

Santiago de Cali, 5 de diciembre del 2024

Doctor

Diego Luis Linares O.

Director Maestría en Ciencia de Datos

Facultad de Ingeniería y Ciencias

Pontificia Universidad Javeriana de Cali

Asunto: Presentación para evaluación del proyecto aplicado

Cordial Saludo,

Con el fin de cumplir con los requisitos exigidos por la Universidad para optar por el título de Magíster en Ciencia de Datos, nos permitimos presentar a su consideración el proyecto denominado “Identificación automática del núcleo subtalámico mediante el entrenamiento de redes neuronales convolucionales a partir de señales MER”, el cual fue realizado por los estudiantes Andrés Parada Hernández con código 8984943, Christian Camilo Vega Preciado 8985201 y Diego Alejandro Lozano Millán con código 8985630 pertenecientes a la Maestría en Ciencia de Datos, bajo la dirección de Hernán Darío Vargas Cardona.

El suscrito director del Proyecto Aplicado autoriza para que se proceda a hacer la evaluación de este proyecto, toda vez que ha revisado cuidadosamente el documento y avala que ya se encuentra listo para ser presentado y sustentado oficialmente.


Atentamente,

Estudiantes:

Firma 1:  Nombre: Diego Alejandro Lozano Millán C.C.: 1113786772

Firma 2: Christian Vega Nombre: Christian Camilo Vega C.C.: 1075295219

Firma 3:  Nombre: Andrés Parada Hernández C.C.: 16070776

Director: 
Firma: _____ Nombre: Hernán Darío Vargas C.C.: 1097721437

Documentación anexa:

Resumen del Proyecto Aplicado en formato digital (máximo 1 página).

Una copia digital (PDF) del documento del proyecto aplicado

FICHA RESUMEN
PROYECTO APLICADO – MAESTRÍA EN CIENCIA DE DATOS

**TITULO: IDENTIFICACIÓN AUTOMÁTICA DEL NÚCLEO
SUBTALÁMICO MEDIANTE EL ENTRENAMIENTO DE REDES
NEURONALES CONVOLUCIONALES A PARTIR DE SEÑALES MER**

1. **ÁREA DE TRABAJO:** Procesamiento digital de señales
2. **TIPO DE PROYECTO:** Aplicado
3. **ESTUDIANTE(S):** Andrés Parada Hernández, Christian Camilo Vega y Diego Alejandro Lozano.
4. **CORREO ELECTRÓNICO:**
 - aphz22@javerianacali.edu.co
 - christianvega@javerianacali.edu.co
 - diegoalejandrolozano@javerianacali.edu.co
5. **DIRECCIÓN Y TELEFONO:**
 - Cra. 10 # 4B-51 Sibaté – Cundinamarca, 3123068180
 - Cra 9 w # 25 b 20 Neiva – Huila. 3015177507
 - Calle 11 # 2 – 27 Roldanillo – Valle del Cauca, 3147333417
6. **DIRECTOR:** Hernán Darío Vargas Cardona
7. **VINCULACIÓN DEL DIRECTOR:** Tiempo completo
8. **CORREO ELECTRÓNICO DEL DIRECTOR:**
 - hernan.vargas@javerianacali.edu.co
9. **CO-DIRECTOR:** No aplica
10. **GRUPO O EMPRESA QUE LO AVALA:** No aplica
11. **OTROS GRUPOS O EMPRESAS:** No aplica
12. **PALABRAS CLAVE:** Núcleo Subtalámico, Enfermedad de Parkinson, Red Neuronal, Señales MER, Estimulación Cerebral Profunda.
13. **FECHA DE INICIO:** 2024/01/03
14. **FECHA DE FINALIZACIÓN:** 2024/12/15
15. **RESUMEN:**

Según la Organización Mundial de la Salud, la enfermedad de Parkinson ha impactado de manera significativa a la población mayor en todo el mundo. Para reducir los efectos secundarios de los medicamentos y superar las limitaciones de las terapias convencionales, la estimulación cerebral profunda (DBS, por sus siglas en inglés) se ha consolidado como una alternativa de gran valor. No obstante, su aplicación plantea desafíos importantes. Este procedimiento se centra en la implantación de microelectrodos en el núcleo subtalámico (STN, Subthalamic Nucleus) para, posteriormente, aplicar una estimulación que disminuye los temblores característicos de la enfermedad. Aunque se han logrado avances en la identificación del STN, el proceso sigue siendo complejo y requiere el desarrollo de sistemas automatizados que permitan a los especialistas tomar decisiones informadas con mayor facilidad.

