para filtrado. Entrenado con PD-GITA . . . . .                                                   | 34 |
| 5.11. Resumen de accuracy en validación con ambas bases de datos. Entrenado con PD-GITA . . . . .                                                          | 36 |
| 5.12. Resumen de métricas en caso de datos duplicados. Entrenado con PD-GITA . . . . .                                                                     | 36 |
| 5.13. Comparación de métricas de distintas técnicas de aumento de data con respecto a datos duplicados.<br>Entrenado con PD-GITA . . . . .                 | 37 |
| 5.14. Comparación de métricas en las distintas técnicas de aumento de data. Entrenado con PD-Neurovoz .                                                    | 38 |
| 5.15. Comparación de métricas de mezcla técnicas de aumento de data. Entrenado con PD-GITA . . . . .                                                       | 41 |
| 5.16. Comparación de métricas en mezcla de bases de datos. Validado con un promedio de ambas bases de datos.                                               | 41 |
| 5.17. Comparación de métricas en mezcla de bases de datos. Comparación entre modelo base y modelo con<br>Dominio Adversario. Obtenida desde [20] . . . . . | 42 |
| 5.18. Reporte de variabilidad de datos. Entrenado con mezcla de base de datos y aumento de data. . . . .                                                   | 42 |

# Índice de Figuras

|                                                                                                                                                                             |    |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----|
| 2.1. Representación de colores y sus canales. Figura obtenida del apunte del profesor Nicolás Torres para la asignatura ELO328 - Procesamiento digital de imágenes. . . . . | 4  |
| 2.2. Convolución de filtro e imagen. Figura obtenida del apunte del profesor Nicolás Torres para la asignatura ELO328 - Procesamiento digital de imágenes. . . . .          | 4  |
| 2.3. Arquitectura clásica de una CNN. Figura obtenida del apunte del profesor Nicolás Torres para la asignatura ELO328 - Procesamiento digital de imágenes. . . . .         | 5  |
| 2.4. Cálculo de la transformada rápida de Fourier (FFT). Imagen de MathWorks. . . . .                                                                                       | 6  |
| 2.5. Escala de conversión entre Mels y Hertz. . . . .                                                                                                                       | 7  |
| 2.6. Banco de filtros de Mel (Mel filterbank). Imagen de [18]. . . . .                                                                                                      | 7  |
| 2.7. Comparación entre espectrograma clásico y espectrograma de Mel. . . . .                                                                                                | 7  |
| 2.8. Técnicas de aumento de data en imágenes. Imagen de [22] . . . . .                                                                                                      | 8  |
| 4.1. Cambio de velocidad con dilatación temporal. $rate = 0,5$ . . . . .                                                                                                    | 20 |
| 4.2. Cambio de tono de voz con 4 semitonos hacia arriba. . . . .                                                                                                            | 20 |
| 4.3. Cambio de tono de voz con 4 semitonos hacia abajo. . . . .                                                                                                             | 20 |
| 4.4. Enmascaramiento en frecuencia y en tiempo. . . . .                                                                                                                     | 21 |
| 4.5. Comparación entre señal original y señal con ruido incorporado con distintas SNR. . . . .                                                                              | 22 |
| 4.6. Audio filtrado utilizando una frecuencia de corte entre 500 y 1000 Hz. . . . .                                                                                         | 23 |
| 5.1. Speed change: Curva de accuracy durante un entrenamiento de 100 épocas. . . . .                                                                                        | 25 |
| 5.2. Speed change: Curva de pérdida durante un entrenamiento de 100 épocas. . . . .                                                                                         | 25 |
| 5.3. Speed change: Curva ROC. Entrenado con PD-GITA. . . . .                                                                                                                | 25 |
| 5.4. Speed change: T-SNE de un entrenamiento con PD-GITA. . . . .                                                                                                           | 26 |
| 5.5. Pitch change: Curva de accuracy durante un entrenamiento de 100 épocas. . . . .                                                                                        | 27 |
| 5.6. Pitch change: Curva de pérdida durante un entrenamiento de 100 épocas. . . . .                                                                                         | 27 |
| 5.7. Pitch change: Curva ROC. Entrenado con PD-GITA . . . . .                                                                                                               | 27 |
| 5.8. Pitch change: T-SNE de entrenamiento con PD-GITA . . . . .                                                                                                             | 28 |
| 5.9. Enmascaramiento: Curva de accuracy durante un entrenamiento de 100 épocas. . . . .                                                                                     | 30 |
| 5.10. Enmascaramiento: Curva de pérdida durante un entrenamiento de 100 épocas. . . . .                                                                                     | 30 |
| 5.11. Enmascaramiento: Curva ROC. Entrenado con PD-GITA. . . . .                                                                                                            | 30 |
| 5.12. Enmascaramiento: T-SNE de entrenamiento con PD-GITA . . . . .                                                                                                         | 31 |
| 5.13. Ruido: Curva de accuracy durante un entrenamiento de 100 épocas. . . . .                                                                                              | 32 |
| 5.14. Ruido: Curva de pérdida durante un entrenamiento de 100 épocas. . . . .                                                                                               | 32 |
| 5.15. Ruido: Curva ROC. Entrenado con PD-GITA. . . . .                                                                                                                      | 32 |
| 5.16. Ruido: T-SNE de entrenamiento con PD-GITA . . . . .                                                                                                                   | 33 |
| 5.17. Filtrado: Curva de accuracy durante un entrenamiento de 100 épocas. . . . .                                                                                           | 34 |
| 5.18. Filtrado: Curva de pérdida durante un entrenamiento de 100 épocas. . . . .                                                                                            | 34 |
| 5.19. Filtrado: Curva ROC. Entrenado con PD-GITA. . . . .                                                                                                                   | 34 |
| 5.20. Filtrado: T-SNE de entrenamiento con PD-GITA . . . . .                                                                                                                | 35 |
| 5.21. Duplicado: T-SNE de un entrenamiento con PD-GITA . . . . .                                                                                                            | 37 |
| 5.22. Ruido: Curva de accuracy durante un entrenamiento de 100 épocas. Entrenado con PD-Neurovoz. . .                                                                       | 39 |
| 5.23. Ruido: Curva de pérdida durante un entrenamiento de 100 épocas. Entrenado con PD-Neurovoz. . .                                                                        | 39 |

|                                                                                 |    |
|---------------------------------------------------------------------------------|----|
| 5.24. Ruido: Curva ROC. Entrenado con PD-Neurovoz. . . . .                      | 39 |
| 5.25. Ruido: T-SNE de un entrenamiento con PD-Neurovoz. . . . .                 | 40 |
| 5.26. Mezcla: Curva de accuracy durante un entrenamiento de 100 épocas. . . . . | 41 |
| 5.27. Mezcla: Curva de pérdida durante un entrenamiento de 100 épocas. . . . .  | 41 |
| 5.28. Gráfica de T-SNE para mezcla de base de datos . . . . .                   | 43 |



# 1 | Introducción

La enfermedad de Parkinson es un trastorno de movimiento causado por una degeneración progresiva de células del sistema nervioso, se caracteriza por la presencia de complicaciones motoras, dificultad del habla, entre otras afecciones en los pacientes que la padecen.

Esta enfermedad ha sido estudiada por más de 200 años, el primer ensayo publicado respecto a esta enfermedad data del año 1817 escrito por James Parkinson, quien da nombre a la enfermedad. En este ensayo se expuso por primera vez un detalle de la sintomatología y se dieron los primeros indicios de que se trata de una enfermedad neurodegenerativa, caracterizada por su desarrollo progresivo [1]. Según estimaciones actuales, la Organización Mundial de la Salud (OMS) ha informado que la enfermedad de Parkinson en el año 2019 causó alrededor de 329.000 fallecimientos, lo que equivale a un aumento de más del 100 % desde el año 2000 y que su prevalencia se ha duplicado en los últimos 25 años [2].

Si bien la enfermedad de Parkinson es una de las dos enfermedades neurodegenerativas más comunes, su detección suele estar ligada a la aparición notoria de síntomas, sin embargo, existen estudios que indican que pacientes de la enfermedad presentan cambios patológicos como la muerte de neuronas dopaminérgicas previo a la aparición de síntomas clínicos [3]. Las neuronas dopaminérgicas son las encargadas de producir dopamina, un neurotransmisor que permite el funcionamiento tanto del sistema motor como de funciones cognitivas del ser humano.

Frente a esto, existe la creciente necesidad de poder diagnosticar en etapas tempranas el padecimiento y así poder actuar con tratamientos y fármacos que tengan una mayor efectividad en dicha etapa [4]. A día de hoy existen diversos estudios exponiendo trabajos con distintas técnicas para la orientación y diagnóstico temprano de la enfermedad, desde el uso de imágenes de resonancia magnética (MRI) hasta grabaciones de voz de pacientes.

Tomándose de lo último, la voz ha abierto un gran campo de estudio pues síntomas asociados al habla, como el volumen o la fluidez de ésta son de los primeros en aparecer, sin embargo, no son tan notorios para el oído humano en dichas primeras etapas y por tanto no es tan simple determinar la presencia de la enfermedad simplemente escuchando la voz de una persona. Ante esto y con el auge de las tecnologías de reconocimiento de voz utilizando técnicas de Machine Learning y Deep Learning es que se ha concentrado una gran cantidad de estudios para lograr la detección tanto de la enfermedad de Parkinson como de otras afecciones a través del uso de estas tecnologías.

## 1.1. Motivación y contexto

Dentro de las redes más comunes y que son ampliamente usadas en la actualidad se encuentran las llamadas Redes Neuronales Convolucionales o **CNN** por sus siglas en inglés, las que en simples palabras permiten la clasificación de imágenes en base a una serie de etapas de procesamiento. Si bien la detección de la enfermedad de Parkinson a través de la voz conlleva el uso de archivos de audio, el uso de **CNN** es posible gracias a técnicas de procesamiento de señales que permiten una representación gráfica de la voz.

Para poder entrenar modelos basados en CNN u otras técnicas de Deep Learning son necesarios una gran cantidad de datos, sobretodo para que dichos modelos generen resultados generalizables y puedan tener aplicaciones en el mundo real, sin embargo, esto no siempre es posible debido a las limitaciones asociadas a la toma o recolección de datos relacionadas al problema que se esté trabajando.

Una forma ampliamente usada para trabajar y abordar este problema es el aumento de data mediante distintas técnicas y transformaciones que se pueden aplicar tanto a imágenes como a señales de audio. Esto genera la llamada data sintética, es decir, datos creados en base a datos reales que han sido alterados por algún algoritmo o técnica y que son incluidos dentro del conjunto de entrenamiento de la red, esto con los objetivos de hacer crecer dicho conjunto e inducir variabilidad en los datos para que las redes mejoren su precisión y generalización.

Respecto a la enfermedad de Parkinson, es importante mencionar que entre las pruebas que las personas realizan para la grabación de estas bases de datos se encuentran la fonación sostenida de distintas vocales como /a/ u /o/, la entonación de pseudo palabras como /pa-ta-ka/ o /pekta/ e incluso la lectura de pequeños fragmentos de textos. La elección de qué tipo de audio se utilizará dependerá netamente del enfoque del estudio y cómo se procesarán los datos.

## 1.2. Planteamiento del problema

De manera más específica, el problema de la detección de Parkinson a través de la voz, utilizando redes neuronales convolucionales (CNN) yace en la cantidad de datos necesarios para obtener modelos con una buena precisión y generalización que permitan llegar a una aplicación para el mundo real. Para obtener los datos es necesario grabar la voz de personas con la enfermedad de Parkinson (PD) y personas sin la enfermedad (HC), lo que supone un gran desafío no solo porque se necesita de equipos de alta gama para la toma de datos precisos, sino que también por la disponibilidad de personas dispuestas a participar de estos de estudios. En la actualidad existen algunas bases de datos con voces de pacientes y personas sanas, sin embargo no cuentan con una gran cantidad de datos. Estas suelen contar con alrededor de 100 personas, lo cual no representa un número tan significativo si lo que se busca es generalizar un modelo.

## 1.3. Objetivos

En base al problema planteado anteriormente, se define como objetivo principal de este trabajo de memoria estudiar cómo influye el uso de distintas técnicas de aumento de data en la clasificación de la enfermedad de Parkinson utilizando redes neuronales convolucionales mediante señales de voz de la fonación /pa-ta-ka/.

Para alcanzar éste objetivo se plantean los siguientes objetivos específicos:

1. Estudiar y probar al menos 3 técnicas distintas de aumento de data para señales de audio de voz humana.
2. Entrenar un modelo basado en redes convolucionales con las distintas técnicas de aumento de data probadas.
3. Realizar validación cruzada para evaluar la generalización del modelo que ha utilizado data aumentada.

## 1.4. Alcances y resultados esperados

Se espera que las técnicas de aumento de data logren mejorar la precisión del modelo en la detección de la enfermedad de Parkinson dentro de la misma base de datos y a su vez se logre mejorar la generalización basándose en la variabilidad en los datos que entregarán las distintas técnicas a utilizar. En dicho caso, se podrá contar con información relevante para continuar el trabajo de modelos de detección de enfermedades, donde el uso de data sintética signifique aporte en dicha tarea.

Por otro lado, si el uso de las técnicas de aumento de data no muestran resultados satisfactorios, se tendrá información útil respecto a un camino que no es necesario continuar y por tanto se deberá explorar otras alternativas que ayuden a solventar el problema propuesto.

## 2 | Estado del Arte

Para entender y contextualizar este trabajo de memoria se necesita explicar tres conceptos principales, la enfermedad de Parkinson y cómo ésta afecta a la voz humana, las redes neuronales convolucionales (CNN) y el aumento de data o data augmentation. Particularmente es necesario repasar los principales avances realizados en la detección de enfermedades a través de la voz utilizando estos últimos dos conceptos.

Por lo anterior, este capítulo será dividido en tres secciones principales, la primera entrega una explicación de cómo afecta la enfermedad de Parkinson a la voz, luego, una revisión de lo que es una red neuronal convolucional y cuales son los principales aportes y trabajos realizados con ellas en la detección de enfermedades.

Finalmente, la tercera sección desarrolla la idea del aumento de data, mostrando sus avances al ser aplicada en la detección de imágenes y sonidos para finalmente entregar una revisión de trabajos relacionados a la detección de enfermedades utilizando redes neuronales convolucionales y aplicando técnicas de aumento de data para mejorar el rendimiento de ellas.

### 2.1. Enfermedad de Parkinson y su afección a la voz

La enfermedad de Parkinson es un trastorno neurodegenerativo que afecta de manera progresiva en el tiempo. Se caracteriza por la degeneración del sistema nervioso central, específicamente la disminución de la producción de dopamina. Esta reducción de dopamina afecta principalmente las funciones motoras del cuerpo, dando lugar a síntomas como temblores, rigidez muscular y dificultades de equilibrio.

Los síntomas de la enfermedad de Parkinson generalmente se clasifican en dos grupos: motores y no motores. Entre los síntomas motores se incluyen los temblores característicos, movimientos involuntarios (discinesias), pérdida de equilibrio y otros. Por otro lado, los síntomas no motores pueden abarcar demencia, trastornos del sueño y otras afecciones. Además, es importante destacar que muchos de estos síntomas tanto motores como no motores tienden a manifestarse en etapas más avanzadas de la enfermedad.

Además de los síntomas motores, las afecciones a la voz también son comunes en las etapas iniciales de la enfermedad de Parkinson debido a la pérdida de dopamina. Este neurotransmisor es fundamental para diversas funciones motoras, incluido el movimiento de los pliegues vocales y por tanto, la falta de dopamina puede generar problemas en el habla y la voz. El estudio [5] detalla una serie de complicaciones en la voz de las personas con Parkinson, divididas en dos grupos principales: fonación (producción de sonido) y prosodia (ritmo y entonación).

La disfonía es un ejemplo común de problema en la fonación de las personas, caracterizada por una pérdida en el timbre de la voz. Esta pérdida se traduce en una disminución de la razón armónico-ruido (HNR), que es una métrica que indica la cantidad de ruido aditivo presente en una señal de voz. Una HNR mayor suele asociarse con una mejor calidad de voz, por lo tanto, es correcto afirmar que la disfonía implica una disminución en la calidad vocal.

Por otro lado, la monotonía en el habla o en el volumen son ejemplos comunes de problemas en la prosodia. Se caracterizan por ser voces carentes de matices, lo que se traduce en disminuciones en la variabilidad de la frecuencia fundamental como de la intensidad, esta última relacionada con la energía con la que se emite el habla.

El estudio [6] entrega un análisis de diversos trabajos realizados en torno a la detección de la enfermedad de Parkinson a través de variadas técnicas y arquitecturas. Algunos de ellos trabajan con arquitecturas de Deep Learning

como redes neuronales convolucionales, mientras que otros con arquitecturas de Machine Learning clásico como Support Vector Machines. Por otro lado, se trabaja con la clasificación mediante parámetros y características de las señales de voz como HNR (Razón armónica-ruido), NHR (Razón ruido-armónica), frecuencia fundamental ( $f_0$ ), la transformada de Fourier para el análisis en frecuencia, entre otras. Lo anterior evidencia la importancia que se le ha dado a la detección de la enfermedad de Parkinson y la constante búsqueda de nuevas y mejores técnicas para cumplir con este objetivo.

## 2.2. Redes neuronales convolucionales

La convolución es una operación matemática que describe la superposición entre dos funciones, es decir, se realiza una multiplicación sucesiva entre dos funciones en la que una de las funciones se encuentra 'estática' y la otra es deslizada a través de la primera a medida que se van multiplicando sus valores.

Esta operación es importante pues es la componente principal de las redes neuronales convolucionales, en ellas, la convolución es utilizada para detectar patrones en una imagen, los que serán utilizados para la clasificación.

Para procesar las imágenes es importante saber que estas son trabajadas como matrices, donde cada coordenada (píxel) corresponde a un valor entre 0 y 255 o bien entre 0 y 1 en caso de trabajar con imágenes normalizadas. Otro concepto importante es el de canal, un canal hace referencia una tercera dimensión que puede tener la imagen relacionada al color. En el caso de imágenes en escala de grises estas cuentan con un sólo canal, mientras que para imágenes a color, cuentan con 3 canales R, G y B. Cada canal será su propia matriz y cada píxel representa la intensidad del color. En el caso de imágenes RGB se utiliza una superposición de los 3 canales para lograr la imagen completa.

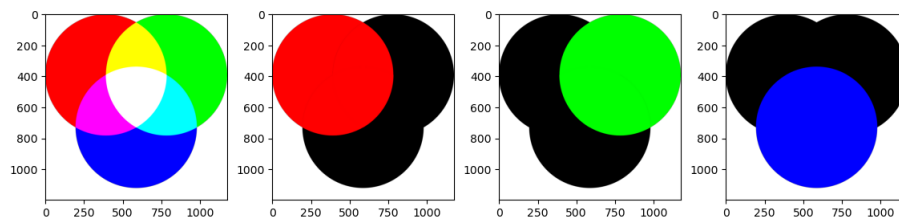


**Figura 2.1:** Representación de colores y sus canales. Figura obtenida del apunte del profesor Nicolás Torres para la asignatura ELO328 - Procesamiento digital de imágenes.

De manera más específica, la aplicación de filtros se realiza mediante la convolución entre una máscara (o filtro), una matriz cuyas dimensiones son típicamente 3x3, y los píxeles de la imagen como se muestra en la siguiente figura:

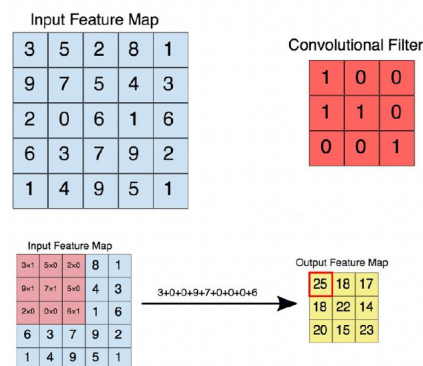


**Figura 2.2:** Convolución de filtro e imagen. Figura obtenida del apunte del profesor Nicolás Torres para la asignatura ELO328 - Procesamiento digital de imágenes.

Dicho filtro recorre todos los píxeles de la imagen y genera una nueva matriz, cuyas dimensiones finales dependerán de diversos parámetros pero suele ser del mismo tamaño o menor que la imagen original. Este proceso corresponde a la primera etapa de una CNN clásica y sirve para la extracción de características, luego del filtro se utiliza una capa de ‘Pooling’ con el objetivo de reducir la dimensión, esta capa es similar a la anterior en la que se aplica un filtro a la imagen pero de dimensiones 2x2, típicamente se utiliza Max-pooling, donde se selecciona el píxel de mayor valor dentro de dicho filtro.