Para abordar estos desafíos en la identificación del STN mediante señales MER, se formuló la siguiente pregunta de investigación principal: ¿Cómo identificar el STN mediante el entrenamiento de modelos basados en CNN con las señales MER?. A partir de esta interrogante, se derivan otras preguntas clave que estructuran el proceso de investigación: ¿Cuáles son los procedimientos necesarios para llevar a cabo el preprocesamiento de una base de datos de señales MER previamente etiquetadas, incluyendo registros del STN y otras estructuras relevantes en la EP?; ¿Cuáles son las arquitecturas o modelos basados en CNN más adecuados para implementar en la identificación del STN a partir de señales MER?; y, finalmente, ¿Cómo se puede realizar una evaluación robusta y confiable del rendimiento de los diferentes modelos de CNN en el contexto de clasificación, utilizando métricas apropiadas para medir su eficacia?

Estas preguntas orientan el desarrollo de una metodología sólida que contempla desde el procesamiento de datos hasta la elección de modelos y la evaluación rigurosa de su desempeño, con el objetivo de optimizar la identificación del STN en pacientes con Parkinson.



Pontificia Universidad
JAVERIANA
Cali

**IDENTIFICACIÓN AUTOMÁTICA DEL NÚCLEO SUBTALÁMICO
MEDIANTE EL ENTRENAMIENTO DE REDES NEURONALES
CONVOLUCIONALES A PARTIR DE SEÑALES MER**

Diego Alejandro Lozano Millán

Código 8985630

Andrés Parada Hernández

Código 8984943

Christian Camilo Vega Preciado

Código 8985201

*Proyecto Aplicado para optar al título de
Magister en Ciencia de Datos*

Director

Hernán Darío Vargas Cardona

FACULTAD DE INGENIERÍA Y CIENCIAS
MAESTRÍA EN CIENCIA DE DATOS
SANTIAGO DE CALI, DICIEMBRE 5 DE 2024

TABLA DE CONTENIDO

1.	INTRODUCCIÓN	1
2.	DEFINICIÓN DEL PROBLEMA.....	3
2.1.	PLANTEAMIENTO DEL PROBLEMA.....	3
2.2.	FORMULACIÓN DEL PROBLEMA	5
3.	OBJETIVOS DEL PROYECTO.....	6
3.1.	OBJETIVO GENERAL	6
3.2.	OBJETIVOS ESPECÍFICOS.....	6
4.	MARCO TEÓRICO Y ANTECEDENTES	7
4.1.	ENFERMEDAD DE PARKINSON.....	7
4.1.1.	Tratamientos	7
4.1.2.	Estimulación Cerebral Profunda	8
4.1.3.	Núcleo Subtalámico	8
4.2.	REDES NEURONALES CONVOLUCIONALES	9
4.2.1.	Operaciones Convolutivas	10
4.2.2.	Capas de Agrupación (Pooling).....	10
4.2.3.	Capas Completamente Conectadas	10
4.3.	MÉTRICAS DE EVALUACIÓN.....	11
4.3.1.	Matriz de confusión.....	11
4.3.2.	Exactitud (Accuracy).....	11
4.3.3.	Precisión (Precision)	12
4.3.4.	Recuperación (Recall).....	12
4.3.5.	F1-score	12
4.3.6.	Curva ROC-AUC	12

4.4.	TRANSFERENCIA DE CONOCIMIENTO	13
4.5.	ANTECEDENTES.....	14
4.5.1.	Representación óptima de señales MER aplicada a la identificación de estructuras cerebrales durante la estimulación cerebral profunda	14
4.5.2.	El aprendizaje multipaciente aumenta la precisión para la identificación del núcleo subtalámico en la estimulación cerebral profunda	14
4.5.3.	Representación dispersa de señales MER para la localización del núcleo subtalámico en la cirugía de la Enfermedad de Parkinson	15
4.5.4.	Representación óptima de señales MER mediante el método de Frames aplicado a la cirugía de la Enfermedad de Parkinson	15
4.5.5.	Localización automática de electrodos de estimulación cerebral profunda en imágenes de tomografía computarizada: aplicación a la Enfermedad de Parkinson.....	16
4.5.6.	Algoritmos de clasificación lineal para la identificación de zonas cerebrales	16
4.5.7.	Software para análisis de actividad neuronal en tiempo real durante neurocirugía ...	17
4.5.8.	Selección de características mediante un conjunto de paquetes de ondículas óptimas y máquina de aprendizaje: aplicación a señales de registro de microelectrodos (MER)	17
5.	PREPARACIÓN DE LOS DATOS Y DEFINICIÓN DEL ENTORNO.....	18
5.1.	CONFIGURACIÓN DEL ENTORNO DE TRABAJO	18
5.2.	DESCRIPCIÓN DEL CONJUNTO DE DATOS	19
5.3.	PREPROCESAMIENTO DE DATOS	20
6.	DESARROLLO DE MODELOS.....	24
6.1.	CREACIÓN DE LOS MODELOS	25
6.1.1.	MODELOS CNN 1D	25
6.1.2.	MODELOS CNN 2D SIN TRANSFER LEARNING	26
6.1.3.	MODELOS CNN 2D CON TRANSFER LEARNING	30
6.1.4.	ENTRENAMIENTO DE LOS MODELOS.....	32

7.	ANÁLISIS DE RESULTADOS	34
7.1.	EVALUACIÓN DEL RENDIMIENTO DE LOS MODELOS.....	34
7.1.1.	RENDIMIENTO DE LOS MODELOS CNN 1D.....	34
7.1.2.	RENDIMIENTO DE LOS MODELOS CNN 2D SIN TRANSFER LEARNING ...	35
7.1.3.	RENDIMIENTO DE LOS MODELOS CNN 2D CON VGG19.....	36
7.1.4.	RENDIMIENTO DE LOS MODELOS CNN 2D CON RESNET50	38
7.2.	COMPARACIÓN DE MODELOS	39
7.3.	INTERPRETACIÓN DE RESULTADOS	40
7.4.	LIMITACIONES Y CONSIDERACIONES.....	40
8.	CONCLUSIONES Y TRABAJOS FUTUROS	42
8.1.	CONCLUSIONES	42
8.2.	TRABAJOS FUTUROS	43
9.	REFERENCIAS BIBLIOGRÁFICAS	44
10.	ANEXOS.....	49
	Anexo A: Estructura de los Modelos 1D.....	49
	A.1. Modelo 1	49
	A.2. Modelo 2.....	49
	A.3. Modelo 3.....	50
	A.4. Modelo 4.....	50
	A.5. Modelo 5.....	51
	Anexo B: Estructura de los Modelos 2D sin TL	52
	B.1. Modelos desarrollados con los escalogramas de la función madre CGAU	52
	B.2. Modelos desarrollados con los escalogramas de la función madre CMOR.....	53
	B.3. Modelos desarrollados con los escalogramas de la función madre MEXH.....	55