Comúnmente, las redes neuronales convolucionales cuentan con 2 secciones, la primera encargada de la extracción y aprendizaje de características y la segunda encargada de la clasificación. En tal caso, la primera sección consta de una serie de bloques de filtros convolucionales seguidos de capas de pooling. Luego, se utiliza una capa de conexión de tipo Flatten, la cual transforma la matriz resultante de la primera sección en un vector que será utilizado en la segunda. La sección de clasificación consta de una serie de capas ‘fully connected’ y finalmente una capa Softmax para entregar la predicción final. En la siguiente figura se explica dicho proceso.

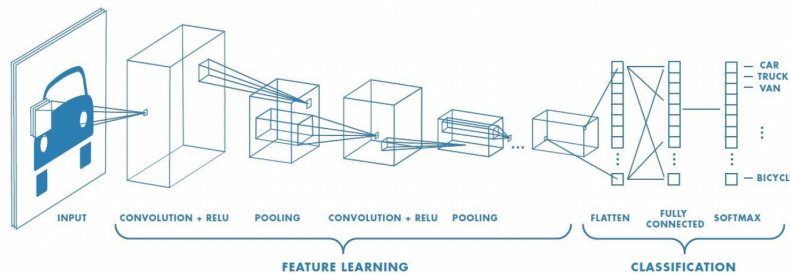


**Figura 2.3:** Arquitectura clásica de una CNN. Figura obtenida del apunte del profesor Nicolás Torres para la asignatura ELO328 - Procesamiento digital de imágenes.

Este tipo de redes ha sido utilizada en una variedad de tareas como la visión por computador, procesamiento de lenguaje natural, segmentación y por sobretodo para tareas de clasificación de imágenes [7]. Por otro lado, ha sido trabajada en diversos estudios y se han propuesto distintas arquitecturas y modelos pre-entrenados como AlexNet, GoogLeNet o SqueezeNet, los que varían el diseño básico agregando o quitando capas, alterando el orden u otros cambios con el objetivo de mejorar el rendimiento frente a los problemas que se trabajan y los recursos disponibles [8].

Actualmente, han sido utilizadas para la detección de enfermedades neurodegenerativas como la enfermedad de Alzheimer o de Parkinson, e incluso para la detección temprana de nódulos en las glándulas mamarias o en los pulmones en base a imágenes médicas obtenidas con técnicas como ultrasonido o resonancias magnéticas [9] consiguiendo grandes resultados. Un ejemplo de esto es el trabajo realizado en [10], donde utilizando redes neuronales convolucionales e imágenes de resonancia magnética se logra reportar una exactitud del 99,9 % en la detección de la patología.

Por otro lado, en el artículo [11] se presenta un híbrido entre una CNN y una red LSTM (Long short-term memory) para la clasificación de imágenes de histopatologías de tumores relacionados al cáncer de mama en benignos o malignos, logrando una exactitud del 99 % para la clasificación binaria y un 92,5 % para una clasificación multi-clase en la que se consideran subtipos de estos tumores.

En esta línea, el artículo [12] realiza un repaso por las principales arquitecturas y enfoques que tiene el trabajo con redes neuronales convolucionales junto al uso de imágenes médicas, quedando expuesto el masivo uso de este tipo de redes en este contexto.

### 2.2.1. Uso de la voz en la detección de patologías

Si bien se ha mencionado sólo el uso de exámenes cuyos resultados son imágenes médicas para la detección de patologías utilizando CNN, estos no son el único enfoque que se ha tomado para realizar esta tarea. Mediante la voz humana es posible detectar no solo algunas de las afecciones relacionadas a la enfermedad de Parkinson mencionadas

anteriormente como la disfonía sino que otros problemas asociados como los nódulos en los pliegues vocales [13], o también desarrollos asociados al reconocimiento de emociones [14], logrando una exactitud del 78 % y una menor tasa de falsos positivos, e incluso la detección patologías de origen psicológico como la depresión, como se expone en [15], donde se presenta un sistema clínico de detección de dicha patología.

Según lo escrito anteriormente, las CNN deben ser trabajadas con imágenes debido a su naturaleza, por tanto, para poder utilizar señales de audio con este tipo de red es necesario aplicar una conversión. El análisis de audio con CNN suele ser trabajado con espectrogramas, un tipo de representación que muestra el comportamiento espectral de la señal respecto del tiempo. Específicamente, para el caso de voz humana, se trabaja con los llamados espectrogramas de Mel.

Un espectrograma es generado calculando sucesivamente la transformada rápida de Fourier (FFT) en ventanas de la señal para luego graficar el espectro resultante respecto del tiempo a medida que dicha ventana va avanzando a lo largo del audio, además, se utiliza una escala de color que representa la magnitud de cada frecuencia representada en el espectrograma. A continuación se puede ver una representación del proceso de obtención de cada ventana de señal.

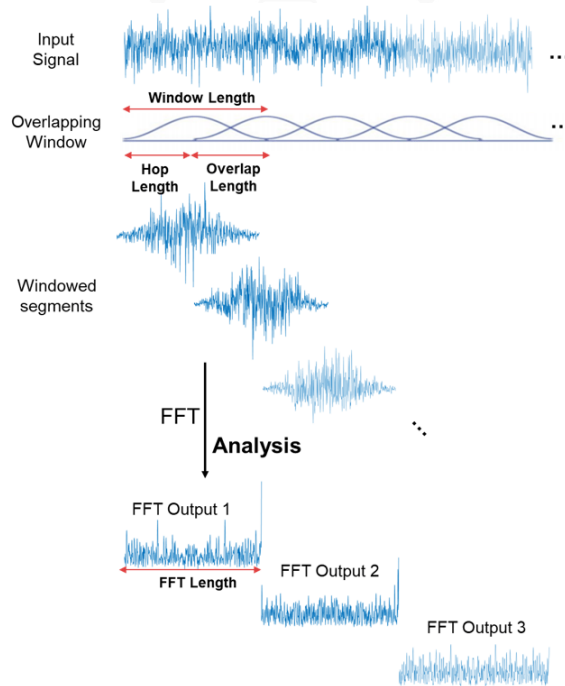


**Figura 2.4:** Cálculo de la transformada rápida de Fourier (FFT). Imagen de MathWorks.

Por otro lado, la escala de Mel se basa en unidades de ‘Pitch’ o bien ‘Mels’, los que representan a la frecuencia en Hertz en base a una transformación no lineal. Esta escala se caracteriza por representar las diferencias de frecuencia de manera equitativa en base a la percepción de las personas, esto pues, una persona no percibe de la misma manera una diferencia de tonos a baja frecuencia que una diferencia de igual magnitud pero en alta frecuencia.

La transformación entre Mels y Hertz utilizada en la librería ‘Torchaudio’ [16], parte del framework ‘PyTorch’ [17] es:

$$mel = 2595 \cdot \log_{10} \left( 1 + \frac{freq}{700} \right) \quad (2.1)$$

Cuya representación gráfica se encuentra en la figura 2.5

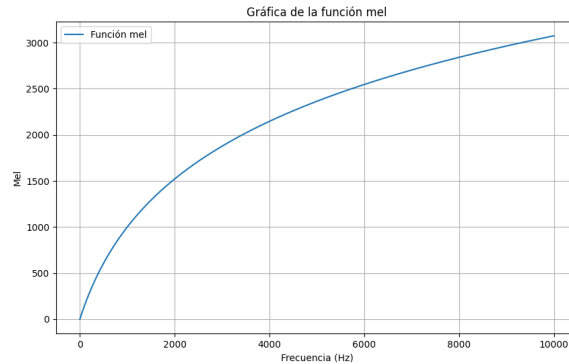


**Figura 2.5:** Escala de conversión entre Mels y Hertz.

Finalmente, un espectrograma de Mel consta de escalar las frecuencias de un espectrograma común utilizando un conjunto de filtros de Mel (Mel filterbank). Este banco cuenta con una serie de filtros triangulares que emulan el comportamiento de la escala de Mel, se caracterizan por contar con una magnitud de 1 en su frecuencia central y tendiendo a 0 en sus extremos. Para escalar el espectrograma original se define una cantidad de filtros a utilizar en base a las frecuencias que se trabajan. A continuación se puede ver un ejemplo de un banco de filtros de Mel:

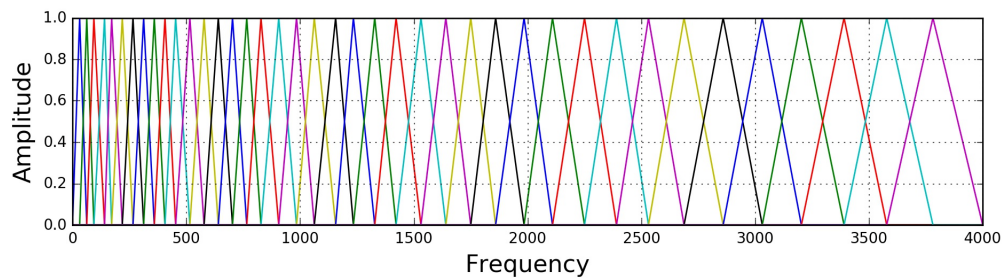


**Figura 2.6:** Banco de filtros de Mel (Mel filterbank). Imagen de [18].

A continuación se puede ver una comparación entre un espectrograma original y el espectrograma de Mel.

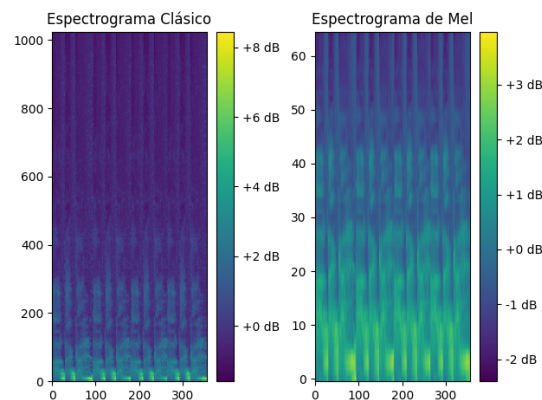


**Figura 2.7:** Comparación entre espectrograma clásico y espectrograma de Mel.

A modo de ejemplificar el uso de espectrogramas en la detección de la enfermedad de Parkinson se rescata el trabajo desarrollado en [19], donde los autores trabajan en un modelo capaz de detectar esta patología con una exactitud del 99 % utilizando la voz de personas entonando la vocal /a/ de manera sostenida. Dentro de su implementación se realiza la conversión de las señales de audio a espectrogramas para poder trabajar con CNN, demostrando que esta vía genera resultados alentadores para el diagnóstico ambulatorio de la enfermedad.

Por otro lado, en [20] se trabaja un modelo de clasificación que trabaja con la fonación /pa-ta-ka/ y que además evalúa la generalización del modelo entrenado. El modelo base logra una exactitud de clasificación del 80,8 %, sin embargo, al realizar una prueba cruzada, esta medida baja considerablemente a 47,7 %, evidenciando uno de los principales problemas mencionados, donde debido a la falta de datos de entrenamiento y la variabilidad en ellos se genera resultados buenos pero que no son replicables.

Finalmente, a modo de reafirmar lo anterior, se presenta el estudio [21], en el cual también se trabaja la detección de Parkinson a través de la voz y se realiza una prueba de validación cruzada con distintas bases de datos, reportando caídas cercanas al 50 % en exactitud de clasificación en distintos casos.

## 2.3. Aumento de data

Con el fin de abordar la problemática expuesta anteriormente, se han desarrollado diversas herramientas y técnicas para realizar un aumento de data en base a transformaciones a los datos originales. Esta nueva data tiene la particularidad de mantener las etiquetas de las imágenes originales y se suele llamar data sintética o data aumentada y es utilizada para entrenar los modelos de deep learning en conjunto a la data original, el objetivo principal de utilizar este tipo de datos es robustecer al modelo frente a variaciones de tamaño, color, orientación u otras características. Esto permite aumentar la cantidad y la variabilidad de los datos, además de mejorar las métricas del modelo en gran parte de los casos.

En imágenes, se trabaja con distintos tipos de transformaciones, las clásicas y más utilizadas están asociadas a la posición y tamaño de los objetos que se visualizan, es decir, se trabaja con la rotación en distintos ángulos, acercamientos/alejamientos o inversión (espejo).

Sin embargo, la generación de nuevos datos con esta finalidad no solo es posible con dichas transformaciones, también se puede realizar aplicando filtros de color, añadir ruido, superponer capas para difuminar la imagen e incluso utilizar redes generativas antagónicas (GAN). En la siguiente imagen se puede ver un ejemplo de distintas técnicas de aumento de data en imágenes:

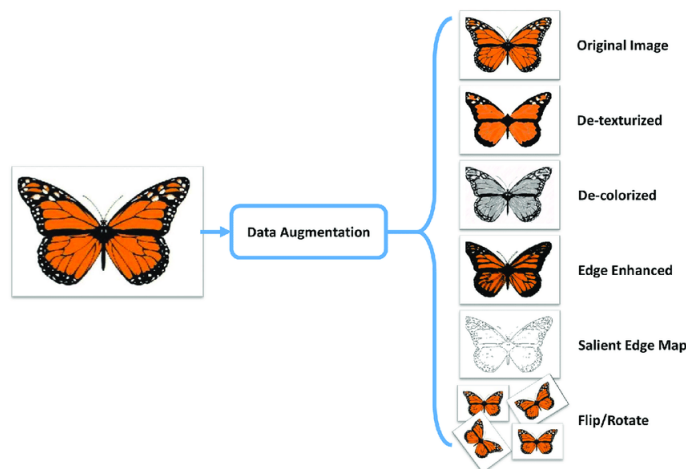


**Figura 2.8:** Técnicas de aumento de data en imágenes. Imagen de [22]

En [23], se realiza una revisión de técnicas de aumento de data y cuál es su eficacia en la clasificación de imágenes, llegando a la conclusión que el aumento mediante técnicas clásicas ya mejora bastante y entrega resultados prometedores, además, a pesar de que el uso de *GAN* también resulta en mejoras a la clasificación, el costo computacional que requieren para generar las nuevas imágenes no la hacen una opción tan atractiva frente al comportamiento de las transformaciones mencionadas.

### 2.3.1. Aumento de data en señales de audio

En el contexto del procesamiento de señales de audio, también es posible encontrar técnicas de aumento de datos, aunque distan de las empleadas en el procesamiento de imágenes. Entre las diversas estrategias propuestas, se destacan ‘pitch shift’, que modifica la frecuencia fundamental del audio, y ‘time stretch’, que permite ajustar la duración temporal del sonido, ya sea acelerándolo o ralentizándolo. Estas técnicas son fundamentales para enriquecer conjuntos de datos limitados y mejorar la capacidad de generalización de los modelos de procesamiento de señales de audio.

Es importante mencionar que si bien, las grabaciones de voz son transformadas a un espectrograma, que es una imagen, aplicar técnicas destinadas a imágenes sobre estas representaciones carece de sentido pues la información está en la forma del espectrograma y por tanto hacer una rotación, cambiar sus colores o técnicas relacionadas puede generar información no relevante o no relacionada al problema que se está trabajando.

En [24], se expone una serie de estudios que aplican técnicas de aumento de data para la detección de distintas patologías, entre las más mencionadas se encuentran el cambio en la frecuencia fundamental (*pitch change*) y la inclusión de ruido al audio original. Además, en el mismo estudio se realiza una comparación entre distintas técnicas y su aporte utilizando distintas arquitecturas de redes neuronales. Centrándose en CNN de 2 dimensiones se obtiene que las técnicas utilizadas logran mejorar las métricas del modelo base entre un 2 y 3 % aproximadamente. En sus conclusiones resaltan la recomendación del uso de las 6 técnicas de aumento de datos probadas en tareas de detección de patologías a través de la voz. Estas técnicas se revelan como beneficiosas al aumentar la variabilidad de los datos, lo que contribuye a mejorar la generalización del modelo y a controlar el fenómeno de ‘overfitting’, donde el modelo se ajusta demasiado a los datos de entrenamiento. Esto sugiere que la implementación de estrategias de aumento de datos puede ser crucial para mejorar el rendimiento y la robustez de los modelos en esta área de investigación.

Destacan además el uso de la técnica de enmascaramiento debido a presentar un mejor rendimiento en la tarea realizada. Ésta técnica es una de las utilizadas en este trabajo y es detallada más adelante, pero principalmente consiste en enmascarar bloques temporales y de frecuencia. Cabe destacar que el estudio mencionado trabaja con bases de datos libres y no específicamente en la detección de la enfermedad Parkinson por lo que sus resultados no son comparables con los de éste trabajo pero sí sirven como guía.

Por otro lado, en [25], se presenta un modelo de detección de la enfermedad de Parkinson a través de la voz utilizando técnicas de aumento de data, específicamente se utiliza 6 técnicas aplicadas directamente a la señal de audio y se reporta que en el caso de vocales sostenidas, en 5 de cada 6 casos, las técnicas utilizadas logran mejorar la exactitud de clasificación en un máximo de 9 %. Mientras que para frases más complejas esta mejora no es tan significativa. Finalmente se menciona que se debe trabajar en estudiar cómo afecta en los resultados cada una de las modificaciones de la voz de manera separada y como ayudan a la generalización del modelo, siendo este uno de los objetivos del presente trabajo de memoria.

## 3 | Revisión de técnicas de aumento de data



Existen diversas técnicas que permiten realizar aumento de data en señales de audio, las que pueden ser clasificadas en tres grandes grupos, técnicas aplicadas sobre el audio en el dominio temporal, técnicas aplicadas sobre sus espectrogramas (dominio de la frecuencia) y generación de datos nuevos mediante redes *GAN* o *VAE*. A continuación se realizará una revisión concisa de una variedad de técnicas para luego centrarse y detallar las que serán aplicadas más adelante.

### 3.1. Técnicas aplicadas en señal de audio en el dominio temporal

1. **Corrimiento temporal (*Time shifting*):** Consiste en aplicar un corrimiento de duración aleatoria hacia la izquierda o hacia la derecha de manera circular. La técnica es utilizada en [25], sin embargo no se entregan mayores detalles sobre su rendimiento. Cuenta con una variación llamada ‘reverse’, la que consiste en voltear la señal de audio de derecha a izquierda.
2. **Cambio en el volumen (*Volume change*):** Implica un cambio en la amplitud de la señal mediante un aumento o disminución de ésta.
3. **Recorte (*Clipping*):** implica limitar la amplitud del audio a un máximo permitido, recortando la señal si dicho límite es excedido. Una variante de esta técnica, conocida como normalización de picos (*peak normalization*), ajusta la amplitud de la señal para que el pico más alto alcance un máximo permitido. Estas técnicas se suelen utilizar en situaciones donde se necesita controlar la amplitud del audio, como en la producción musical para garantizar una reproducción adecuada y evitar la distorsión
4. **Re-muestreo (*Resample*):** Esta técnica se caracteriza por modificar la frecuencia de muestreo de la señal, ya sea aumentándola o disminuyéndola. El re-muestreo es comúnmente utilizado en conjunto con otras técnicas de procesamiento de señales de audio para adaptar la señal a ciertos requerimientos o estándares. Por ejemplo, se puede utilizar en el procesamiento de audio para cambiar la duración temporal de la señal sin alterar su contenido espectral.
5. **Remix:** En esta técnica se realiza la creación de nuevas señales de audio mediante el reordenamiento de intervalos de tiempo de la señal original.
6. **Atenuación (*Fade*):** Consiste en la aplicación progresiva de una disminución en la amplitud de la señal, ya sea al principio (atenuación de entrada) o al final (desvanecimiento de salida).
7. **Inversión de la polaridad:** Es utilizada comúnmente en modelos que buscan ser robustos frente a inversiones de fase. Se aplica multiplicando por  $-1$  el valor de la magnitud, o, en otras palabras, invirtiendo la fase.
8. **Cambio de canales:** Esta técnica es utilizada para evitar sesgos de posición y consiste en intercambiar la información del canal izquierdo con la del canal derecho, sin embargo, no puede ser utilizada en señales de audio mono dado que sólo cuentan con un canal.

## 3.2. Técnicas escogidas para aumentar datos

### 3.2.1. Cambios de velocidad (*speed change*)

La técnica de aumento de data ‘speed change’ o cambio de velocidad en español, también es conocida como ‘time stretch’, indicando una dilatación o compresión en el tiempo del audio. Es importante destacar que esta técnica no realiza un cambio en el tono de voz, sólo en la velocidad, comportamiento que se logra gracias al algoritmo ‘Phase Vocoder’, el cual corrige discontinuidades en la fase de la señal de audio durante la dilatación o compresión. Estas correcciones garantizan que la señal sea continua en el dominio del tiempo y se obtenga la modificación deseada sin afectar el tono de la voz.

La motivación para utilizar esta técnica es la posibilidad de simular lecturas más rápidas o más lentas del fonema /pa-ta-ka/, buscando que el modelo sea más robusto frente a cambios en la velocidad de habla pues si bien la lentitud en el habla se relaciona con uno de los síntomas de la enfermedad de Parkinson es importante que no sea la única característica considerada dado que puede haber personas sanas que hablen más lento por otras razones.

Para utilizar éste método se trabaja con la función ‘time\_stretch’ de ‘Librosa’ basándose en lo expuesto en [24]. Esta función debe ser configurada con una tasa o rate, que representa un factor de aceleración en caso de ser superior a 1 o de ralentización en caso de estar entre 0 y 1. En caso de utilizar el valor 1 la señal resultante no presenta modificaciones.

Dentro de la implementación de esta técnica se cuenta con 3 fases, en primer lugar se calcula la **transformada de Fourier en tiempo corto (STFT)** generando un vector complejo con la magnitud y la fase de la respuesta en frecuencia de la señal. Luego se utiliza la implementación de *phase\_vocoder* que incluye la librería *Librosa* la cual realiza la aceleración o ralentización respectiva en base al parámetro *rate*, y finalmente se utiliza la **transformada inversa (ISTFT)** para devolver la señal al dominio temporal.

### 3.2.2. Cambio del tono de voz (*pitch change*)

Esta técnica de aumento de data consta de un cambio en la frecuencia fundamental del audio, en otras palabras la frecuencia más baja de la onda de sonido transmitido.

La frecuencia fundamental de las personas varía entre los 110 a los 230 Hz aproximadamente, dependiendo del género de la persona. Típicamente la voz masculina oscila en frecuencias más bajas, mientras que las voces femeninas oscilan en las frecuencias más altas del rango mencionado.

Otro concepto importante para el trabajo con *pitch change* es el de una ‘Octava’, que hace referencia al espacio entre una frecuencia y el doble de ésta. Por ejemplo, entre 110 y 220 Hz hay una octava de diferencia. Si dicha octava es dividida en 12 partes iguales, cada rango resultante será considerado *un semitono*.

La principal motivación para utilizar y probar esta técnica es la posibilidad de simular nuevas voces, dado que se puede modificar la frecuencia fundamental, se generará espectrogramas que asemejen tonos de voz distintos, dando la posibilidad de contar con ‘nuevos pacientes’.

Para llevar a cabo este cambio en la frecuencia fundamental se utiliza la función ‘pitch\_shift’ de la librería *Librosa*, como se menciona en [24]. En su implementación, se sigue un proceso que implica dos etapas principales.

Primero, se utiliza la función *time\_stretch* mencionada anteriormente. Esta función se utiliza con un factor de cambio (*rate*), que se calcula utilizando la siguiente fórmula:

$$rate = 2^{\frac{n_{steps}}{bins\_per\_octave}} \quad (3.1)$$

Donde *bins\_per\_octave* indica la cantidad de divisiones que habrá en una octava, parámetro que por defecto es igual a 12 y por tanto cada paso será considerado un semitono. El parámetro *n\_steps* determina la cantidad de pasos que se modifica la señal, aumentando si es positivo y disminuyendo si es negativo.

Después de aplicar *time\_stretch*, se necesita ajustar la señal resultante al tamaño original del audio de entrada para mantener la velocidad del habla. Esto se logra mediante un módulo de *remuestreo*, donde se modifica la nueva frecuencia de muestreo generada por *time\_stretch* hacia la frecuencia de muestreo original del audio de entrada. Esto, a su vez, conlleva a un cambio en el tono de la voz. El cambio en la frecuencia de muestreo se rige por la siguiente ecuación:

$$newSR = \frac{oldSR}{2^{\frac{nsteps}{bins\_per\_octave}}} \quad (3.2)$$

Donde oldSR es la frecuencia de muestreo original y newSR es la nueva frecuencia de muestreo ajustada. Este proceso garantiza que la señal final tenga la misma velocidad del habla que la señal original, pero con la frecuencia fundamental modificada según los parámetros especificados.

### 3.2.3. Enmascaramiento

SpecAugment es un conjunto de técnicas aplicadas directamente en el espectrograma de una señal de audio, este fue publicado por el equipo de investigación de Google y está enfocado en tareas de reconocimiento de voz [26]. Este consta de la aplicación de 3 transformadas sobre el espectrograma, la primera es una deformación temporal, mientras que las otras 2 se relacionan a la aplicación de una máscara ya sea en la frecuencia o en el tiempo.

Aplicar estas máscaras implica cubrir selectivamente la información en una ventana específica de frecuencia o tiempo, cuyo tamaño se elige de manera aleatoria dentro de ciertos límites definidos por parámetros máximos. Al trabajar con la creación de imágenes aumentadas, se manejan tres parámetros principales. El primero es el parámetro de **frequency/time masking**, utilizado para limitar el tamaño máximo de la ventana que se aplicará, se representan por las letras **F/T** según corresponda. Este parámetro se define al llamar a la función. Los otros dos parámetros se configuran durante el proceso de enmascaramiento de un espectrograma con estas técnicas. El primero es el tamaño de la ventana a cubrir ( $f/t$ ), que se elige aleatoriamente dentro de una distribución uniforme que va desde 0 hasta F/T. El segundo es el punto inicial de la ventana a cubrir, denotado como  $f_0 / t_0$ , y se selecciona aleatoriamente en un rango de  $[0, \nu - f) / [0, \tau - t)$ , respectivamente. Donde  $\nu$  representa el número total de canales de frecuencia y  $\tau$  el número total de intervalos de tiempo.

En su estudio se indica que la primera técnica busca robustecer el modelo frente a deformaciones en el tiempo, mientras que las otras dos buscan robustecerlo frente a pérdidas de información ya sea espectral o bien del discurso. Frente a esto se decide que en este trabajo se utilizará solo las técnicas de enmascaramiento temporal y en frecuencia, con el fin simular la pérdida de información por cortes en el habla de pacientes o pérdidas por el procesamiento del audio. Además, en [24] también se considera sólo estas dos transformaciones.

Para aplicar las técnicas de enmascaramiento se importa dos funciones desde la sección de transformadas de *TorchAudio* llamadas **frequencyMasking** y **timeMasking**. En cada una se debe configurar el parámetro F/T respectivamente, que indica el tamaño máximo de la ventana a enmascarar como se menciona anteriormente. Para realizar el enmascaramiento, dentro de la implementación, ambas técnicas realizan un llamado interno a la función **mask\_along\_axis\_iid** desarrollada por *TorchAudio*, en la que se cuenta con un parámetro que indica el eje en el cuál se enmascarará (Eje X para el tiempo, Eje Y para la frecuencia).

Para realizar dicho enmascaramiento primero se selecciona la cantidad de bloques de frecuencia o tiempo que serán cubiertos en base al parámetro máximo que es entregado por el usuario, luego se genera una máscara binaria con las mismas dimensiones del espectrograma de entrada en la que se marca con el valor 0 las frecuencias o tiempos que serán enmascarados y con 1 los valores que permanecerán sin cambios. Luego la máscara es aplicada multiplicando cada punto del espectrograma por su valor en la máscara, en el caso de las frecuencias o tiempos a cubrir serán multiplicados por 0 y por tanto serán atenuados.

### 3.2.4. Añadir ruido blanco gaussiano

Como es mencionado en [24], añadir ruido a un audio es una de las técnicas más comunes de aumento de data y es normalmente usada para mejorar la robustez frente a contaminación acústica en las señales a analizar o bien artefactos introducidos por las tecnologías utilizadas en la toma de datos. En el caso de [25] se utiliza ruido de color para simular señales grabadas en otros ambientes, mientras que en [24] utilizan ruido aditivo de ‘murmillos’ utilizando una base de datos cuya referencia se encuentra en el mismo artículo.

En general, la técnica consta de generar un vector de valores aleatorios según una distribución de probabilidad (gaussiana, uniforme u otras), escalar dichos valores en base a un factor de ruido que determina cuánto afectará el ruido a la señal y luego sumar este vector a la señal de audio generando una señal ruidosa.

En este caso se trabajará con la razón señal a ruido (SNR) para calcular el factor de ruido que deberá escalar cada audio. Esta SNR se calcula con la siguiente ecuación:

$$SNR = 10 \cdot \log \left( \frac{P_{signal}}{P_{noise}} \right) \quad (3.3)$$

Luego, realizando un despeje matemático, la potencia del ruido se puede calcular con la siguiente expresión:

$$P_{noise} = \frac{P_{signal}}{10^{\frac{SNR}{10}}} \quad (3.4)$$

Finalmente, el factor de ruido se puede calcular como:

$$Noisefactor = \sqrt{P_{noise}} \quad (3.5)$$

En general, mientras más alta la SNR, mejor es la calidad del audio pues existe poca presencia de ruido.

### 3.2.5. Filtrado

Un filtro se define como un algoritmo diseñado para atenuar determinadas frecuencias según su configuración. Esta técnica, también empleada en [25], resulta fundamental para fortalecer el modelo ante la presencia de ruidos de fondo o frecuencias no deseadas.

Existen distintos tipos de filtros dependiendo de su comportamiento en frecuencia, entre los principales se encuentran los filtros pasa bajos, que atenúan las frecuencias por sobre una determinada ‘frecuencia de corte’, mientras que los filtros pasa altos atenúan por debajo dicha frecuencia de corte. Por su parte, los filtros pasa banda cuentan, como su nombre lo indica, con una banda de frecuencias de paso, cualquier frecuencia fuera de ese rango será atenuada, mientras que los filtros rechaza banda cuentan con un comportamiento contrario, en el que las frecuencias dentro de una determinada banda serán atenuadas.

Son distintas las librerías de Python que implementan herramientas de filtrado, por su parte *TorchAudio*, dentro de su funcionalidad ‘sox\_effects’ permite el uso de filtros y otras técnicas de aumento de data, sin embargo no se encuentra lo suficientemente documentado para saber como manejar de buena manera sus parámetros.

Otro caso es la librería ‘SciPy’ [27] la cual cuenta con una amplia variedad de funciones para el trabajo con señales, entre las cuales se encuentra una para diseñar filtros Butterworth, un tipo de filtro cuya banda de paso de frecuencias tiene a ser lo más plana posible y su transición de caída busca ser suave. Dicha implementación es una buena opción, sin embargo se genera un sólo tipo de filtro y por tanto cada audio sería atenuado desde la misma frecuencia en adelante.

Finalmente, la librería ‘Audiomentations’<sup>1</sup> cuenta con distintas funciones que permiten aplicar distintos tipos de filtros cuyas características son escogidas de manera aleatoria en base a los parámetros mínimos y máximos que sean configurados. Esta implementación es una buena opción para realizar aumento de data pues induce variabilidad en los resultados.

<sup>1</sup><https://github.com/iver56/audiomentations>

## 4 | Implementación técnica

### 4.1. Herramientas

Para la implementación técnica de éste trabajo de memoria se opta por usar el lenguaje de programación *Python* debido a su extenso ecosistema de desarrollo relacionado con el Aprendizaje Profundo y sus librerías asociadas. Específicamente, se utiliza el framework *PyTorch* para la implementación de redes neuronales, *Scikit-Learn* [28] para el cálculo de métricas, y *Librosa*, *TorchAudio* y *Audiomentations* para el procesamiento de las señales de audio y la aplicación de técnicas de aumento de datos. Además se hace uso de librerías fundamentales para la manipulación de números, gráficas, etc., como *Numpy* [29] o *Matplotlib* [30], entre otras.

Por otro lado, el código es desarrollado y ejecutado utilizando la plataforma *Google Colab* debido a los recursos de GPU y almacenamiento que ofrece de manera gratuita. Finalmente, el código generado es cargado a un repositorio en *GitHub* para que esté disponible para futuros trabajos.<sup>2</sup>

Para analizar y clasificar las voces presentes en las bases de datos utilizando una red neuronal convolucional se utilizará la representación gráfica de su comportamiento en frecuencia respecto del tiempo, es decir, su espectrograma de Mel. Este tipo de representación es generada utilizando la función *MelSpectrogram* disponible en el módulo de transformadas de *TorchAudio*.

### 4.2. Modelo

Es importante mencionar que para contar con una base de comparación y conocer si el aumento de data es efectivamente una ayuda a la detección de manera generalizada de la enfermedad de Parkinson se utiliza como referencia el código del artículo [20] disponible en *GitHub*<sup>3</sup>. La implementación será dividida en 4 etapas principales que serán detallados a continuación:

1. **Procesamiento de datos:** Ésta primera etapa realiza principalmente dos transformaciones a las señales de audio, primero se realiza una normalización en base al máximo valor de amplitud para luego aplicar una segmentación que genera intervalos con una duración de 400 ms y una superposición del 50 %. Esto genera segmentos de igual tamaño, estos segmentos son asociados tanto a la etiqueta del audio original como al sujeto al que pertenecen.
2. **Espectrogramas de Mel:** La siguiente etapa realiza el cálculo del espectrograma de Mel de cada uno de los segmentos obtenidos anteriormente. Para el cálculo de estos espectrogramas se utiliza una ventana de 15 ms, un tamaño de salto de 10 ms y 65 bandas de Mel. Esto genera espectrogramas de 65x41 puntos.
3. **Entrenamiento del modelo:** Como es mencionado anteriormente, se trabaja con la misma red utilizada en [20]. Esta red sigue una arquitectura estándar de redes neuronales convolucionales que consta de 2 módulos, un extractor de características y uno clasificador.

El extractor de características realiza la tarea de identificar y extraer las características relevantes de los espectrogramas para su posterior clasificación. Este extractor consta de dos bloques convolucionales secuenciales. Cada

---

<sup>2</sup><https://github.com/MatiValenzuela/EL0308-DataAugmentationForParkinsonDetection>

<sup>3</sup>[https://github.com/Emiroji/Domain\\_Adversarial\\_CNN\\_Speech\\_Parkinson\\_Classification/blob/main](https://github.com/Emiroji/Domain_Adversarial_CNN_Speech_Parkinson_Classification/blob/main)

bloque incluye una capa convolucional 2D con un tamaño de kernel de 8 y 9 (para cada bloque, respectivamente), seguida de una capa de normalización por lotes (*Batch Normalization*), una función de activación ReLU, una capa de Max Pooling para reducir la dimensionalidad de los datos y, finalmente, una capa de Dropout para evitar el sobreajuste de la red.

Luego, el módulo de clasificación es responsable de determinar si un espectrograma corresponde a una persona con la enfermedad de Parkinson o no. Para conectar este módulo con la información obtenida de los bloques convolucionales anteriores, se utiliza una capa de aplanamiento (*Flatten*), que transforma los datos en un formato adecuado para su procesamiento por capas densas.

Este módulo de clasificación consiste en dos capas completamente conectadas (*Fully connected*), donde la primera utiliza la función de activación ReLU y la segunda utiliza la función de activación Softmax para generar la clasificación. Ambas capas están conectadas a través de una capa de Dropout, que ayuda a regularizar el modelo y evitar el sobreajuste durante el entrenamiento.

Respecto al entrenamiento como tal, se utiliza la función *StratifiedGroupKFold* de la librería *Scikit-learn* con la finalidad de dividir los datos en 10 conjuntos (*Fold*) de entrenamiento y validación verificando que los sujetos seleccionados (y sus espectrogramas asociados) para validación no se mezclen con los de entrenamiento. Una característica importante de esta función es que permite el trabajo con grupos, como se menciona con anterioridad, cada audio correspondiente a un paciente en específico es segmentado para generar espectrogramas de igual tamaño y por tanto, según la duración del audio de la persona podrá tener una mayor o menor cantidad de datos asociados. *StratifiedGroupKFold* considera dicha información y la división es realizada según grupos de datos, por ejemplo, si un sujeto es destinado a validación, todos su espectrogramas asociados serán considerados como tal.

Por otro lado, se verifica que sólo sujetos reales sean seleccionados para validación y la data aumentada sólo pueda ser incluida en el entrenamiento, dicha verificación se cumple anexando la data aumentada al entrenamiento luego de realizar la separación entre datos originales de entrenamiento y de validación.

Cada una de estas divisiones (*Fold*) se utiliza para realizar un entrenamiento de 100 épocas, es decir, se entrenará el modelo 10 veces con variaciones en los sujetos utilizados para validar, luego se calculará métricas promedio de dichos entrenamientos. Es importante mencionar que cada uno de estos entrenamientos es independiente del anterior, por tanto se reinicia los pesos del modelo antes de comenzar un nuevo entrenamiento. Además, los datos de entrenamiento son entregados al modelo de manera aleatoria por tanto no se sigue un mismo orden de sujetos a la hora de ajustar un modelo.

4. **Evaluación:** Ésta última sección consta del cálculo de distintas métricas y gráficas para la evaluación del modelo. Se comienza con los siguiente índices:

- **Accuracy:** Es una de las métricas más comunes para el estudio del comportamiento de modelos de *Machine Learning*, en español se puede traducir como exactitud e indica la razón de predicciones correctas respecto del total de predicciones. En la implementación de *Scikit-learn* se aplica la siguiente expresión:

$$accuracy(\hat{y}, y) = \frac{1}{n_{muestras}} \sum_{i=0}^{n_{muestras}-1} 1(\hat{y}_i = y_i) \quad (4.1)$$

Ésta ecuación realiza una sumatoria de los casos en que la predicción del modelo ( $\hat{y}$ ) y el valor real de la clasificación ( $y$ ) son iguales, es decir, el modelo acertó. Luego se divide entre la cantidad de pruebas realizadas. Si bien esta métrica entrega una idea general del comportamiento del modelo, no se recomienda juzgar si éste es bueno o malo respecto a la tarea para la que fue entrenado sólo considerando ésta medida dado que frente a conjuntos de datos de entrenamiento desbalanceado, es decir, más elementos de una clase que de otras, el **accuracy** puede ser muy alto por sólo detectar de buena manera la clase más abundante.

Cabe destacar que a la hora de reportar los resultados de este trabajo se entrega dos *accuracy* distintas, la primera es denominada *accuracy by windows* la cual evalúa la exactitud en la detección en base a una ventana de audio, es decir, en base a un sólo espectrograma, por otro lado se encuentra la denominada *accuracy by subject*, que entrega la exactitud de detección de la enfermedad de Parkinson por sujeto, es decir, considera todos los segmentos asociados a una persona en específico o en otras palabras, la grabación de voz completa.

De lo anterior, la métrica considerada en el análisis es *accuracy by subject* pues importa conocer si el modelo logra detectar la presencia de la enfermedad de Parkinson en base a la información completa de una persona y no solo en una ventana de señal.

- **Sensitivity:** Ésta medida también es conocida como *Recall* o tasa de verdaderos positivos, por tanto, para entender su funcionamiento es necesario conocer qué es un verdadero positivo (VP).

En el contexto de la medicina se utiliza dicho concepto para identificar a pacientes correctamente diagnosticados con una enfermedad, otros conceptos relacionados son:

- Falso positivo (FP): La persona no padece la enfermedad, sin embargo, la prueba a la que fue sometida indica que sí la padece.
- Verdadero negativo (VN): Similar al verdadero positivo, sólo que en éste caso, la persona no padece la enfermedad y la prueba indica que dicho diagnóstico.
- Falso negativo (FN): En este último caso, la persona padece la enfermedad pero la prueba indica que no la padece.

En el contexto de *Deep Learning*, dichos conceptos son utilizados de manera similar, sin embargo, en vez de una prueba para determinar si la persona padece o no la enfermedad es el modelo el que predice si la persona la padece y dicha respuesta es contrastada con la información real para determinar si se trata de un verdadero positivo o alguna de las otras clasificaciones. Por ejemplo, si el modelo determina que una persona padece la enfermedad de Parkinson y el dato está clasificado como tal entonces se considera como un verdadero positivo.

Considerando lo anterior, la sensibilidad se calcula como la cantidad de verdaderos positivos sobre el total de verdaderos positivos y falsos negativos, es decir:

$$sensitivity = \frac{VP}{VP + FN} \quad (4.2)$$

La sensibilidad se puede interpretar como el porcentaje de personas correctamente diagnosticadas con la presencia de la enfermedad de Parkinson (PD). El valor correspondiente a esta métrica se obtiene utilizando la librería *Imbalanced learn* [31], parte de *Scikit-learn*, específicamente la función **sensitivity\_specificity\_support**.

- **Specifity:** Es una métrica complementaria a la sensibilidad e indica el porcentaje de personas correctamente diagnosticadas con la ausencia de la enfermedad de Parkinson (HC), en otras palabras, la tasa de verdaderos negativos. Se calcula cómo:

$$specifity = \frac{VN}{VN + FP} \quad (4.3)$$

Y su valor se calcula utilizando la misma función de *imbalanced learn* mencionada anteriormente.