Anexo C: Estructura de los modelos creados con VGG19.....	57
C.1. Modelos desarrollados con los escalogramas de la función madre CGAU	57
C.2. Modelos desarrollados con los escalogramas de la función madre CMOR.....	58
C.3. Modelos desarrollados con los escalogramas de la función madre MEXH.....	60
Anexo D: Estructura de los modelos creados con ResNet50	62
D.1. Modelos desarrollados con los escalogramas de la función madre CGAU	62
D.2. Modelos desarrollados con los escalogramas de la función madre CMOR	63
D.3. Modelos desarrollados con los escalogramas de la función madre MEXH	65
Anexo E: Enlace al repositorio del proyecto	66
Anexo F: Resultados de los modelos 1D.....	67
F.1. Modelo 1	67
F.2. Modelo 2	67
F.3. Modelo 3	68
F.4. Modelo 4	68
F.5. Modelo 5	69
Anexo G: Resultados de los modelos 2D sin TL	69
G.1. Modelos desarrollados con los escalogramas de la función madre CGAU.....	69
G.2. Modelos desarrollados con los escalogramas de la función madre CMOR.	71
G.3. Modelos desarrollados con los escalogramas de la función madre MEXH.	72
Anexo H: Resultados de los modelos creados con VGG19	74
H.1. Modelos desarrollados con los escalogramas de la función madre CGAU.....	74
H.2. Modelos desarrollados con los escalogramas de la función madre CMOR.	75
H.3. Modelos desarrollados con los escalogramas de la función madre MEXH.	77
Anexo I: Resultados de los modelos creados con ResNet50.....	79
I.1. Modelos desarrollados con los escalogramas de la función madre CGAU.	79

I.2. Modelos desarrollados con los escalogramas de la función madre CMOR.....	80
I.3. Modelos desarrollados con los escalogramas de la función madre MEXH.....	82

1. INTRODUCCIÓN

En las últimas décadas, el envejecimiento de la población ha provocado un alarmante aumento en las enfermedades neurodegenerativas, siendo el Parkinson una de las más prevalentes. Este trastorno progresivo afecta gravemente la movilidad y la calidad de vida de quienes lo padecen, debido a la disminución de los niveles de dopamina causada por la degeneración de neuronas en la sustancia negra. Esta deficiencia desencadena síntomas motores incapacitantes, como temblores, rigidez y bradicinesia [1][2]. La alta prevalencia de esta enfermedad y el impacto de sus síntomas subrayan la urgente necesidad de mejorar los métodos de tratamiento y manejo, especialmente en sus etapas avanzadas [3].

Con el avance de la enfermedad de Parkinson (EP), los tratamientos convencionales, como la administración de levodopa, suelen perder efectividad en etapas avanzadas, lo que ha impulsado la adopción de terapias complementarias, como la estimulación cerebral profunda (DBS, *Deep Brain Stimulation*) [2]. La DBS es un procedimiento quirúrgico en el que se implantan microelectrodos en áreas específicas del cerebro, principalmente en el núcleo subtalámico (STN, *Subthalamic Nucleus*), debido a su rol en la modulación de los circuitos motores y su conexión directa con la corteza motora [4] [5]. Este enfoque ha mostrado ser altamente efectivo en el control de los síntomas motores y en la mejora de la calidad de vida en pacientes con EP avanzada. Sin embargo, la exactitud en la colocación de los microelectrodos en el STN es fundamental, ya que una colocación adecuada es clave para maximizar la efectividad del procedimiento.

La identificación del STN durante la DBS es un proceso complejo que requiere métodos avanzados para mejorar la exactitud del procedimiento. En este contexto, se han explorado las redes neuronales convolucionales (CNN, Convolutional Neural Network) aplicadas al procesamiento de las señales generadas por los microelectrodos de registro (MER, Microelectrode Recordings), las cuales permiten captar la actividad neuronal en tiempo real [6].

En este proyecto, se desarrollaron diversas redes CNN entrenadas para procesar señales MER y detectar patrones específicos del STN. Aunque estas redes muestran un potencial significativo para asistir en la identificación de esta región, el alcance de la investigación se limitó a su desarrollo, sin haberse logrado aún su validación en un entorno clínico real. No obstante, los resultados obtenidos sugieren que este enfoque podría tener aplicaciones clínicas en el futuro, representando un avance hacia el uso de la inteligencia artificial, especialmente el campo del Deep Learning, en procedimientos neuroquirúrgicos complejos como la DBS para el tratamiento de la EP en fases avanzadas.