La especificidad, en conjunto con la sensibilidad, evidencian el desempeño del modelo en la tarea de clasificar un diagnóstico. Estas métricas ofrecen información precisa sobre la efectividad del modelo para determinar si una persona tiene o no la enfermedad de Parkinson. Se espera que ambas métricas tengan valores cercanos entre sí, lo que indica un equilibrio en el modelo y su capacidad para detectar tanto a las personas que padecen la enfermedad como a las que no. Valores distantes entre la sensibilidad y la especificidad pueden indicar que el modelo tiende a detectar mejor un caso por sobre el otro después del entrenamiento.

- **F1 score:** Esta medida es conocida como una mezcla entre dos métricas, la precisión y la sensibilidad, es útil cuando los datos de entrenamiento no están balanceados y por tanto puede ser un buen complemento para la *accuracy*. Según la librería *Scikit-learn* es calculada cómo:

$$F1 = \frac{2 \cdot VP}{(2 \cdot VP) + FN + FP} \quad (4.4)$$

- **MCC (Matthews correlation coefficient):** Esta métrica es ampliamente utilizada en problemas de clasificación binaria ya que mide la calidad de dicha tarea en una escala entre -1 y 1. Para su valor máximo

(1), el coeficiente representa una clasificación perfecta, por otro lado para su valor mínimo (-1) indica una clasificación inversa, mientras que en caso de tener un valor intermedio (0) indica una clasificación promedio [32]. En la implementación de *Scikit-learn* este coeficiente es calculado según la siguiente ecuación:

$$MCC = \frac{(VP \cdot VN) - (FP \cdot FN)}{\sqrt{(VP + FP)(VP + FN)(VN + FP)(VN + FN)}} \quad (4.5)$$

- **AUC:** Corresponde al área bajo la curva **ROC** (Receiver operating characteristic), esta curva cuenta con la tasa de falsos positivos en el eje X y la tasa de verdaderos positivos en el eje Y, ambos ejes entre los valores 0 y 1. Cuenta también con una marca en su diagonal que indica el comportamiento de una predicción aleatoria, por tanto, que la curva se mantenga por sobre dicha diagonal indica un comportamiento mejor al de una simple predicción aleatoria de la presencia o ausencia de la enfermedad.

Así, la métrica *AUC* es simplemente el cálculo del área bajo dicha curva *ROC* como se menciona con anterioridad, mientras más cercano a 1 su valor el modelo es considerado mejor en la tarea de la clasificación, pues indica que la gráfica tiende a su esquina superior izquierda, lo que a su vez indica una mayor tasa de verdaderos positivos frente a una menor tasa de falsos positivos.

En la implementación de *Scikit-learn*, el área es calculada con una aproximación trapezoidal.

Estas métricas son ampliamente utilizadas en la literatura y entregan un panorama bastante amplio sobre el comportamiento de la red frente al problema trabajado y por este motivo que son calculadas [33].

Es importante mencionar que dichas métricas son calculadas con el conjunto de validación pertenecientes a la misma base de datos utilizada para entrenar, mientras que para evidenciar la capacidad de generalización, sólo se calcula y reporta la exactitud del modelo (*accuracy*) frente a datos de prueba pertenecientes a otro conjunto de datos. Además, al momento de ser reportadas, se entrega el valor promedio entre las distintas instancias (*Fold*) de entrenamiento y la desviación estándar asociada a la métrica.

Luego, para acompañar esta información se obtiene las tres gráficas detalladas a continuación:

1. **Loss:** Esta gráfica se asocia con el comportamiento progresivo de la función de pérdida a medida que avanzan las épocas del entrenamiento. En este caso la función de pérdida utilizada es **CrossEntropyLoss**, parte del framework *Pytorch*.

Esta función consta de 2 pasos, en primer lugar se calcula las probabilidades asociadas al vector de predicciones utilizando *Softmax* para luego aplicar la función de pérdida *negative log-likelihood*, que mide principalmente qué tan cercanos son los valores de la predicción con los diagnósticos reales de las personas.

Idealmente, la pérdida disminuirá conforme avance el entrenamiento, indicando un aprendizaje del modelo a medida que las predicciones se acerquen más a los valores reales, lo que resultará en una menor pérdida. La gráfica compara las curvas de pérdida con los datos de entrenamiento y los datos de validación, mostrando el desempeño del modelo en ambos conjuntos de datos.

2. **Accuracy:** Con la finalidad de evidenciar cómo progresa la exactitud del modelo en la predicción de la presencia o ausencia de la enfermedad de Parkinson, se grafica el valor del *accuracy* luego de cada época de entrenamiento. Esta gráfica cuenta con la comparación entre dos curvas de manera similar a la anterior.

El conjunto entre la curva de pérdida y la curva de *accuracy* se suele llamar *curvas de aprendizaje* pues ilustran cómo la red aprende a realizar la tarea solicitada.

3. **ROC:** Corresponde a la gráfica Receiver operating characteristic mencionada anteriormente, es calculada utilizando la función *roc\_curve* de *Scikit-learn*.

Es importante mencionar que estas 3 gráficas incluyen tanto la curva que representa el promedio de la métrica calculada en las 10 iteraciones de entrenamiento como la desviación estándar obtenida de la misma forma.

Después de obtener estas gráficas se calcula 3 métricas que permiten evaluar el comportamiento del modelo frente a distintas situaciones:

1. **Generalización:** Para verificar el comportamiento frente a datos nuevos para el modelo, se calcula la métrica *accuracy* al probar con espectrogramas provenientes de una segunda base de datos, distinta a la utilizada para entrenar.
2. **Divergencia de ‘Kullback-Leibler’ (KL):** Esta métrica se utiliza con la finalidad de evidenciar cómo influyen las técnicas de aumento de data en la variabilidad de los datos y cómo su uso afecta a la capacidad del modelo de detectar la enfermedad de Parkinson.

La *divergencia de KL* mide la diferencia entre dos distribuciones de probabilidad [34], por lo que es aplicada entre la distribución de los datos originales y la distribución de los datos aumentados. Mientras mayor sea su valor indica que la distribución de datos aumentados tiene una mayor diferencia con la distribución de los datos originales y por tanto se logra aumentar la variabilidad en los sujetos. Por otro lado, mientras menor sea su valor, se puede interpretar que los sujetos aumentados son más parecidos a los sujetos originales.

Esta divergencia puede ser calculada con la siguiente expresión, considerando P y Q distribuciones de probabilidad:

$$KL(P|Q) = \sum_i P(i) \frac{P(i)}{Q(i)} \quad (4.6)$$

3. **Traza de la matriz de Covarianza:** Con la misma finalidad que la métrica anterior, se calcula la traza de la matriz de covarianza entre la distribución de los datos originales y de los aumentados, esto para quienes padecen o no la enfermedad de Parkinson de manera separada utilizando las funciones **cov** y **trace** de Numpy. En otras palabras, se obtiene una covarianza para personas HC y otra para PD, donde cada una compara datos originales con datos aumentados.

La matriz de covarianza es una matriz cuadrada que guarda los valores de la covarianza entre cada uno de los elementos de los vectores trabajados, por otro lado, la traza representa la varianza total de esta matriz y es calculada con la siguiente expresión:

$$T = \sum_{i=1}^p s_i^2 \quad (4.7)$$

Donde p es el orden de la matriz y  $s_i^2$  corresponde a los valores de la diagonal de la matriz de covarianza.

Finalmente, con el fin de evidenciar cómo se comporta la red al recibir datos aumentados se proyecta la respuesta de la última capa *fully connected*, previa al cálculo de probabilidades de la predicción, en un plano de dos dimensiones utilizando el algoritmo ‘t-distributed stochastic neighbor embeddings’ (t-SNE).

Este algoritmo se selecciona por su capacidad de preservar las relaciones locales entre los puntos en la visualización, lo que nos permite analizar de manera efectiva las relaciones establecidas por el modelo. Para realizar el cálculo de estas proyecciones se utiliza la función TSNE de la biblioteca *Scikit-learn*.

La visualización resultante cuenta con cuatro agrupaciones distintas: Los datos originales asociados a personas que padecen o no la enfermedad (PD y HC, respectivamente) y los datos aumentados de personas PD y HC. Estas agrupaciones nos permiten explorar cómo los datos aumentados se integran en el espacio de características aprendidas por el modelo.

## 4.3. Bases de datos

En el presente trabajo de memoria se utiliza dos bases de datos para el estudio de la enfermedad de Parkinson a través de la voz. El uso de dos conjuntos distintos tiene la finalidad de poder realizar la validación cruzada.

Es importante mencionar que en ambas bases de datos se tiene distintos tipos de grabaciones con distintas pruebas, sin embargo, para poder realizar una comparación directa con [20] se utiliza sólo la fonación /pa-ta-ka/.

1. **PD-GITA:** Ésta es una base de datos grabada en la Clínica Noel en Medellín, Colombia. Consta de las grabaciones de voz de un total de 100 personas, 50 de ellas personas con la enfermedad de Parkinson (PD) y las 50 restantes personas sin la enfermedad (HC) [35]. Entre las principales características de esta base de datos se encuentra la homogeneidad de sus registros dado que se encuentra equilibrado en género y edades. Otro punto importante de destacar es que todas las personas participantes de las grabaciones son hispanohablantes y cuenta con distintas pruebas, como vocales sostenidas, palabras y frases. Los datos fueron muestreados a una frecuencia de 44,1 kHz y una resolución de 16-bit en un ambiente controlado.
2. **PD-Neurovoz:** Es una base de datos de origen Español cuyos datos fueron recolectados por el Departamento de Otorrinolaringología y Neurología del hospital Gregorio Marañón en Madrid, España [36],[37]. Ésta cuenta con una cantidad de personas ligeramente menor que PD-GITA, específicamente 86 personas participantes, donde 44 de ellas padecen la enfermedad de Parkinson (PD) y las 42 restantes no lo padecen. Al igual que PD-GITA, los datos fueron muestreados con una frecuencia de 44,1 kHz y una resolución de 16-bit bajo condiciones ambientales controladas.

Cabe destacar que los archivos de audio provenientes de la base de datos PD-Neurovoz tienen una mayor duración que los de PD-GITA y por tanto, a pesar de ser menos sujetos, luego de la etapa de procesamiento donde las grabaciones de voz son segmentadas de manera equitativa, se genera más ventanas de señal que por consecuencia genera una mayor cantidad de espectrogramas con respecto a PD-GITA.

## 4.4. Implementación de aumento de data

A continuación se entrega un detalle de la implementación de cada una de las técnicas seleccionadas. A modo general, gran parte de ellas es aplicada en la etapa de procesamiento de datos, agregando un parámetro booleano llamado *want\_to\_augment* que permite verificar si se aplicará o no el aumento de datos.

En caso de querer aplicar alguna técnica, se procede al cálculo de una probabilidad que determina si la técnica será aplicada o no a un audio en específico, esto puede ser modificado con el fin de definir el porcentaje de señales de audio del conjunto de datos que serán aumentados. Por defecto está configurado en 1, es decir, se aumenta el 100 % de los datos, basándose en lo expuesto en [24].

Luego de aplicada la técnica al audio completo, se realiza la segmentación mencionada, la cual será realizada 2 veces la primera con la señal de audio original y la segunda con la señal de audio aumentada. La única excepción es la aplicación del enmascaramiento, el cual se procesa en la etapa de cálculo de espectrogramas, siguiendo una implementación similar al resto de técnicas, es decir, realizando las mismas validaciones y luego de ser aplicada se calcula tanto el espectrograma de la señal original como el de la señal aumentada.

Retomando lo mencionado en la sección 4.2, los datos aumentados se trabajan de forma separada a los datos originales hasta que estos últimos son separados en entrenamiento y validación. Esta forma de trabajar los datos aumentados también es útil para el cálculo de métricas asociadas a la diferencia entre datos reales y sintéticos como las gráficas t-SNE, divergencia y covarianza.

Otro punto importante es que las etiquetas de las imágenes originales se conservan de manera íntegra en las imágenes sintéticas, mientras que el ID del sujeto al que pertenece cada segmento es modificado con un *offset* de 1000 con el fin de diferenciar sujetos originales y sujetos aumentados. Por ejemplo, si el sujeto número 1 es aumentado, los espectrogramas originales están asociados al ID número 1, mientras que los espectrogramas aumentados están asociados al ID número 1001.

#### 4.4.1. Cambios de velocidad (speed change)

Esta técnica de aumento de data es aplicada utilizando la función *time\_stretch* de *Librosa*, calculando el valor de la tasa (*rate*) de cambio de la velocidad de manera aleatoria en una distribución uniforme en un rango entre 0.5 y 1.5 (ambos valores incluidos).

En la siguiente imagen se puede ver una comparación entre un audio original y su versión aumentada utilizando el parámetro *rate* = 0,5:

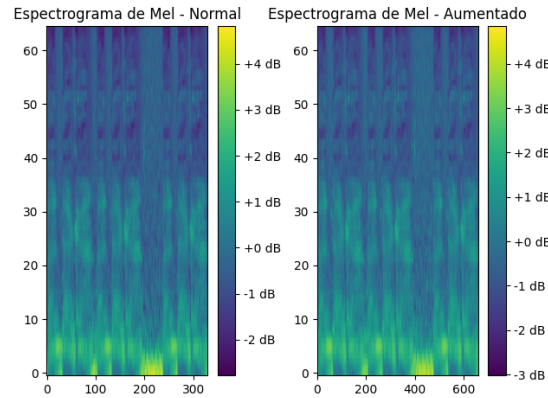


**Figura 4.1:** Cambio de velocidad con dilatación temporal. *rate* = 0,5

Se puede notar que el espectro no es modificado, sin embargo el tiempo es el doble que el anterior. En el caso de la segmentación realizada en el procesamiento de datos, dicha dilatación genera que el audio aumentado tendrá una mayor cantidad de segmentos asociados a un sujeto respecto a su versión original.

#### 4.4.2. Cambio del tono de voz (pitch change)

Según lo mencionado en la descripción de esta técnica, se utiliza la función *pitch\_shift* de *Librosa*. Para trabajar con aumentos o disminuciones de semitonos se configura el parámetro *bins\_per\_octave* en 12. Por otro lado, la cantidad de pasos que se aumentará o disminuirá se calcula de manera aleatoria en un rango de -4 a 4, es decir, se disminuirá o aumentará como máximo 4 semitonos. A continuación se muestra la comparación entre la señal original y un aumento de 4 semitonos y luego con una disminución de 4 semitonos.

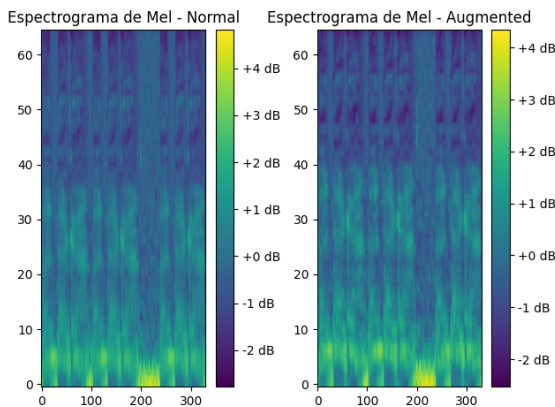


**Figura 4.2:** Cambio de tono de voz con 4 semitonos hacia arriba.

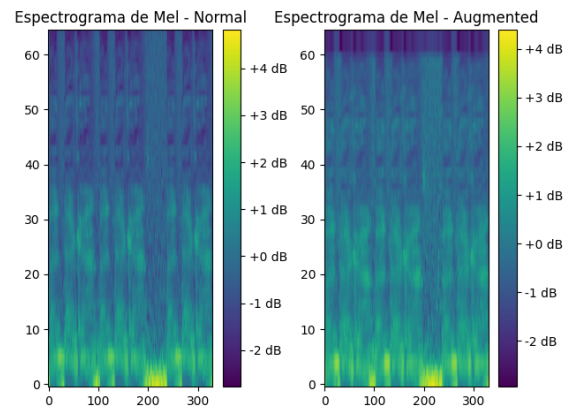


**Figura 4.3:** Cambio de tono de voz con 4 semitonos hacia abajo.

Por un lado, en la figura 4.2 se puede notar el aumento de la frecuencia sobretodo en las zonas de mayor potencia. Mientras que en la figura 4.3 se puede notar que además de disminuir las frecuencias también se agrega información de baja potencia en la parte superior para compensar la información faltante.

### 4.4.3. Enmascaramiento

Para este caso, solo se debe configurar dos parámetros, el primero relacionado al máximo número de bloques de frecuencia que serán enmascarados y el segundo es el número máximo de bloques temporales a enmascarar, con el fin de determinar parámetros óptimos para este problema en particular se realizará pruebas con los siguientes máximos:

1. Time = 6, Freq = 10
2. Time = 10, Freq = 14
3. Time = 14, Freq = 18
4. Time = 18, Freq = 22

La combinación de parámetros que entregue mejores resultados en el modelo será seleccionada para el computo de métricas y las conclusiones asociadas a esta técnica.

Una vez configurados los enmascaramientos, se aplican directamente al espectrograma de manera compuesta, es decir, se aplica en el tiempo y luego sobre el resultado se aplica la máscara en frecuencia.

Un ejemplo de estas máscaras aplicadas se muestra en la siguiente figura:

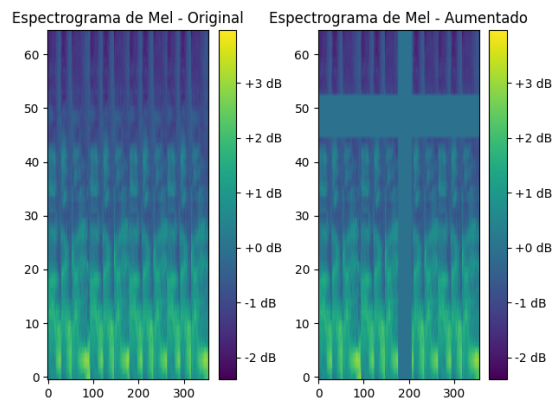


**Figura 4.4:** Enmascaramiento en frecuencia y en tiempo.

En este caso se utiliza un máximo de 100 para *timeMasking* y 18 para *frequencyMasking*, utilizar 100 para el parámetro temporal es netamente para ilustrar su comportamiento en un audio completo. Es importante mencionar que ambos enmascaramientos trabajan con ‘zero padding’, es decir, se reemplaza la información de la banda a enmascarar por ceros.

#### 4.4.4. Añadir ruido blanco gaussiano

Como se menciona en la descripción de esta técnica se trabaja con ruido blanco gaussiano aditivo. Para generar la señal de ruido se utiliza la función *Randn* del módulo *Random* de *Numpy* que permite generar un vector de números aleatorios según una distribución normal estándar. Este vector es escalado con un factor de ruido que es calculado considerando las SNR: 5 – 10 – 25 – 30dB para realizar distintas pruebas, estos valores son los utilizados en [24] para realizar una prueba similar. Por último, dicho vector de ruido es sumado a la señal de audio. A continuación se puede ver una comparación entre la señal original y versiones con ruido incorporado con 5 y 25 dB de SNR:

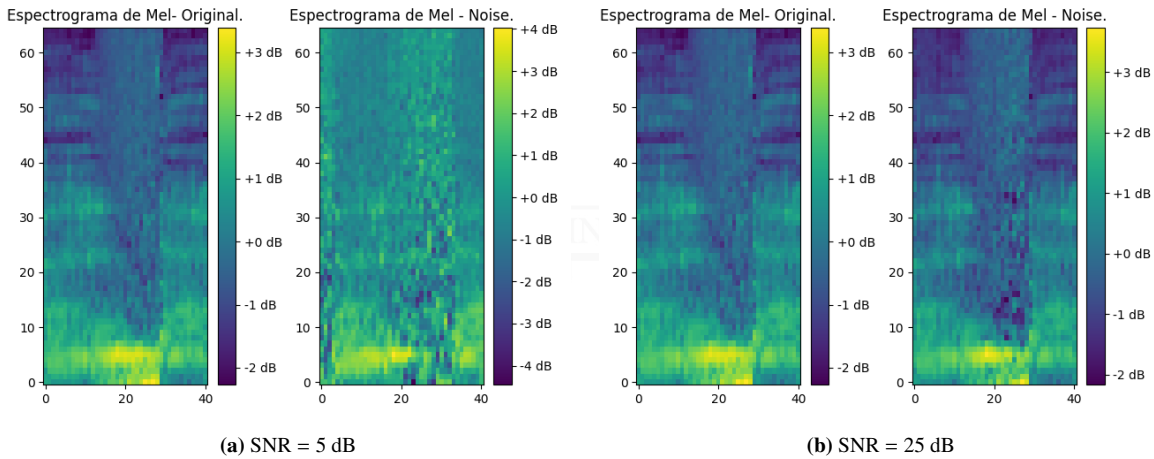


**Figura 4.5:** Comparación entre señal original y señal con ruido incorporado con distintas SNR.

Se puede notar que en el caso de 5dB de SNR la representación se vuelve bastante ruidosa y se puede llegar a perder la información importante relacionada a la patología, mientras que en el caso de 25 dB las representaciones resultan a simple vista iguales, dado que la potencia de ruido es bastante baja.