2. DEFINICIÓN DEL PROBLEMA

2.1. PLANTEAMIENTO DEL PROBLEMA

La EP es una patología neurodegenerativa progresiva que afecta significativamente la calidad de vida de los pacientes, principalmente debido a la pérdida de células en la sustancia negra, lo que ocasiona una disminución en los niveles de dopamina, un neurotransmisor esencial para el control de los movimientos motores. Los pacientes que padecen esta enfermedad experimentan síntomas como temblores, rigidez muscular, bradicinesia, y, en fases avanzadas, trastornos del equilibrio y la marcha [7]. Aunque los tratamientos farmacológicos, como la levodopa, proporcionan alivio temporal, su efectividad disminuye con el tiempo, lo que lleva a la necesidad de intervenciones quirúrgicas más avanzadas, como la DBS.

La DBS se ha consolidado como una de las opciones más eficaces para tratar el Parkinson en fases avanzadas, ya que mejora los síntomas motores mediante la intervención en el STN. Sin embargo, la exactitud en la localización de esta estructura durante la cirugía es crucial para maximizar los beneficios terapéuticos y minimizar los efectos adversos. Sistemas como NEUROZONE han demostrado ser herramientas útiles en la identificación del STN, logrando una tasa de éxito superior al 85% con clasificadores como Naive Bayes [8]. No obstante, persisten desafíos relacionados con la exactitud de la identificación, especialmente debido a la naturaleza no estacionaria de las señales MER, que pueden verse afectadas por factores fisiológicos y ambientales.

Con el objetivo de mejorar la precisión en la localización del STN, se han explorado enfoques como el aprendizaje supervisado y no supervisado. Entre estos, los modelos de mezcla gaussiana, junto con algoritmos como el Expectation-Maximization (EM) y la inferencia bayesiana variacional, han mostrado resultados prometedores, con una exactitud del 85% en señales MER y hasta un 90% en electrocardiogramas (ECG) [9]. Sin embargo, aún se requiere optimización adicional para asegurar que estos sistemas sean lo suficientemente confiables para su uso en entornos clínicos de alta demanda.

La investigación sobre la DBS no solo se centra en los avances tecnológicos en la identificación del STN, sino también en la comprensión de los efectos a largo plazo de esta intervención. Estudios como el de Díaz Maroto et al. han abordado el impacto de la DBS en pacientes con Parkinson, destacando la importancia de una localización precisa para maximizar los beneficios terapéuticos y reducir complicaciones, como la disartria o la depresión, que pueden surgir cuando el electrodo no se implanta adecuadamente [10].

En este contexto, surge la necesidad de seguir perfeccionando las técnicas de identificación y procesamiento de señales, combinando enfoques tradicionales con métodos más avanzados de aprendizaje automático. Solo mediante la integración de estas nuevas metodologías será posible mejorar los resultados quirúrgicos y ofrecer a los pacientes con Parkinson una opción terapéutica más segura y efectiva.

2.2. FORMULACIÓN DEL PROBLEMA

En el contexto de la identificación del STN a partir de señales MER se planteó la siguiente pregunta de investigación:

- ¿Cómo identificar el STN mediante el entrenamiento de modelos basados en CNN con las señales MER?

También se plantearon las siguientes preguntas secundarias (formulación):

- ¿Cuáles son los procedimientos necesarios para llevar a cabo el preprocesamiento de una base de datos de señales MER previamente etiquetadas con registros del STN y otras estructuras relevantes en la EP?
- ¿Cuáles son las arquitecturas o modelos más adecuados basados en CNN para implementar en la identificación del STN a partir de señales MER?
- ¿Cómo se puede llevar a cabo una evaluación robusta y confiable del rendimiento de los diferentes modelos de redes neuronales convolucionales en el contexto de clasificación, considerando las métricas más apropiadas para medir su eficacia?

3. OBJETIVOS DEL PROYECTO

3.1. OBJETIVO GENERAL

Identificar automáticamente el STN mediante la utilización de modelos de **CNN** entrenados con señales MER en el contexto de la DBS.