#### 4.4.5. Filtrado

La última técnica aplicada corresponde al filtrado con un filtro pasabajos utilizando la función *LowPassFilter* de *Audiomentations*. Para utilizar la función se configura tres parámetros básicos, el primero relacionado a la máxima frecuencia de corte, dicha configuración permite contar con una mayor variabilidad dado que cada vez que se aplique el filtrado se hará con una frecuencia de corte distinta. Se realiza cuatro pruebas con frecuencias de corte máxima distintas, los valores a utilizar son 1000 – 2000 – 3000 – 4000 Hz, de manera similar a las técnicas anteriores, las conclusiones asociadas se hacen considerando el mejor resultado. Con el fin de no generar frecuencias de corte que afecten la información vocal se define un parámetro asociado a la mínima frecuencia de corte que puede ser seleccionada igual a 500 Hz.

Por otro lado se configura un parámetro denominado **p** que determina el porcentaje de señales que serán aumentadas, dado que este control es realizado de forma manual en el código, **p** se configura igual a 1 con el fin que todas las señales de audio sean aumentadas por esta función.

La elección de un filtro pasabajos se justifica tomando en consideración que la frecuencia fundamental de la voz humana ronda los 200 Hz y por tanto las frecuencias superiores no se consideran significativas para determinar si una persona padece o no de la enfermedad de Parkinson, con esta técnica se busca que el modelo logre discriminar entre la información útil a baja frecuencia y el ruido de alta frecuencia. La siguiente figura muestra una comparación entre el espectrograma de una señal de audio original y su señal filtrada con una frecuencia de corte máxima igual a 1000 Hz y una mínima frecuencia de corte de 500 Hz como se menciona anteriormente:

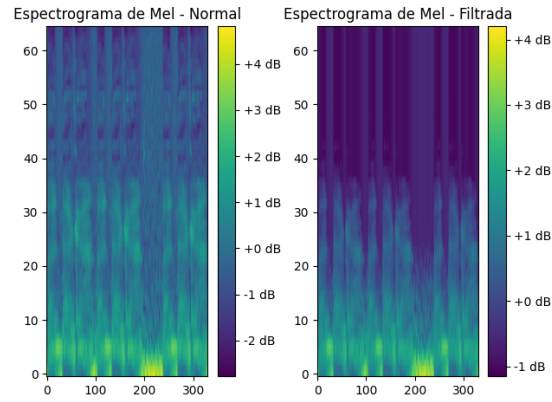


**Figura 4.6:** Audio filtrado utilizando una frecuencia de corte entre 500 y 1000 Hz.

Se observa un notorio corte a la altura aproximada de 38 Mels de frecuencia, se puede notar que considerar 1000 Hz puede servir para entrenar modelos que detecten sólo la información relevante a baja frecuencia.

## 5 | Experimentos y Resultados

En este capítulo se expone un detalle de todas las pruebas realizadas con cada una de las técnicas, para luego entregar un reporte general con los mejores resultados de cada una y comparaciones entre ellas. Además, utilizando las técnicas con mejores resultados se trabaja en entrenamientos con mezclas de técnicas para evaluar el comportamiento conjunto.

Cabe destacar que las principales pruebas son realizadas entrenando el modelo con el conjunto de datos PD-GITA, por tanto, el aumento de data se realiza sobre estas señales de audio y en caso de la validación cruzada se utiliza la base de datos PD-Neurovoz sin aumentar sus datos.

Por otro lado, con el fin de evaluar el comportamiento inverso, también se realiza pruebas en las que se entrena y aumenta la base de datos PD-Neurovoz y se realiza una validación cruzada con PD-Gita. En la sección 5.6 se entrega un reporte general de estas métricas.

Finalmente, se entrega el siguiente resumen de métricas de un modelo entrenado bajo las mismas condiciones pero sin data aumentada.

**Tabla 5.1:** Resumen de métricas obtenidas en [20]. Entrenado con PD-GITA

|                                    | PD-GITA         | PD-Neurovoz     |
|------------------------------------|-----------------|-----------------|
| <b>Accuracy by Subject ( % )</b>   | 80.8 $\pm$ 13.4 | 80.1 $\pm$ 16.7 |
| <b>Sensibility ( % )</b>           | 81.8            | 80.5            |
| <b>Specifity ( % )</b>             | 80.0            | 81.00           |
| <b>AUC</b>                         | 0.9             | 0.9             |
| <b>Cross validation acc. ( % )</b> | 47.7 $\pm$ 3.5  | 56.3 $\pm$ 2.6  |

### 5.1. Cambios de velocidad (*speed change*)

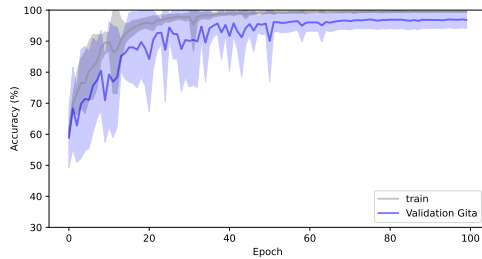
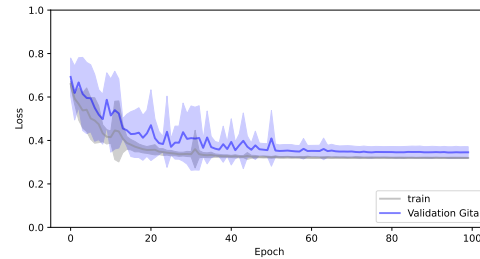
En el caso de esta técnica de aumento de data, dado que trabaja con una selección aleatoria del parámetro utilizado sólo se realiza una prueba para evaluar la estabilidad de las métricas entregadas, esta prueba consta de entrenar el modelo en reiteradas ocasiones utilizando el mismo conjunto de datos (no calcular nuevas imágenes) y comparar los resultados. Se espera que las métricas entreguen resultados cercanos, dado que en caso de tener una gran variabilidad indica que el aprendizaje del modelo depende en gran medida del orden en que los datos de entrenamiento son utilizados, lo cuál puede conllevar a repensar la forma de aplicar el aumento de data.

Es importante mencionar que el orden en que los datos son entregados en el entrenamiento se realiza de manera aleatoria, por lo que el modelo será ajustado en distinto orden cada vez que se ejecuta un entrenamiento. En la siguiente tabla se muestra un resumen de las cinco iteraciones realizadas.

**Tabla 5.2:** Resumen de métricas para speed change. Entrenado con PD-GITA

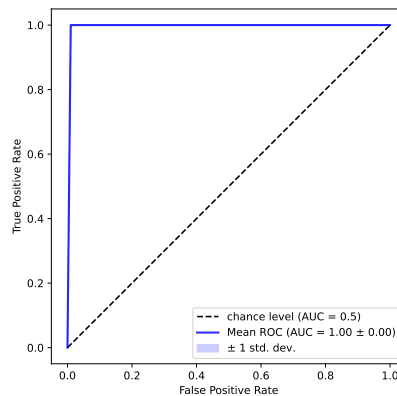
| N° Iteración | Accuracy by Windows | Accuracy by Subjects | Sensitivity | Specifity | F1 score | MCC  | AUC  |
|--------------|---------------------|----------------------|-------------|-----------|----------|------|------|
| 1            | 95.98 $\pm$ 3.44 %  | 99.00 $\pm$ 3.00 %   | 98.00 %     | 100.00 %  | 98.89 %  | 0.98 | 1.00 |
| 2            | 95.98 $\pm$ 2.68 %  | 97.89 $\pm$ 4.23 %   | 98.00 %     | 98.00 %   | 97.78 %  | 0.96 | 1.00 |
| 3            | 95.78 $\pm$ 3.98 %  | 99.00 $\pm$ 3.00 %   | 98.00 %     | 100.00 %  | 98.89 %  | 0.98 | 1.00 |
| 4            | 94.96 $\pm$ 3.67 %  | 98.00 $\pm$ 4.00     | 98.00 %     | 98.00 %   | 97.98 %  | 0.96 | 0.99 |
| 5            | 96.84 $\pm$ 2.76 %  | 97.89 $\pm$ 4.23 %   | 98.00 %     | 98.00 %   | 97.78 %  | 0.96 | 1.00 |

Dada la estabilidad de las métricas, se concluye que el entrenamiento es invariante al orden en que se entrega los datos. Por otro lado, de las iteraciones anteriores se entrega las curvas de aprendizaje de la iteración número 5. Considerando esta información, la iteración número 5 genera las siguientes curvas:

**Figura 5.1:** Speed change: Curva de accuracy durante un entrenamiento de 100 épocas.**Figura 5.2:** Speed change: Curva de pérdida durante un entrenamiento de 100 épocas.

El análisis clásico de estas curvas se relaciona con el sobre ajuste que se puede generar en el modelo, este no es el caso dado que tanto la curva de validación como la curva de entrenamiento tienen un crecimiento (o decrecimiento) proporcional y cercano, el caso de tener un sobre ajuste se da cuando la curva de validación se aleja de la curva de entrenamiento, lo que indica que el modelo aprende a detectar de buena forma los datos de entrenamiento pero no datos nuevos como los de validación. Por otro lado, se puede notar un entrenamiento poco prolijo, en que durante las primeras épocas presenta varias caídas en exactitud para luego estabilizarse.

Por otro lado, dicha iteración cuenta con una AUC igual a 1, lo que indica una muy buena clasificación, dicho valor se relaciona con la siguiente curva ROC:

**Figura 5.3:** Speed change: Curva ROC. Entrenado con PD-GITA.

Con el fin de entender cómo son considerados los datos aumentados dentro del entrenamiento se adjunta la

siguiente figura obtenida con el algoritmo *t-SNE*:

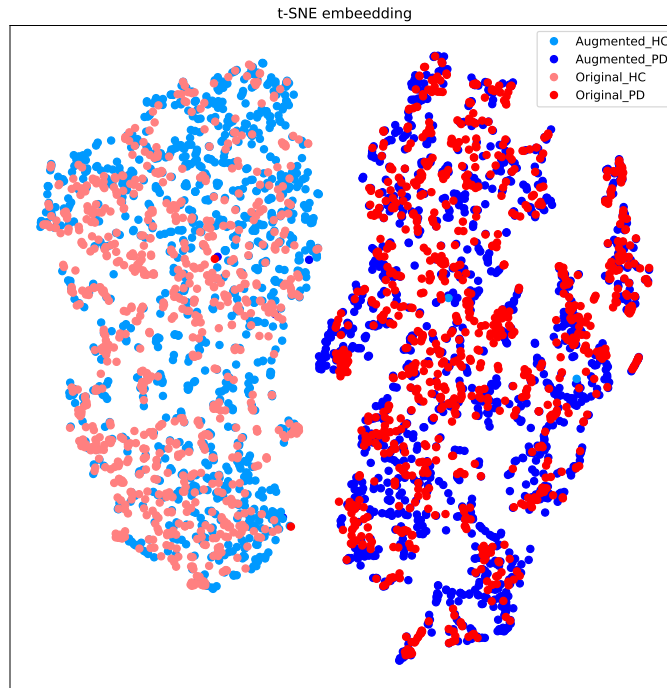


**Figura 5.4:** Speed change: T-SNE de un entrenamiento con PD-GITA.

La figura 5.4 logra evidenciar que el modelo sólo logra clasificar 2 grandes grupos de datos, el asociado a personas con la enfermedad de Parkinson y el de quienes no la padecen. Así, se concluye que la técnica *speed change* logra generar datos que se mezclan con los originales para aumentar de manera efectiva la data entregada en el entrenamiento, sin embargo, siguen una distribución muy similar a los datos originales, incluso en algunos casos tanto el dato aumentado como el original se superponen.

Respecto a los resultados de validación cruzada con la base de datos PD-Neurovoz se obtiene un *accuracy by subjects* de 54,423,81 %, donde se nota una caída de aproximadamente 45 % respecto a la exactitud de clasificación con la misma base de datos de entrenamiento y por tanto, se puede concluir que esta técnica de aumento de data no ayuda significativamente a la generalización. Sin embargo, si se compara la información con la tabla 5.1 se puede notar que el uso de esta técnica logra mejorar en aproximadamente un 10 % las métricas reportadas.

Otras métricas obtenidas desde este modelo son las asociadas a la variabilidad de los datos, sin embargo estas serán reportadas en la sección 5.6 en conjunto a las métricas de las demás técnicas y un modelo base.

## 5.2. Cambio del tono de voz (*pitch change*)

Utilizando la técnica de cambio en el tono de voz, se realiza dos pruebas, siendo la primera la misma prueba realizada con *speed change* con la diferencia que sólo se entrena el modelo en dos ocasiones. Por otro lado, con la segunda prueba se busca observar si la mejora en las métricas es progresiva o aumenta de manera exponencial incluso con pocos datos aumentados.

### 5.2.1. Prueba 1

A continuación se entrega una tabla que resume los resultados de las dos iteraciones de entrenamiento

**Tabla 5.3:** Resumen de métricas para pitch change. Entrenado con PD-GITA

| N° Iteración | Accuracy Windows     | Accuracy Subjects   | Sensitivity | Specifity | F1 score | MCC  | AUC  |
|--------------|----------------------|---------------------|-------------|-----------|----------|------|------|
| 1            | $87.47 \pm 10.30 \%$ | $92.96 \pm 7.93 \%$ | 91.83 %     | 94.00 %   | 92.70 %  | 0.86 | 0.97 |
| 2            | $87.47 \pm 9.66 \%$  | $89.87 \pm 9.22 \%$ | 85.83 %     | 94.00 %   | 88.98 %  | 0.81 | 0.97 |

En este caso se logra llegar a la misma conclusión anterior, el modelo es invariante al orden en que los datos son entregados. Por otro lado, se puede notar que utilizando *pitch change* se logran resultados ligeramente menores a los obtenidos en *speed change* si se compara con la tabla 5.2, sin embargo, de todas formas se presenta una mejora con respecto al modelo base presentado en la tabla 5.1.

Considerando la iteración número 2 se obtiene las siguientes gráficas:

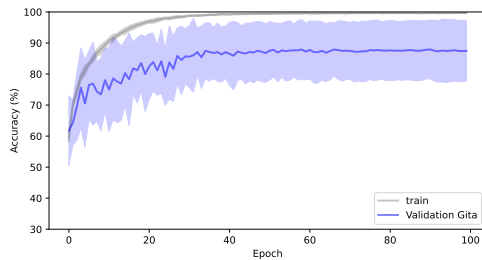


**Figura 5.5:** Pitch change: Curva de accuracy durante un entrenamiento de 100 épocas.

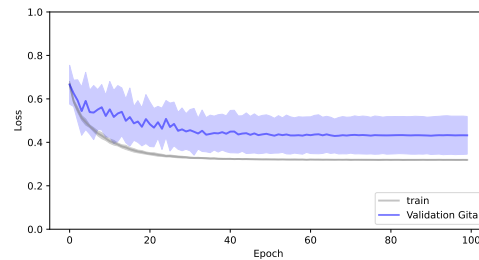


**Figura 5.6:** Pitch change: Curva de pérdida durante un entrenamiento de 100 épocas.

En este caso, se puede notar una mayor variabilidad en ambas curvas utilizando los datos de validación, además de una lejanía considerable respecto al entrenamiento. Esta discrepancia sugiere que la técnica de cambio en el tono de voz podría estar contribuyendo al sobreajuste del modelo. Además, se tiene la siguiente curva ROC:

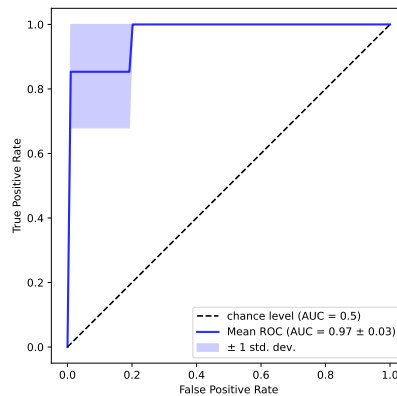


**Figura 5.7:** Pitch change: Curva ROC. Entrenado con PD-GITA

Se puede notar que a diferencia de speed change, con el cambio en el tono de voz no se logra una clasificación perfecta, sin embargo, no deja de ser un buen resultado de clasificación.

Luego, se presenta la siguiente gráfica t-SNE con la información de la data original y la data aumentada:

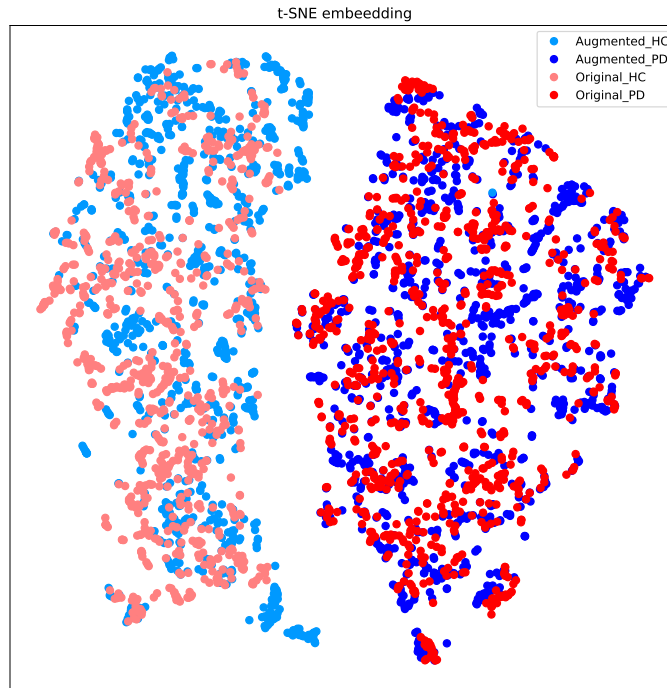


**Figura 5.8:** Pitch change: T-SNE de entrenamiento con PD-GITA

Nuevamente se logra apreciar sólo dos grandes grupos, lo que indica que la data aumentada se mezcla de buena manera con la data original. Finalmente, la validación cruzada reporta una exactitud de clasificación por sujeto de  $58,14 \pm 3,08 \%$ , una mejora de aproximadamente  $4 \%$  respecto de *speed change* y de un  $10 \%$  respecto del modelo base, lo que sin duda es un resultado alentador para la técnica de *pitch change* y abre la posibilidad de realizar pruebas mezclando esta técnica con otras para evaluar un comportamiento conjunto y solventar el posible sobreajuste.

### 5.2.2. Prueba 2

Esta prueba busca conocer si la técnica de aumento de data mejora el comportamiento del modelo de manera progresiva o bien, si esta mejora con el simple hecho de estar utilizando el aumento de data.

Para esto, se realizó un ajuste en el porcentaje de datos que es afectado por el aumento de data a un  $0,025 \%$ , lo que corresponde a 51 imágenes sintéticas para el conjunto de datos PD-GITA, que cuenta con un total de 1962 imágenes luego del procesamiento de datos, generando en total 2013 imágenes para el entrenamiento. Los resultados de esta prueba se exponen a continuación:

**Tabla 5.4:** Resumen de métricas para pitch change. Entrenado con PD-GITA y menor cantidad de datos aumentados.

|                          |                     |
|--------------------------|---------------------|
| <b>Accuracy Windows</b>  | 75.98 $\pm$ 8.87 %  |
| <b>Accuracy Subjects</b> | 81.76 $\pm$ 12.11 % |
| <b>Sensitivity</b>       | 83.83 %             |
| <b>Specifity</b>         | 80.00 %             |
| <b>F1 score</b>          | 81.83 %             |
| <b>MCC</b>               | 0.65                |
| <b>AUC</b>               | 0.88                |

Si se realiza una comparación directa con la tabla 5.1 se puede notar que no se presentan grandes mejoras por lo que se puede concluir a partir de estos resultados que la mejora en la clasificación del modelo está directamente relacionada con la cantidad adicional de datos que se incluyen a la hora del entrenamiento.

### 5.3. Enmascaramiento

Respecto a esta tercera técnica de aumento de data se realiza sólo una prueba, la cual se relaciona con entrenar y evaluar un modelo con distintos parámetros de enmascaramiento temporal y en frecuencia, es importante recordar que cada uno de ellos indica el máximo número de bloques consecutivos de tiempo o frecuencia serán enmascarados. Se realiza un total de 4 iteraciones cuyos resultados se resumen en la siguiente tabla:

**Tabla 5.5:** Resumen de métricas en prueba de parámetros para enmascaramiento. Entrenado con PD-GITA

| <b>Parámetros</b>       | <b>Accuracy Windows</b> | <b>Accuracy Subjects</b> | <b>Sensitivity</b> | <b>Specifity</b> | <b>F1 score</b> | <b>MCC</b> | <b>AUC</b> |
|-------------------------|-------------------------|--------------------------|--------------------|------------------|-----------------|------------|------------|
| <b>Time 6, Freq 10</b>  | 98.27 $\pm$ 1.61 %      | 99.00 $\pm$ 3.00 %       | 98.00 %            | 100.00 %         | 98.89 %         | 0.98       | 1.00       |
| <b>Time 10, Freq 14</b> | 97.31 $\pm$ 2.57 %      | 100.00 $\pm$ 0.00 %      | 100.00 %           | 100.00 %         | 100.00 %        | 1.00       | 1.00       |
| <b>Time 14, Freq 18</b> | 97.41 $\pm$ 2.92 %      | 100.00 $\pm$ 0.00 %      | 100.00 %           | 100.00 %         | 100.00 %        | 1.00       | 1.00       |
| <b>Time 18, Freq 22</b> | 95.07 $\pm$ 3.51 %      | 98.00 $\pm$ 4.00 %       | 98.00 %            | 98.00 %          | 97.98 %         | 0.96       | 1.00       |

Además, su validación cruzada se resume en la siguiente tabla:

**Tabla 5.6:** Resumen de métricas en prueba de validación cruzada para enmascaramiento. Entrenado con PD-GITA

| <b>Parámetros</b>       | <b>Accuracy Windows</b> | <b>Accuracy Subjects</b> |
|-------------------------|-------------------------|--------------------------|
| <b>Time 6, Freq 10</b>  | 45.43 $\pm$ 1.38 %      | 51.40 $\pm$ 1.45 %       |
| <b>Time 10, Freq 14</b> | 45.04 $\pm$ 1.00 %      | 52.56 $\pm$ 1.45 %       |
| <b>Time 14, Freq 18</b> | 43.40 $\pm$ 2.10 %      | 51.86 $\pm$ 2.16 %       |
| <b>Time 18, Freq 22</b> | 46.07 $\pm$ 3.21 %      | 50.47 $\pm$ 2.03 %       |

Dada la información de las tablas 5.5 y 5.6 se define que los parámetros óptimos para este problema son 10 bloques consecutivos para el caso de enmascaramiento temporal y 14 bloques para el relacionado a la frecuencia. Por tanto, a continuación se muestra las distintas gráficas obtenidas con dichas variables:

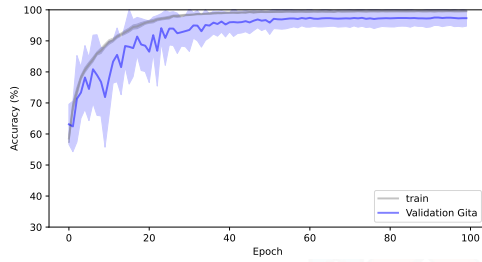


**Figura 5.9:** Enmascaramiento: Curva de accuracy durante un entrenamiento de 100 épocas.

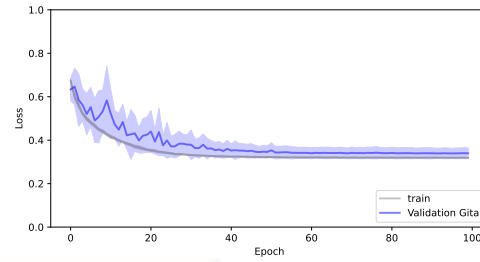


**Figura 5.10:** Enmascaramiento: Curva de pérdida durante un entrenamiento de 100 épocas.

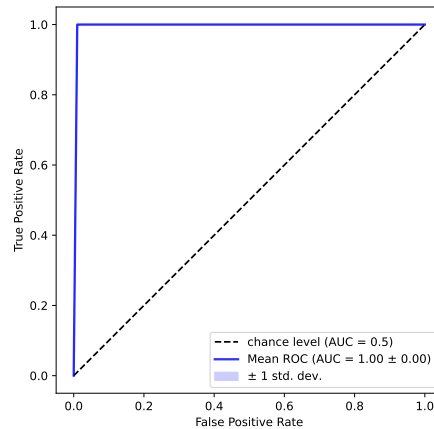


**Figura 5.11:** Enmascaramiento: Curva ROC.  
Entrenado con PD-GITA.

De las figuras 5.9, 5.10 y 5.11 se concluye que el uso de los enmascaramientos afecta positivamente en la clasificación de la enfermedad de Parkinson. Por su lado, las curvas de *accuracy* y pérdida representan un buen aprendizaje de la red, mientras que la curva ROC evidencia una clasificación perfecta. Por otro lado, si se evalúa la habilidad de generalización del modelo, comparando la información de las tablas 5.1 y 5.6 se puede notar que el enmascaramiento genera una leve mejora en la exactitud de clasificación.

Luego, se muestra el resultado obtenido por el algoritmo T-SNE en función de los datos originales y los datos aumentados con esta técnica:

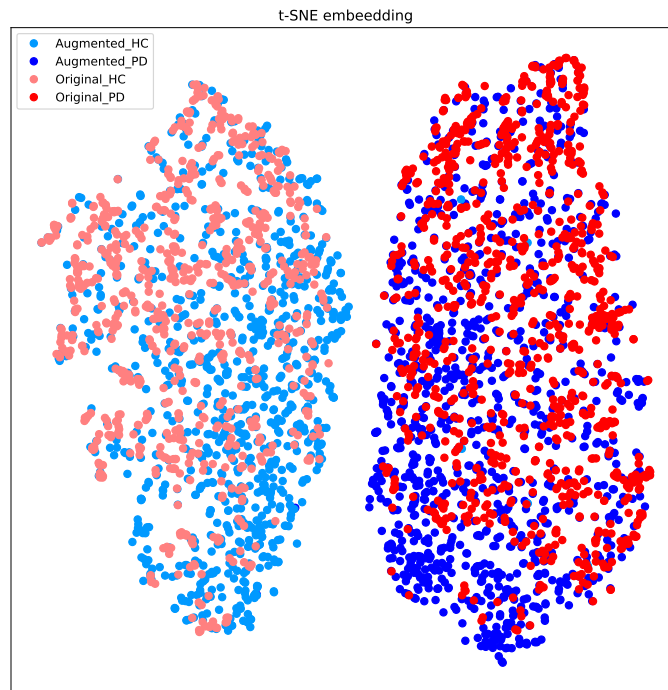


**Figura 5.12:** Enmascaramiento: T-SNE de entrenamiento con PD-GITA

De manera similar a las técnicas anteriores, esta técnica logra generar datos nuevos que se mezclan de buena forma con los datos originales, además, a simple vista se puede decir que el enmascaramiento logra abarcar de mejor manera ‘nuevos espacios’ que la data original no cubre, sobretodo hacia la zona inferior.

## 5.4. Añadir ruido blanco gaussiano

De manera similar a lo realizado con la técnica de enmascaramiento, en este caso se realiza una prueba para evaluar qué SNR es el óptimo en este problema. A continuación se reporta las diversas métricas calculadas

**Tabla 5.7:** Resumen de métricas en prueba de parámetro para ruido. Entrenado con PD-GITA

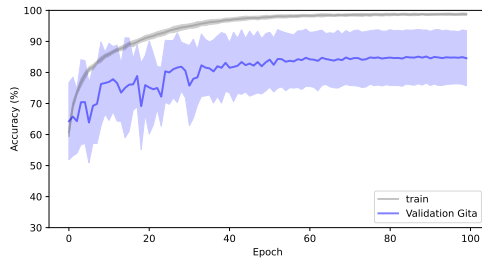
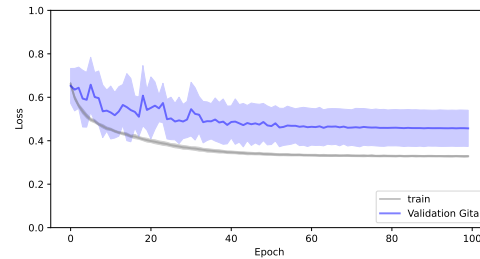
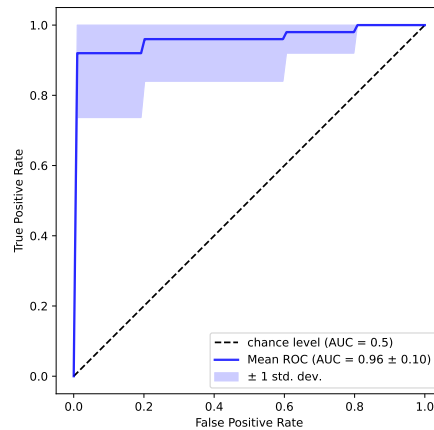
| SNR (dB)  | Accuracy Windows   | Accuracy Subjects   | Sensitivity | Specifity | F1 score | MCC  | AUC  |
|-----------|--------------------|---------------------|-------------|-----------|----------|------|------|
| <b>5</b>  | 81.36 $\pm$ 9.33 % | 86.98 $\pm$ 11.00 % | 90.33 %     | 84.00 %   | 87.38 %  | 0.75 | 0.94 |
| <b>10</b> | 84.58 $\pm$ 8.90 % | 91.98 $\pm$ 9.81 %  | 92.33 %     | 92.00 %   | 91.87 %  | 0.85 | 0.96 |
| <b>25</b> | 94.60 $\pm$ 3.49 % | 98.09 $\pm$ 3.82 %  | 98.33 %     | 98.00 %   | 98.18 %  | 0.96 | 1.00 |
| <b>30</b> | 97.91 $\pm$ 1.66 % | 100.00 $\pm$ 0.0 %  | 100.00 %    | 100.00 %  | 100.00 % | 1.00 | 1.00 |

**Tabla 5.8:** Resumen de métricas en prueba de validación cruzada para ruido. Entrenado con PD-GITA

| SNR (dB) | Accuracy Windows    | Accuracy Subjects   |
|----------|---------------------|---------------------|
| 5        | $41.76 \pm 1.56 \%$ | $45.70 \pm 2.55 \%$ |
| 10       | $46.42 \pm 2.24 \%$ | $49.77 \pm 2.64 \%$ |
| 25       | $45.19 \pm 2.04 \%$ | $49.88 \pm 2.57 \%$ |
| 30       | $45.74 \pm 2.75 \%$ | $50.12 \pm 3.54 \%$ |

Desde esta información se puede decir que la mejor SNR sería de 30 dB dado que logra una clasificación perfecta, sin embargo, se debe considerar que una SNR alta indica que la potencia del ruido es baja y por tanto, se puede estar generando duplicados exactos de la señal de audio. Dicho caso se aborda en una de las pruebas de la sección 5.6. Por otro lado, se puede concluir que mientras menor sea la potencia del ruido, mejor es el resultado, pero con el riesgo mencionado anteriormente.

Considerando lo anterior, se define una  $SNR = 10$  como la mejor para este caso, la principal razón de esta conclusión es su notable mejora en la exactitud de clasificación de sujetos en validación cruzada respecto al uso de  $SNR = 5dB$  y  $SNR = 25dB$ , donde se mejora en aproximadamente 4 % respecto al primer caso y la diferencia con el segundo caso es de apenas un 0,11 %. Con dicha SNR se obtiene las siguientes gráficas:

**Figura 5.13:** Ruido: Curva de accuracy durante un entrenamiento de 100 épocas.**Figura 5.14:** Ruido: Curva de pérdida durante un entrenamiento de 100 épocas.**Figura 5.15:** Ruido: Curva ROC. Entrenado con PD-GITA.

Además, se obtiene la siguiente gráfica T-SNE:

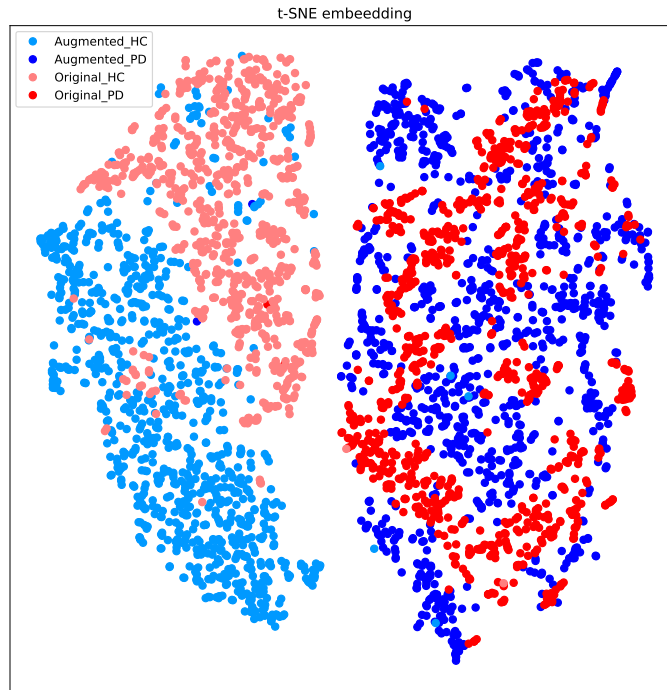


**Figura 5.16:** Ruido: T-SNE de entrenamiento con PD-GITA

Respecto a las curvas de exactitud y pérdida (figuras 5.13 y 5.14 respectivamente), se puede notar un problema similar al de pitch change, en el que se muestra una alta variabilidad y una tendencia a sobre ajuste del modelo. Por otro lado, respecto a la figura 5.16, se evidencia una separación entre los datos originales y aumentados de personas sin la enfermedad de Parkinson (HC), esto quiere decir que el modelo tiende a diferenciar los datos sintéticos de los originales en dicho caso.

## 5.5. Filtrado

Por último, para el caso de el aumento de data utilizando un filtro pasabajos, se realiza una prueba para determinar la mejor frecuencia de corte máxima que podrá ser escogida de manera aleatoria por la función utilizada. A continuación se adjunta una tabla con los resultados:

**Tabla 5.9:** Resumen de métricas en prueba de parámetro para filtrado. Entrenado con PD-GITA

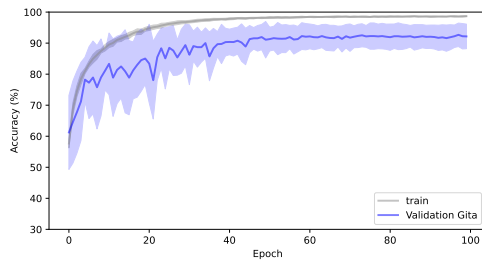
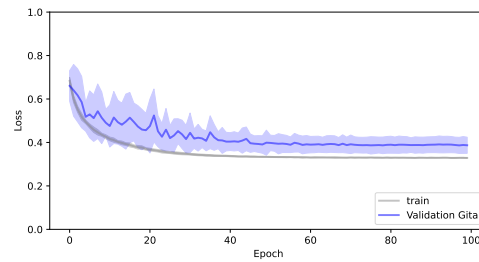
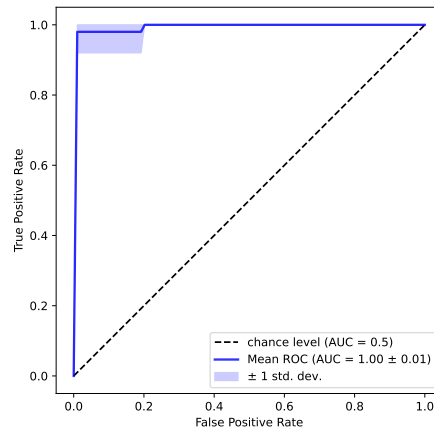
| Máxima $f_c$ (Hz) | Accuracy Windows    | Accuracy Subjects   | Sensitivity | Specifity | F1 score | MCC  | AUC  |
|-------------------|---------------------|---------------------|-------------|-----------|----------|------|------|
| <b>1000</b>       | $89.72 \pm 6.06 \%$ | $95.89 \pm 6.74 \%$ | 98.00 %     | 94.00 %   | 95.98 %  | 0.92 | 1.00 |
| <b>2000</b>       | $89.65 \pm 5.65 \%$ | $96.00 \pm 6.63 \%$ | 98.00 %     | 94.00 %   | 96.31 %  | 0.93 | 1.00 |
| <b>3000</b>       | $94.60 \pm 5.19 \%$ | $97.89 \pm 4.23 \%$ | 98.00 %     | 98.00 %   | 97.78 %  | 0.96 | 1.00 |
| <b>4000</b>       | $90.17 \pm 8.39 \%$ | $94.98 \pm 6.74 \%$ | 93.83 %     | 96.00 %   | 94.75 %  | 0.90 | 0.98 |
| <b>5000</b>       | $92.17 \pm 2.89 \%$ | $96.09 \pm 4.79 \%$ | 96.33 %     | 96.00 %   | 96.16 %  | 0.93 | 1.00 |

Además, se considera la siguiente tabla con la información respecto a la validación cruzada:

**Tabla 5.10:** Resumen de métricas en prueba de validación cruzada para filtrado. Entrenado con PD-GITA

| Máxima $f_c$ (Hz) | Accuracy Windows    | Accuracy Subjects   |
|-------------------|---------------------|---------------------|
| <b>1000</b>       | $48.58 \pm 2.76 \%$ | $51.40 \pm 3.60 \%$ |
| <b>2000</b>       | $48.27 \pm 2.08 \%$ | $51.74 \pm 2.51 \%$ |
| <b>3000</b>       | $46.96 \pm 2.94 \%$ | $50.35 \pm 4.41 \%$ |
| <b>4000</b>       | $44.48 \pm 2.10 \%$ | $46.98 \pm 4.91 \%$ |
| <b>5000</b>       | $48.58 \pm 2.21 \%$ | $52.91 \pm 3.72 \%$ |

De las tablas anteriores se puede concluir que el mejor parámetro para este problema es  $f_c = 5000$  debido a que cuenta con una mejor *accuracy* de validación cruzada, sin embargo todas las métricas reportadas son bastante similares. Luego, se reporta las siguientes gráficas asociadas a dicha frecuencia de corte máxima:

**Figura 5.17:** Filtrado: Curva de accuracy durante un entrenamiento de 100 épocas.**Figura 5.18:** Filtrado: Curva de pérdida durante un entrenamiento de 100 épocas.**Figura 5.19:** Filtrado: Curva ROC. Entrenado con PD-GITA.

De las figuras se puede notar resultados bastante similares a las técnicas anteriores, e incluso desde la figura 5.19 se puede interpretar una clasificación casi perfecta, mientras que en las figuras 5.17 y 5.18 se puede notar un buen aprendizaje del modelo pero con una mayor variabilidad respecto a *Speed change* o enmascaramiento. Respecto a su comportamiento de datos proyectado utilizando T-SNE se tiene:

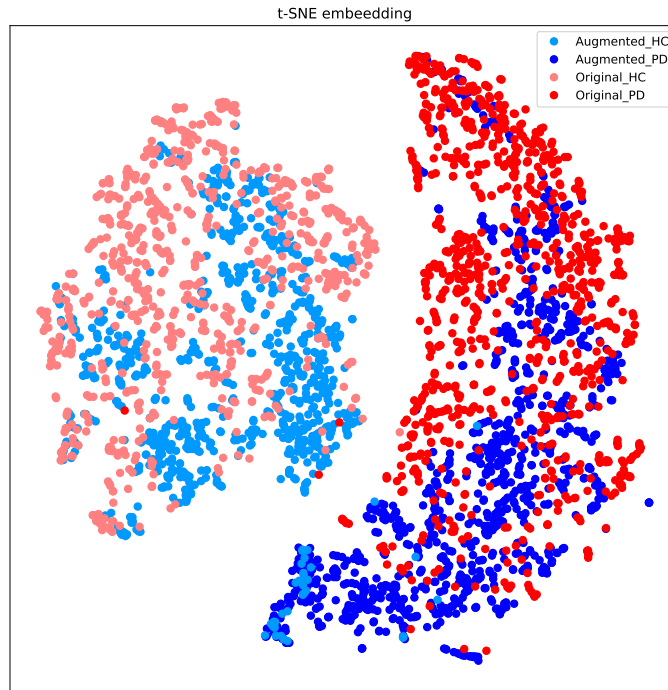


**Figura 5.20:** Filtrado: T-SNE de entrenamiento con PD-GITA

En la figura 5.20 se puede notar que ocurre una diferencia bastante notoria respecto a las anteriores gráficas T-SNE, donde en la parte inferior central, algunos datos aumentados de personas sin la enfermedad de Parkinson son confundidos por la red como parte del grupo de personas que sí padecen la enfermedad, utilizando las técnicas anteriores esto ocurría pero se trataba de casos aislados, sin embargo, utilizando filtrado es una cantidad no menor la que es confundida. Por otro lado, los datos aumentados para casos HC y PD no se mezclan del todo con la información original, si bien se logra contar con sólo 2 grupos, la data sintética expande la distribución de los datos.