3.2. OBJETIVOS ESPECÍFICOS

- Preprocesar la base de datos de señales MER etiquetadas con registros del STN y otras estructuras relevantes en la enfermedad de Parkinson.
- Identificar el STN mediante el entrenamiento con señales MER de diferentes arquitecturas de **CNN**.
- Evaluar los modelos entrenados empleando métricas estándar del estado del arte para clasificación y soporte diagnóstico.

4. MARCO TEÓRICO Y ANTECEDENTES

La sección del marco teórico se sumerge en un análisis integral de diversos aspectos fundamentales para contextualizar el proyecto. Iniciando con la EP, se explorará su complejidad fisiopatológica. La atención se dirigirá a los tratamientos disponibles para abordar los síntomas de esta enfermedad neurodegenerativa, destacando en particular la técnica de DBS. Dentro de este contexto, se profundizó en el papel crucial del STN y en cómo esta región neural se conecta con la aplicación de la DBS. Para cerrar, se abordó las CNN, exponiendo tanto su definición como sus diversas aplicaciones. Este enfoque estructurado busca proporcionar una visión completa de los elementos clave que sustentan el desarrollo del proyecto.

4.1. ENFERMEDAD DE PARKINSON

La EP es un trastorno neurodegenerativo crónico que afecta principalmente el sistema nervioso central, dando lugar a la pérdida progresiva de células nerviosas dopaminérgicas en la sustancia negra del cerebro [11]. Esta degeneración neuronal resulta en la disminución de la producción de dopamina, un neurotransmisor crucial para la regulación del movimiento y la coordinación motora. Los síntomas característicos de la enfermedad de Parkinson incluyen temblores en reposo, rigidez muscular, bradicinesia (movimientos lentos), y alteraciones en la postura y el equilibrio [12].

4.1.1. Tratamientos

Los tratamientos para la EP abarcan diversas estrategias destinadas a mitigar los síntomas y mejorar la calidad de vida de los pacientes. Entre los enfoques terapéuticos más comunes se encuentran la administración de medicamentos, la fisioterapia, la terapia ocupacional y, en casos más avanzados, la cirugía, particularmente la DBS [13]. Estos tratamientos buscan aliviar la disminución de la dopamina y controlar los síntomas motores característicos de la enfermedad, como temblores, rigidez y bradicinesia.

4.1.2. Estimulación Cerebral Profunda

La Estimulación Cerebral Profunda (DBS, por sus siglas en inglés) representa una técnica terapéutica avanzada empleada en el tratamiento de la EP y otros trastornos neurológicos. Consiste en la implantación de electrodos en áreas específicas del cerebro, típicamente el STN o el Globo Pálido, los cuales son conectados a un dispositivo generador de impulsos eléctricos subcutáneo [14]. Estos impulsos eléctricos modulan la actividad neuronal anormal, aliviando los síntomas motores asociados con la enfermedad. La DBS ha demostrado ser eficaz en mejorar la calidad de vida de los pacientes parkinsonianos, ofreciendo un enfoque para aquellos cuyos síntomas no son adecuadamente controlados con tratamientos convencionales [15].

4.1.3. Núcleo Subtalámico

El STN es una estructura neuronal ubicada en la región subtalámica del cerebro, desempeñando un papel clave en los circuitos motores [16]. Este núcleo, esencial en la red de ganglios basales, ha atraído considerable atención en el contexto de la enfermedad de Parkinson y la Estimulación Cerebral Profunda (DBS). La conexión directa del STN con la corteza motora y su influencia en la modulación de la actividad neuronal lo convierten en un objetivo crucial para intervenir en los síntomas motores asociados con la enfermedad [17]. La comprensión detallada de las funciones y la fisiología del STN es esencial para evaluar los impactos terapéuticos de la DBS, destacando su relevancia en la investigación y tratamiento de trastornos neurológicos.

4.2. REDES NEURONALES CONVOLUCIONALES

Las CNN representan una innovadora rama de las redes neuronales artificiales, inspirada en la corteza visual del cerebro [18]. Aunque su desarrollo primario se orienta hacia la resolución de desafíos en visión artificial, como el reconocimiento de patrones en imágenes, su versatilidad se extiende a diversas aplicaciones. Estas incluyen la clasificación de textos, donde las CNN demuestran su capacidad para analizar y entender patrones en datos escritos, así como el procesamiento de lenguaje natural, facilitando la comprensión contextual del lenguaje humano [19]. Más allá de su origen en la imagenología, las CNN se revelan como herramientas adaptables, capaces de abordar problemas complejos en campos como la medicina, finanzas y análisis de datos multidimensionales. Su capacidad para extraer características y aprender patrones las convierte en una tecnología clave en la era de la inteligencia artificial [20].

Su funcionamiento, ilustrado en la Figura 1, se basa en la aplicación de operaciones de convolución, agrupación (pooling) y capas completamente conectadas. A continuación, se describe brevemente cada uno de estos componentes.

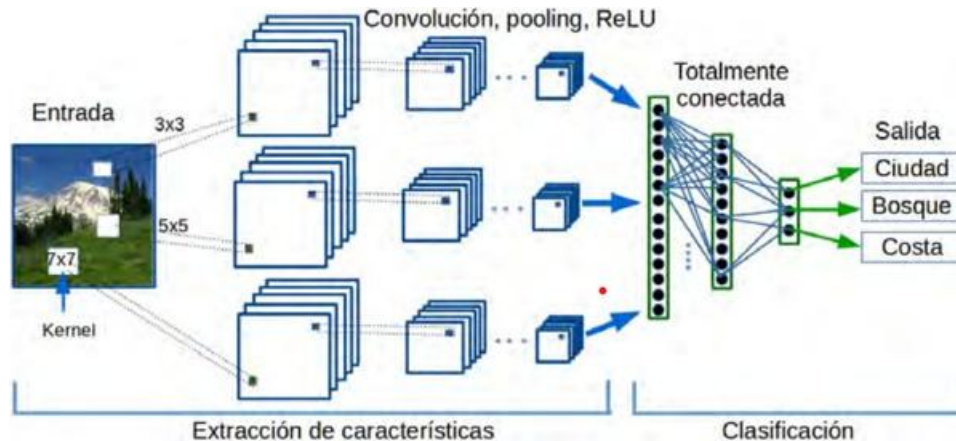


Figura 1. Estructura de una red neuronal convolucional [21].

4.2.1. Operaciones Convolucionales

Las operaciones de convolución son fundamentales en las CNN. Consisten en aplicar filtros a regiones locales de la entrada para detectar patrones específicos. Estos filtros se deslizan a lo largo de la entrada, generando mapas de características que resaltan detalles relevantes. La convolución permite capturar jerarquías de características, desde bordes simples hasta formas más complejas [22].

4.2.2. Capas de Agrupación (Pooling)

Las capas de agrupación (pooling) reducen la dimensionalidad de los mapas de características, preservando la información esencial. El pooling se realiza al seleccionar el valor máximo (max pooling) o promedio (average pooling) de pequeñas regiones en el mapa. Esto disminuye la carga computacional y mejora la tolerancia a las variaciones en la posición de las características [23].

4.2.3. Capas Completamente Conectadas

Después de múltiples capas de convolución y agrupación, se aplican capas completamente conectadas para realizar la clasificación final. Estas capas integran la información aprendida en las etapas anteriores y generan la salida final del modelo [24].

4.3. MÉTRICAS DE EVALUACIÓN

La evaluación del rendimiento de las redes neuronales convolucionales (CNN) es fundamental para validar su eficacia en tareas específicas. Las métricas de evaluación más utilizadas incluyen la matriz de confusión, la exactitud (accuracy), la precisión (precision), la sensibilidad o recuperación (recall), el F1-score y la curva ROC-AUC [25].