## 5.6. Resumen de resultados

En esta sección se realiza un resumen de la información entregada previamente y se compara los resultados con el modelo base reportado en [20], cuyo entrenamiento es realizado sin data sintética. Otra de las pruebas detalladas en esta sección es respecto a la variabilidad que entrega cada una de las técnicas de aumento de data, esto se realiza comparando la divergencia y la covarianza de cada una de las métricas con una prueba de datos duplicados. Por último se entrega un reporte asociado al entrenamiento con la base de datos PD-Neurovoz utilizando las distintas técnicas de aumento de data y su comportamiento al validar de manera cruzada con PD-GITA.

### 5.6.1. Reporte 1

La información sobre las métricas obtenidas con cada una de las técnicas de aumento de data utilizadas se encuentra resumida en la siguiente tabla:

**Tabla 5.11:** Resumen de accuracy en validación con ambas bases de datos. Entrenado con PD-GITA

| Validación con:  | GITA               |                     | Neurovoz           |                    |
|------------------|--------------------|---------------------|--------------------|--------------------|
| Técnica          | Accuracy Windows   | Accuracy Subjects   | Accuracy Windows   | Accuracy Subjects  |
| Original         | 73.71 $\pm$ 8.52 % | 80.76 $\pm$ 13.38 % | 42.43 $\pm$ 3.08 % | 47.67 $\pm$ 3.45 % |
| Pitch Change     | 87.47 $\pm$ 9.66 % | 89.87 $\pm$ 9.22 %  | 55.72 $\pm$ 2.28 % | 58.14 $\pm$ 3.08 % |
| Speed Change     | 96.84 $\pm$ 2.76 % | 97.89 $\pm$ 4.23 %  | 51.90 $\pm$ 3.07 % | 54.42 $\pm$ 3.81 % |
| Enmascaramiento  | 97.31 $\pm$ 2.57 % | 100.00 $\pm$ 0.00 % | 45.04 $\pm$ 1.00 % | 52.56 $\pm$ 1.45 % |
| Añadir ruido     | 84.58 $\pm$ 8.90 % | 91.98 $\pm$ 9.81 %  | 46.42 $\pm$ 2.24 % | 49.77 $\pm$ 2.64 % |
| Filtro pasabajos | 92.17 $\pm$ 3.96 % | 96.09 $\pm$ 4.79 %  | 48.58 $\pm$ 2.21 % | 52.91 $\pm$ 3.72 % |

Se destaca que la fila con la información original corresponde al entrenamiento sin aumento de data. Respecto a la información de esta tabla y los comportamientos expuestos anteriormente se define a *pitch change* como la técnica de aumento de data que más aporta en la generalización del modelo pues presenta una mejora sobre el 10 % respecto al entrenamiento original.

Por otro lado, también se considera a *speed change*, enmascaramiento y filtrado como buenas alternativas dado que además presentan grandes mejoras en la validación con la base de datos de entrenamiento.

Con respecto a esta conclusión, en la sección 5.7 se realiza dos pruebas con la mezcla de algunas de estas técnicas. Por último, la técnica de incorporación de ruido se considera una opción que si mejora respecto al modelo base pero no logran resultados tan significativos que justifique su uso, aunque debe ser evaluado cómo afecta en la variabilidad de los datos para juzgar su aporte realmente.

### 5.6.2. Reporte 2

Como se ha mencionado anteriormente, se busca evaluar el comportamiento de duplicar el conjunto de datos sin ninguna técnica de aumento de data y cómo afecta la variabilidad de los nuevos datos en la generalización del modelo. Para esto, se realiza dos pruebas de duplicados, la primera con *speed change* con una tasa igual a 1, lo que indica que no se modificará la velocidad del audio. La segunda se realiza con la incorporación de ruido pero con un factor de 0, es decir, el ruido no afecta en la señal y por tanto se duplica los datos originales sin variaciones. Los resultados se reportan en la siguiente tabla:

**Tabla 5.12:** Resumen de métricas en caso de datos duplicados. Entrenado con PD-GITA

| Validación con: | GITA                |      |                     |                     |                  |                  | Neurovoz           |
|-----------------|---------------------|------|---------------------|---------------------|------------------|------------------|--------------------|
| Técnica         | Accuracy Subjects   | AUC  | Divergencia HC      | Divergencia PD      | Covarianza HC    | Covarianza PD    | Accuracy Subjects  |
| Añadir ruido    | 100.00 $\pm$ 0.00 % | 1.00 | -801.50 $\pm$ 43.69 | -799.49 $\pm$ 43.86 | 11.14 $\pm$ 2.15 | 18.33 $\pm$ 3.44 | 52.79 $\pm$ 2.50 % |
| Speed Change    | 100.00 $\pm$ 0.00 % | 1.00 | -258.43 $\pm$ 16.94 | -271.66 $\pm$ 15.60 | 11.02 $\pm$ 2.25 | 18.40 $\pm$ 1.71 | 49.53 $\pm$ 1.49 % |

En ambas se puede notar que se logra resultados muy buenos con la base de datos de entrenamiento, pero al momento de realizar validación cruzada, los resultados disminuyen considerablemente. Estas pruebas definen un estándar para juzgar el comportamiento de las técnicas utilizadas, en caso de superar o no este umbral, entrega información relevante si la técnica en cuestión aporta un beneficio significativo más allá de simplemente duplicar los datos de entrenamiento.

Las métricas de divergencia KL y de la traza de la matriz de covarianza entregan información relevante para comparar el comportamiento de la distribución de los datos al utilizar técnicas de aumento de data y la variabilidad que ellas entregan. La siguiente figura muestra la distribución de los datos utilizando T-SNE:

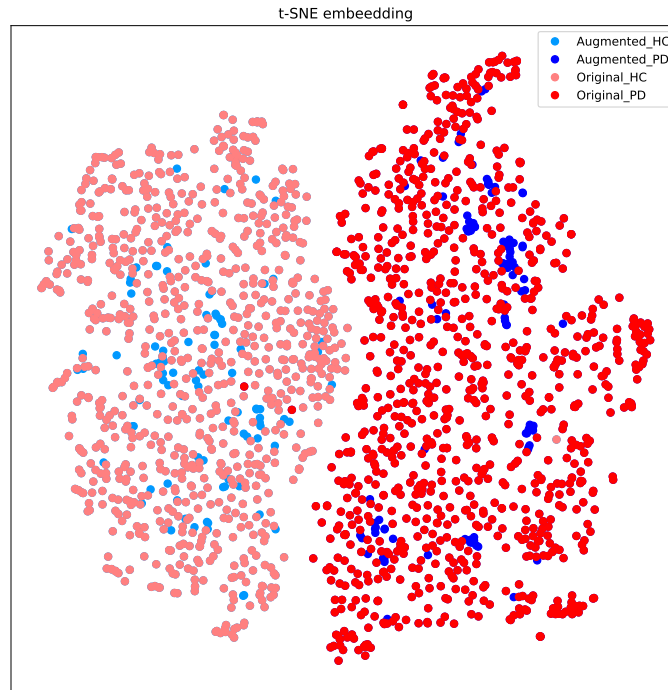


**Figura 5.21:** Duplicado: T-SNE de un entrenamiento con PD-GITA

Se puede notar una superposición casi total de los datos originales y aumentados salvo por algunas excepciones, evidenciando de manera gráfica el duplicado de los datos. Para evaluar cómo se comportan las distintas técnicas de aumento de datos cuando a la variabilidad que le entregan a los datos de entrenamiento se elabora la siguiente tabla resumen:

**Tabla 5.13:** Comparación de métricas de distintas técnicas de aumento de datos con respecto a datos duplicados. Entrenado con PD-GITA

| Técnica                 | Divergencia         |                     | Covarianza       |                  | GITA                | Neurovoz           |
|-------------------------|---------------------|---------------------|------------------|------------------|---------------------|--------------------|
|                         | HC                  | PD                  | HC               | PD               | Accuracy Subjects   | Accuracy Subjects  |
| <b>Duplicados</b>       | -801.50 $\pm$ 43,69 | -799.49 $\pm$ 43,86 | 11.14 $\pm$ 2.15 | 18.33 $\pm$ 3.44 | 100.00 $\pm$ 0.00 % | 52.79 $\pm$ 2.50 % |
| <b>Pitch Change</b>     | 12.84 $\pm$ 2.23    | 8.72 $\pm$ 1.46     | 26.21 $\pm$ 4.66 | 29.33 $\pm$ 6.95 | 89.87 $\pm$ 9.22 %  | 58.14 $\pm$ 3.08 % |
| <b>Speed Change</b>     | 5.66 $\pm$ 1.03     | 1.53 $\pm$ 1.16     | 15.09 $\pm$ 3.87 | 21.80 $\pm$ 5.81 | 97.89 $\pm$ 4.23 %  | 54.42 $\pm$ 3.81 % |
| <b>Enmascaramiento</b>  | 0.39 $\pm$ 5.34     | -1.34 $\pm$ 4,23    | 13.09 $\pm$ 1.47 | 13.41 $\pm$ 1.56 | 100.00 $\pm$ 0.00 % | 52.56 $\pm$ 1.45 % |
| <b>Añadir ruido</b>     | 25.61 $\pm$ 8.74    | 18.77 $\pm$ 7.39    | 11.98 $\pm$ 1.73 | 15.36 $\pm$ 1.95 | 91.98 $\pm$ 9.81    | 49.77 $\pm$ 2.64 % |
| <b>Filtro pasabajos</b> | 22.65 $\pm$ 10.56   | 24.19 $\pm$ 9.74    | 14.51 $\pm$ 2.49 | 19.96 $\pm$ 3.39 | 96.09 $\pm$ 4.79 %  | 52.91 $\pm$ 3.72 % |

Respecto a la tabla 5.13 se puede concluir que entre mayor la covarianza, es decir, una mayor variabilidad en los datos de entrenamiento debido al aumento realizado, se mejora en la generalización del modelo pero se ‘penaliza’ en la exactitud de la validación con el mismo conjunto de datos. Tal es el caso de pitch change, que logra una mejora de aproximadamente 10 % en la exactitud de validación con PD-Neurovoz y que además tiene una mayor variabilidad en sus datos.

Por otro lado, la técnica de añadir ruido y el modelo con datos duplicados poseen una covarianza similar y menor que el resto de técnicas, al evaluar sus métricas de *accuracy* se puede notar además que sus rendimientos se

encuentran por debajo de las demás técnicas en cuanto a la generalización. Estos hallazgos sugieren que la capacidad de generalización de un modelo está estrechamente relacionada con la varianza que entrega la técnica de aumento de datos utilizada en sus datos de entrenamiento.

Respecto a la divergencia, la técnica de enmascaramiento presenta una divergencia bastante pequeña, lo que indica que la técnica genera datos muy similares a los originales, esto tiene sentido ya que la técnica no altera el audio como tal, sino que sólo se cubre ciertas secciones de su espectrograma. En el caso de añadir ruido al audio y filtro pasabajos, se presenta una alta divergencia, que indica una diferencia en la distribución de datos entre los originales y los aumentados bastante grande, esto principalmente a la naturaleza de cada una de estas técnicas pues modifican de gran forma el audio original.

Se concluye que la divergencia sólo logra entregar información relevante sobre la distribución de los datos y cómo las distintas técnicas entregan datos más o menos similares a los originales, sin embargo, no se logra definir una relación con como afecta dicha distribución al comportamiento general del modelo. Por el contrario, la covarianza si logra una relación de ese estilo e indica de buena forma qué tan variados son los datos entre los originales y los aumentados.

### 5.6.3. Reporte 3

Respecto al uso de PD-Neurovoz como la base de datos para entrenamiento se tiene la siguiente tabla resumen con las métricas asociadas a cada una de las técnicas utilizadas:

**Tabla 5.14:** Comparación de métricas en las distintas técnicas de aumento de data. Entrenado con PD-Neurovoz

| Técnica                 | Neurovoz            | GITA               |
|-------------------------|---------------------|--------------------|
|                         | Accuracy Subjects   | Accuracy Subjects  |
| <b>Original</b>         | 78.89 $\pm$ 19.20 % | 56.20 $\pm$ 2.56 % |
| <b>Pitch change</b>     | 81.39 $\pm$ 12.99 % | 46.80 $\pm$ 5.36 % |
| <b>Speed change</b>     | 98.89 $\pm$ 3.33 %  | 47.50 $\pm$ 2.80 % |
| <b>Enmascaramiento</b>  | 100.00 $\pm$ 0.00 % | 57.00 $\pm$ 2.68   |
| <b>Añadir ruido</b>     | 84.72 $\pm$ 15.64 % | 61.10 $\pm$ 2.30 % |
| <b>Filtro pasabajos</b> | 92.92 $\pm$ 11.29 % | 52.50 $\pm$ 2.80 % |

De la tabla anterior se desprende que las técnicas de aumento de data contribuyen al buen rendimiento del modelo pero sólo en la misma base datos, un comportamiento similar al entrenar y validar con PD-GITA. Sin embargo, en este caso, al validar de manera cruzada se obtiene resultados que disminuyen respecto a un entrenamiento sin data aumentada, un comportamiento contrario al expuesto en el reporte 1. Los casos de excepción son enmascaramiento e inclusión de ruido, que en el caso anterior eran justamente las técnicas que menor mejora presentaban.

Este resultado genera la inquietud de saber qué característica relacionada al tipo de datos trabajados afecta en el rendimiento de una técnica por sobre otra y cómo poder determinar la técnica óptima para cada base de datos usada en el entrenamiento. Al respecto, se considera como posible trabajo futuro el investigar la relación antes mencionada.

Dado que el mejor rendimiento de generalización es obtenido utilizando la técnica de agregar ruido se muestra sus curvas de aprendizaje y ROC asociadas a ella a continuación:

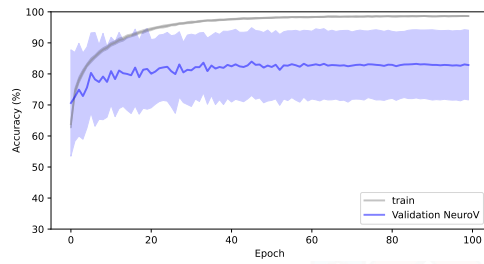


**Figura 5.22:** Ruido: Curva de accuracy durante un entrenamiento de 100 épocas. Entrenado con PD-Neurovoz.

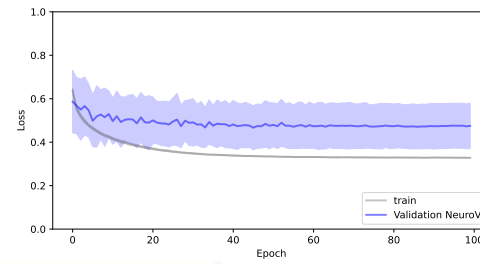


**Figura 5.23:** Ruido: Curva de pérdida durante un entrenamiento de 100 épocas. Entrenado con PD-Neurovoz.

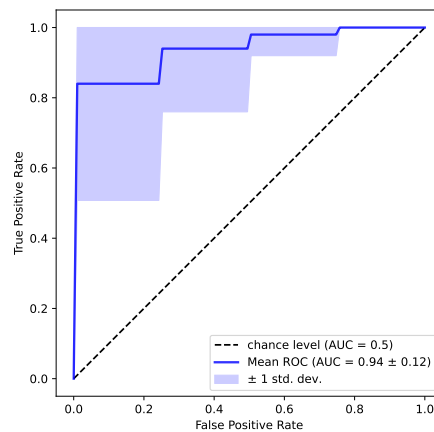


**Figura 5.24:** Ruido: Curva ROC. Entrenado con PD-Neurovoz.

Además, se obtiene la siguiente gráfica T-SNE:

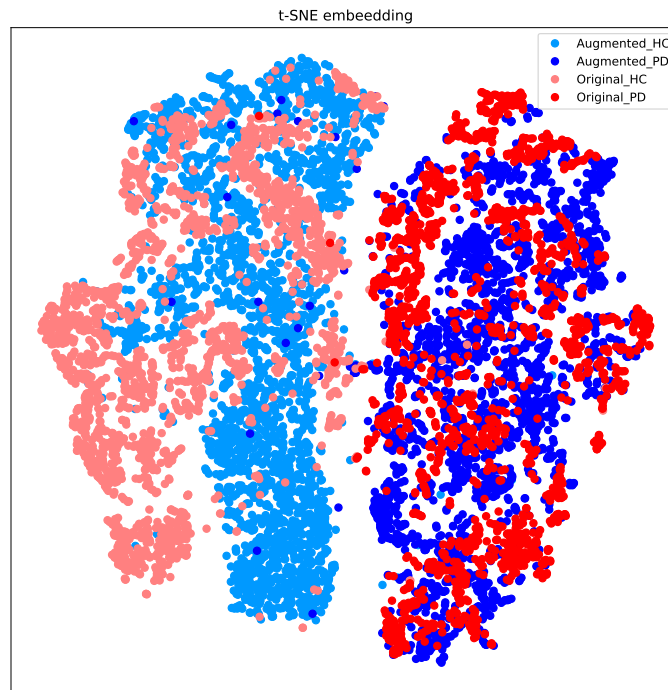


**Figura 5.25:** Ruido: T-SNE de un entrenamiento con PD-Neurovoz.

Al respecto, se puede decir que se cuenta con un comportamiento bastante similar al ocurrido en un entrenamiento con PD-GITA sobretodo en sus curvas de accuracy y pérdida. Además, en la figura 5.25 se puede notar que en el caso de pacientes de la enfermedad, los datos aumentados y originales logran mezclarse de buena forma y se muestra una alta variabilidad al respecto, mientras que para las personas que no padecen la enfermedad no se logran mezclar del todo e incluso se logra detectar una separación entre ambos tipos de datos, muy similar al comportamiento ilustrado en 5.16.

## 5.7. Pruebas adicionales

### 5.7.1. Mezcla de técnicas

En base al desempeño de las distintas técnicas se decide realizar pruebas con una mezcla de técnicas, estas pruebas se caracterizan por concatenar data aumentada obtenida con dos técnicas distintas, las combinaciones que se decide realizar son:

1. Speed change y Pitch change
2. Speed change y Enmascaramiento

La principal razón para utilizar estas 3 técnicas es debido a que según la tabla 5.11 son las que más ayudan en la generalización o que mejoran el rendimiento del modelo.

Es importante mencionar que en estos casos se cuenta con una mayor cantidad de datos para el entrenamiento pues se agrega las imágenes aumentadas de la primera técnica como de la segunda por separado. El resumen de los

resultados se entrega a continuación:

**Tabla 5.15:** Comparación de métricas de mezcla técnicas de aumento de data. Entrenado con PD-GITA

| Técnica                               | Covarianza       |                  | GITA                 | Neurovoz            |
|---------------------------------------|------------------|------------------|----------------------|---------------------|
|                                       | HC               | PD               | Accuracy Subjects    | Accuracy Subjects   |
| <b>Pitch change + Speed Change</b>    | $31.45 \pm 3.52$ | $45.45 \pm 4.37$ | $100.00 \pm 0.00 \%$ | $53.95 \pm 3.61 \%$ |
| <b>Speed change + Enmascaramiento</b> | $16.60 \pm 2.87$ | $23.33 \pm 1.64$ | $100.00 \pm 0.00 \%$ | $51.98 \pm 3.37 \%$ |

De los resultados se puede decir que al mezclar las dos técnicas que entregan una mayor variabilidad en los datos (*pitch change + speed change*) se logra que el nuevo conjunto de datos de entrenamiento cuente con una variabilidad aún mayor, sin embargo, esto no presenta una mejora en la generalización del modelo, más bien, mejora la clasificación dentro de la misma base de datos, logrando la perfección. Esto contradice la idea de que la generalización y la varianza de los datos está directamente relacionada y abre la posibilidad de que una tercera variable, relacionada a la cantidad de datos entre en juego.

A continuación se muestra las curvas de accuracy y pérdida de la combinación *Pitch change* junto a *Speed change*:

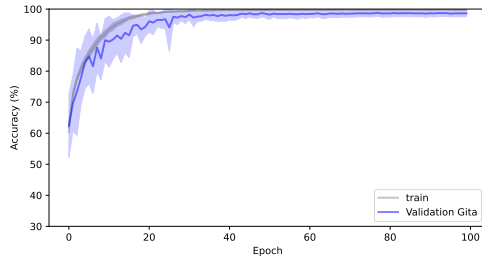


**Figura 5.26:** Mezcla: Curva de accuracy durante un entrenamiento de 100 épocas.

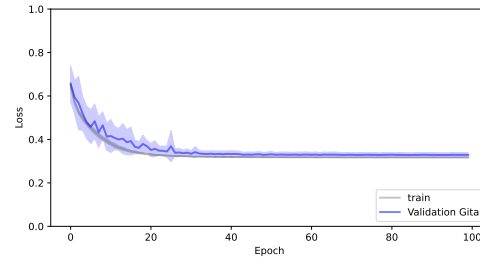


**Figura 5.27:** Mezcla: Curva de pérdida durante un entrenamiento de 100 épocas.

Se puede notar se logra un entrenamiento muy prolijo pero como se menciona anteriormente, no necesariamente implica que se logran resultados replicables.

### 5.7.2. Mezcla de bases de datos

Por último, se realiza un análisis del desempeño de un modelo al ser entrenado con ambas bases de datos en conjunto a data aumentada utilizando la técnica *pitch change*. Se busca además comparar con los resultados obtenidos en [20] al trabajar con dominio adversario. La siguiente tabla resume el comportamiento de dicho modelo con respecto a un entrenamiento sin data aumentada. Es importante mencionar que se aplica *pitch change* a ambas bases de datos y se agrega dicha información al conjunto de entrenamiento.

**Tabla 5.16:** Comparación de métricas en mezcla de bases de datos. Validado con un promedio de ambas bases de datos.

| Técnica             | Accuracy Subjects    | Sensitivity | Specifity | F1 score | MCC  | AUC  |
|---------------------|----------------------|-------------|-----------|----------|------|------|
| <b>Original</b>     | $73.72 \pm 12.77 \%$ | 76.32 %     | 71.56 %   | 74.40 %  | 0.49 | 0.84 |
| <b>Pitch change</b> | $82.98 \pm 8.38 \%$  | 82.34 %     | 83.67 %   | 82.85 %  | 0.68 | 0.92 |

Se considera la siguiente información como referencia:

**Tabla 5.17:** Comparación de métricas en mezcla de bases de datos. Comparación entre modelo base y modelo con Dominio Adversario. Obtenida desde [20]

|             | Baseline Model |           | DomAd Model |           |
|-------------|----------------|-----------|-------------|-----------|
|             | Gita           | Neurovoz  | Gita        | Neurovoz  |
| <b>Acc.</b> | 71.9±8.8       | 76.7±21.6 | 76.1±11.8   | 80.6±19.6 |
| <b>Sens</b> | 70.5           | 80.0      | 76.5        | 80.5      |
| <b>Spec</b> | 74.0           | 73.5      | 76.0        | 81.0      |
| <b>AUC</b>  | 0.9±0.1        | 0.9±0.2   | 0.8±0.2     | 0.9±0.2   |

Es importante mencionar que las métricas entregadas en la tabla 5.17 tanto para el modelo base como para el modelo con dominio adversario son entregadas de manera separada cada una de las bases de datos, es decir, se cuenta con dos conjuntos de validación separados, mientras que en el caso de la tabla 5.16 se realiza una mezcla entre ambas bases de datos en la validación y por tanto las métricas reflejan dicho comportamiento conjunto. Para comparar sus resultados se debe tener en cuenta que las métricas asociadas a cada una de las bases de datos por separado, de manera conjunta tenderán a un valor cercano al promedio entre ambas.

En primer lugar, respecto a la tabla 5.16 se puede notar que el uso del aumento de data mejora en aproximadamente un 10 % la exactitud de clasificación y las demás métricas reportadas, siguiendo el comportamiento ya mencionado al utilizar las técnicas de manera separada.

Luego, al comparar los resultados con la tabla 5.17 se puede notar que se logra resultados similares entre el uso de *pitch change* y dominio adversario, aunque ligeramente mayores en el caso del aumento de data. Así, se considera necesario realizar pruebas relacionadas al uso de aumento de data en conjunto a dominio adversario, sin embargo, esto puede resultar contradictorio pues se podría considerar a la data sintética como un dominio distinto y empeorar el comportamiento por tanto se define como un tema interesante de analizar en trabajos futuros.

Respecto a la variabilidad que entrega el uso de esta técnica se cuenta con la siguiente tabla y gráfica de t-SNE:

**Tabla 5.18:** Reporte de variabilidad de datos. Entrenado con mezcla de base de datos y aumento de data.

|                    | HC    | PD    |
|--------------------|-------|-------|
| <b>Divergencia</b> | 21.67 | 19.87 |
| <b>Covarianza</b>  | 44.99 | 23.86 |

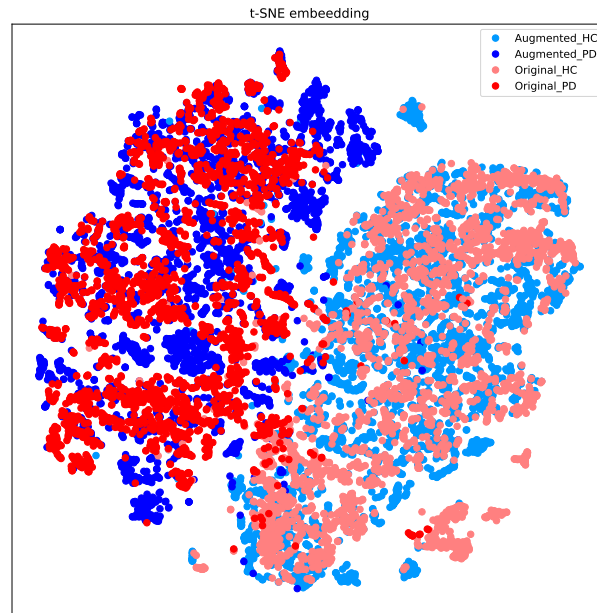


**Figura 5.28:** Gráfica de T-SNE para mezcla de base de datos

De la información anterior, se puede notar una alta variabilidad entregada por la técnica *pitch change*, sin embargo no es la única responsable por dicho comportamiento ya que también se considera la diferencia existente entre los datos originales de cada conjunto. Sin duda su alta covarianza es alentadora respecto a la posible capacidad de generalización de este modelo, sin embargo es necesaria una tercera base de datos para realizar dicha prueba y comprobar la afirmación entregada anteriormente respecto a la relación entre la variabilidad de los datos y la capacidad de generalizar en base a ellos.

## 6 | Conclusiones y trabajo futuro

Considerando la información anterior, se destaca que no existe una técnica óptima general que logre mejorar el comportamiento de un modelo de manera intrínseca, finalmente depende de las características que posea la base de datos utilizada para entrenar.

Por otro lado, en base al trabajo realizado es posible afirmar que la variabilidad que entregue la técnica utilizada con respecto a sus pares originales afecta de manera directa en la mejora a la capacidad de generalizar del modelo entrenado. En tal caso, se considera a *Pitch change* como la técnica óptima a la hora de trabajar con PD-GITA pues logra aumentar de manera efectiva los datos, generando ‘nuevos sujetos’ de manera sintética que permiten al modelo generalizar de mejor manera con respecto al no uso de la técnica.

Además, es importante mencionar la eficacia que tienen todas las técnicas evaluadas en la mejora (algunas más que otras) del comportamiento general del modelo frente a una validación con la misma base de datos. Este comportamiento es logrado independiente la base de datos utilizada para entrenar.

Un caso interesante de analizar como trabajo futuro es el uso del enmascaramiento temporal y en frecuencia, pues en base a la comparación con datos duplicados, logra métricas bastante similares, lo que hace cuestionar si realmente aporta de manera significativa o vale la pena aplicarlo, dado que al simplemente duplicar el conjunto de datos de entrenamiento sin ninguna modificación se logra resultados parecidos.

Respecto a la mezcla de técnicas, si bien se obtuvo buenos resultados, que mejoran el modelo base, no son significativos para justificar el costo computacional que conlleva trabajar con el triple de datos respecto al uso de una técnica, donde incluso se puede obtener mejores resultados.

Por otro lado, en base a la mezcla de bases de datos, se considera necesario estudiar la combinación de técnicas de aumento de data en un modelo de dominio adversario, con el fin de conocer su efectividad bajo dicho ambiente. Además se espera poder realizar una validación cruzada con una tercera base de datos que permita medir y comparar la capacidad de generalización al utilizar datos tan diversos en el entrenamiento.

Por ultimo, se retoma lo expuesto al entrenar con la base de datos PD-Neurovoz, donde se expone la necesidad de estudiar con mayor profundidad la relación entre las características de una determinada base de datos y como afecta una técnica de aumento de data en su información y posteriores resultados. Sin duda una próxima investigación debe seguir dicho camino, con el fin de definir criterios de selección de técnicas que permitan optimizar los modelos entrenados y signifiquen un aporte a la capacidad de extrapolar el aprendizaje de estos.

# Bibliografía

- 
- [1] Mayela Rodríguez-Violante y Amin Cervantes-Arriaga Kenia Arredondo Rosalía Zerón-Martínez. «Breve recorrido histórico de la enfermedad de Parkinson a 200 años de su descripción». En: (2018). URL: [https://www.gacetamedicademexico.com/files/gmm\\_6\\_18\\_720-727.pdf](https://www.gacetamedicademexico.com/files/gmm_6_18_720-727.pdf).
  - [2] Organización Mundial de la Salud. «Enfermedad de Parkinson». En: (2022). URL: <https://www.who.int/es/news-room/fact-sheets/detail/parkinson-disease>.
  - [3] LinLin Xie y LuDong Hu. «Research progress in the early diagnosis of Parkinson's disease». En: (2022). URL: <https://link-springer-com.usm.idm.oclc.org/article/10.1007/s10072-022-06316-0>.
  - [4] P. Pastora y E. Tolosaa. «La enfermedad de Parkinson: diagnóstico y avances en el conocimiento de la etiología y en el tratamiento». En: (2001). URL: <https://www.elsevier.es/es-revista-medicina-integral-63-articulo-la-enfermedad-parkinson-diagnostico-avances-10021650>.
  - [5] Andrew Ma, Kenneth K Lau y Dominic Thyagarajan. «Voice changes in Parkinson's disease: What are they telling us?» En: *Journal of Clinical Neuroscience* 72 (2020), págs. 1-7. ISSN: 0967-5868. doi: <https://doi.org/10.1016/j.jocn.2019.12.029>. URL: <https://www.sciencedirect.com/science/article/pii/S0967586819317631>.
  - [6] Quoc Cuong Ngo, Mohammad Abdul Motin, Nemuel Daniel Pah, Peter Drotár, Peter Kempster y Dinesh Kumar. «Computerized analysis of speech and voice for Parkinson's disease: A systematic review». En: *Computer Methods and Programs in Biomedicine* 226 (2022), pág. 107133. ISSN: 0169-2607. doi: <https://doi.org/10.1016/j.cmpb.2022.107133>. URL: <https://www.sciencedirect.com/science/article/pii/S0169260722005144>.
  - [7] Ahmed A Elngar, Mohamed Arafa, Amar Fathy, Basma Moustafa, Omar Mahmoud, Mohamed Shaban y Nehal Fawzy. «Image classification based on CNN: a survey». En: *Journal of Cybersecurity and Information Management* 6.1 (2021), págs. 18-50.
  - [8] Leiyu Chen, Shaobo Li, Qiang Bai, Jing Yang, Sanlong Jiang y Yanming Miao. «Review of Image Classification Algorithms Based on Convolutional Neural Networks». En: *Remote Sensing* 13.22 (2021). ISSN: 2072-4292. doi: [10.3390/rs13224712](https://doi.org/10.3390/rs13224712). URL: <https://www.mdpi.com/2072-4292/13/22/4712>.
  - [9] L. Cai, J. Gao y D. Zhao. «A review of the application of deep learning in medical image classification and segmentation». En: *Ann Transl Med* 8.11 (jun. de 2020), pág. 713. doi: [10.21037/atm.2020.02.44](https://doi.org/10.21037/atm.2020.02.44).
  - [10] Saman Sarraf, Danielle D DeSouza, John Anderson, Ghassem Tofghi y Alzheimer's Disease Neuroimaging Initiativ. «DeepAD: Alzheimer's disease classification via deep convolutional neural networks using MRI and fMRI». En: *BioRxiv* (2016), pág. 070441.
  - [11] Mahati Munikoti Srikantamurthy, VP Subramanyam Rallabandi, Dawood Babu Dudekula, Sathishkumar Natarajan y Junhyung Park. «Classification of benign and malignant subtypes of breast cancer histopathology imaging using hybrid CNN-LSTM based transfer learning». En: *BMC Medical Imaging* 23.1 (2023), pág. 19.
  - [12] Hang Yu, Laurence T. Yang, Qingchen Zhang, David Armstrong y M. Jamal Deen. «Convolutional neural networks for medical image analysis: State-of-the-art, comparisons, improvement and perspectives». En: *Neurocomputing* 444 (2021), págs. 92-110. ISSN: 0925-2312. doi: <https://doi.org/10.1016/j.neucom.2020.04.157>. URL: <https://www.sciencedirect.com/science/article/pii/S0925231221001314>.

- [13] Hongzhao Guan y Alexander Lerch. «Learning strategies for voice disorder detection». En: *2019 IEEE 13th International Conference on Semantic Computing (ICSC)*. IEEE. 2019, págs. 295-301.
- [14] C Hema y Fausto Pedro Garcia Marquez. «Emotional speech recognition using cnn and deep learning techniques». En: *Applied Acoustics* 211 (2023), pág. 109492.
- [15] Muhammad Muzammel, Hanan Salam, Yann Hoffmann, Mohamed Chetouani y Alice Othmani. «AudVowelConsNet: A phoneme-level based deep CNN architecture for clinical depression diagnosis». En: *Machine Learning with Applications* 2 (2020), pág. 100005.
- [16] Yao-Yuan Yang, Moto Hira, Zhaoeng Ni, Anjali Chourdia, Artyom Astafurov, Caroline Chen, Ching-Feng Yeh, Christian Puhersch, David Pollack, Dmitriy Genzel, Donny Greenberg, Edward Z. Yang, Jason Lian, Jay Mahadeokar, Jeff Hwang, Ji Chen, Peter Goldsborough, Prabhat Roy, Sean Narenthiran, Shinji Watanabe, Soumith Chintala, Vincent Quenneville-Bélair y Yangyang Shi. «TorchAudio: Building Blocks for Audio and Speech Processing». En: *arXiv preprint arXiv:2110.15018* (2021).
- [17] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai y Soumith Chintala. *PyTorch: An Imperative Style, High-Performance Deep Learning Library*. <https://pytorch.org>. Accessed: Feb 13, 2024. 2019.
- [18] Haytham M. Fayek. *Speech Processing for Machine Learning: Filter banks, Mel-Frequency Cepstral Coefficients (MFCCs) and What's In-Between*. 2016. URL: <https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html>.
- [19] Máté Hireš, Matej Gazda, Peter Drotár, Nemuel Daniel Pah, Mohammad Abdul Motin y Dinesh Kant Kumar. «Convolutional neural network ensemble for Parkinson's disease detection from voice recordings». En: *Computers in Biology and Medicine* 141 (2022), pág. 105021. ISSN: 0010-4825. DOI: <https://doi.org/10.1016/j.compbio.2021.105021>. URL: <https://www.sciencedirect.com/science/article/pii/S0010482521008155>.
- [20] E Ibarra-Sulbaran, J Arias-Londoño, M Zañartu y J Godiño-Llorente. «Domain adversarial convolutional neural network for Parkinson's disease detection from speech». En: *Proceedings of the Models and Analysis of Vocal Emissions for Biomedical Applications: 13th International Workshop (MAVEBA 2023)*. 2023, págs. 69-72.
- [21] Máté Hireš, Peter Drotár, Nemuel Daniel Pah, Quoc Cuong Ngo y Dinesh Kumar. «On the Inter-Dataset Generalisation of Machine Learning Approaches to Parkinson's Disease Detection From Voice». En: *International Journal of Medical Informatics* (2023). Available at SSRN: <https://ssrn.com/abstract=4327662> or <http://dx.doi.org/10.2139/ssrn.4327662>. DOI: [10.1016/j.ijmedinf.2023.105237](https://doi.org/10.1016/j.ijmedinf.2023.105237).
- [22] Jamil Ahmad, Khan Muhammad y Sung Baik. «Data augmentation-assisted deep learning of hand-drawn partially colored sketches for visual search». En: *PLOS ONE* 12 (ago. de 2017), e0183838. DOI: [10.1371/journal.pone.0183838](https://doi.org/10.1371/journal.pone.0183838).
- [23] Jason Wang, Luis Perez et al. «The effectiveness of data augmentation in image classification using deep learning». En: *Convolutional Neural Networks Vis. Recognit* 11.2017 (2017), págs. 1-8.
- [24] Farhad Javanmardi, Sudarsana Reddy Kadiri y Paavo Alku. «A comparison of data augmentation methods in voice pathology detection». En: *Computer Speech Language* 83 (2024), pág. 101552. ISSN: 0885-2308. DOI: <https://doi.org/10.1016/j.csl.2023.101552>. URL: <https://www.sciencedirect.com/science/article/pii/S0885230823000712>.
- [25] Máté Hireš, Matej Gazda, Lukáš Vavrek y Peter Drotár. «Voice-Specific Augmentations for Parkinson's Disease Detection Using Deep Convolutional Neural Network». En: *2022 IEEE 20th Jubilee World Symposium on Applied Machine Intelligence and Informatics (SAMI)*. 2022, págs. 000213-000218. DOI: [10.1109/SAMI54271.2022.9780856](https://doi.org/10.1109/SAMI54271.2022.9780856).
- [26] Daniel S. Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D. Cubuk y Quoc V. Le. «SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition». En: *Proc. Interspeech 2019*. 2019, págs. 2613-2617. DOI: [10.21437/Interspeech.2019-2680](https://doi.org/10.21437/Interspeech.2019-2680).

- [27] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt y SciPy 1.0 Contributors. «SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python». En: *Nature Methods* 17 (2020), págs. 261-272. doi: [10.1038/s41592-019-0686-2](https://doi.org/10.1038/s41592-019-0686-2).
- [28] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot y E. Duchesnay. «Scikit-learn: Machine Learning in Python». En: *Journal of Machine Learning Research* 12 (2011), págs. 2825-2830.
- [29] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke y Travis E. Oliphant. «Array programming with NumPy». En: *Nature* 585.7825 (sep. de 2020), págs. 357-362. doi: [10.1038/s41586-020-2649-2](https://doi.org/10.1038/s41586-020-2649-2). URL: <https://doi.org/10.1038/s41586-020-2649-2>.
- [30] J. D. Hunter. «Matplotlib: A 2D graphics environment». En: *Computing in Science & Engineering* 9.3 (2007), págs. 90-95. doi: [10.1109/MCSE.2007.55](https://doi.org/10.1109/MCSE.2007.55).
- [31] Guillaume Lemaître, Fernando Nogueira y Christos K. Aridas. «Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning». En: *Journal of Machine Learning Research* 18.17 (2017), págs. 1-5. URL: <http://jmlr.org/papers/v18/16-365.html>.
- [32] Davide Chicco y Giuseppe Jurman. «The Matthews correlation coefficient (MCC) should replace the ROC AUC as the standard metric for assessing binary classification». En: *BioData Min* 16.1 (2023), pág. 4. doi: [10.1186/s13040-023-00322-4](https://doi.org/10.1186/s13040-023-00322-4).
- [33] Juan Terven, Diana M Cordova-Esparza, Alfonso Ramirez-Pedraza y Edgar A Chavez-Urbiola. «Loss functions and metrics in deep learning. A review». En: *arXiv preprint arXiv:2307.02694* (2023).
- [34] Fernando Pérez-Cruz. «Kullback-Leibler divergence estimation of continuous distributions». En: *2008 IEEE international symposium on information theory*. IEEE. 2008, págs. 1666-1670.
- [35] Juan Rafael Orozco-Arroyave, Julián David Arias-Londoño, Jesús Francisco Vargas-Bonilla, María Claudia Gonzalez-Rátiva y Elmar Nöth. «New Spanish speech corpus database for the analysis of people suffering from Parkinson's disease.» En: *LREC*. 2014, págs. 342-347.
- [36] Laureano Moro-Velazquez, Jorge Andres Gomez-Garcia, Juan Ignacio Godino-Llorente, Jesús Villalba, Jan Rusz, Stephanie Shattuck-Hufnagel y Najim Dehak. «A forced gaussians based methodology for the differential evaluation of Parkinson's Disease by means of speech processing». En: *Biomedical Signal Processing and Control* 48 (2019), págs. 205-220.
- [37] Alex S. Ozbolt, Laureano Moro-Velazquez, Ioan Lina, Ankur A. Butala y Najim Dehak. «Things to Consider When Automatically Detecting Parkinson's Disease Using the Phonation of Sustained Vowels: Analysis of Methodological Issues». En: *Applied Sciences* 12.3 (2022). ISSN: 2076-3417. doi: [10.3390/app12030991](https://doi.org/10.3390/app12030991). URL: <https://www.mdpi.com/2076-3417/12/3/991>.