4.3.1. Matriz de confusión

la matriz de confusión proporciona una visión detallada de los resultados del modelo, delineando la cantidad de clasificaciones correctas e incorrectas para cada clase [25]. La matriz de confusión se muestra en la Tabla 1:

	Clase Real Positiva	Clase Real Negativa
Predicción Positiva	TP	FP
Predicción Negativa	FN	TN

Tabla 1. Definición de Matriz de confusión

Donde:

- **TP:** Verdaderos positivos (instancias positivas correctamente clasificadas).
- **TN:** Verdaderos negativos (instancias negativas correctamente clasificadas).
- **FP:** Falsos positivos (instancias negativas incorrectamente clasificadas como positivas).
- **FN:** Falsos negativos (instancias positivas incorrectamente clasificadas como negativas).

4.3.2. Exactitud (Accuracy)

Esta métrica mide la proporción de predicciones correctas en relación con el total de muestras y proporciona una visión general del rendimiento del modelo [25]. Se calcula mediante la siguiente fórmula:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

4.3.3. Precisión (Precision)

La precisión se enfoca en la proporción de verdaderos positivos en comparación con la suma de verdaderos positivos y falsos positivos. Es particularmente útil cuando se busca minimizar los falsos positivos [25]. Su fórmula es:

$$Precisión = \frac{TP}{TP + FP}$$

4.3.4. Recuperación (Recall)

La recuperación, también conocida como sensibilidad, mide la proporción de verdaderos positivos en comparación con la suma de verdaderos positivos y falsos negativos. Es especialmente relevante en situaciones donde es crucial identificar correctamente las instancias positivas [25]. Se calcula como:

$$Recall = \frac{TP}{TP + FN}$$

4.3.5. F1-score

El F1-score es la media armónica de la precisión y la recuperación, lo que resulta valioso cuando se busca un equilibrio entre ambas métricas, especialmente en casos de desequilibrio en los datos [25]. Su fórmula es:

$$F1 - score = \frac{2 * Precisión * Recall}{Precisión + Recall}$$

4.3.6. Curva ROC-AUC

La curva ROC (Receiver Operating Characteristic) es una gráfica que muestra la relación entre la tasa de verdaderos positivos (sensibilidad) y la tasa de falsos positivos, evaluando la capacidad del modelo para diferenciar entre clases. El área bajo la curva (AUC, Area Under the Curve) es una métrica que resume esta relación en un solo valor: cuanto más cerca esté de 1, mejor será el rendimiento del modelo en términos de clasificación [25].

4.4. TRANSFERENCIA DE CONOCIMIENTO

La transferencia de conocimiento (TL, Transfer Learning) es una técnica que consiste en aprovechar un modelo previamente entrenado en una tarea similar y aplicarlo a un nuevo problema, reutilizando sus características y conocimientos adquiridos. Esta técnica es especialmente útil cuando se cuenta con un conjunto de datos limitado para entrenar un modelo desde cero, ya que permite acelerar el proceso de entrenamiento y mejorar el rendimiento del modelo final [25].

Dentro de este contexto, dos modelos preentrenados ampliamente utilizados para la transferencia de conocimiento son VGG19 y ResNet50. VGG19 es una red neuronal convolucional profunda que cuenta con 19 capas y ha sido entrenada con millones de imágenes, permitiéndole extraer características visuales complejas y útiles para la clasificación de imágenes. Por otro lado, ResNet50 es una red residual que incluye 50 capas y utiliza conexiones residuales para abordar el problema de la degradación que ocurre en redes profundas, facilitando la transmisión de información entre capas y logrando un rendimiento superior en tareas de clasificación de imágenes [26] [27].

4.5. ANTECEDENTES

4.5.1. Representación óptima de señales MER aplicada a la identificación de estructuras cerebrales durante la estimulación cerebral profunda

Este artículo aborda una problemática clave en la cirugía de estimulación cerebral profunda (DBS), particularmente en la identificación precisa del núcleo subtalámico (STN) en pacientes con Parkinson. Los autores presentan un enfoque novedoso basado en el método de frames (MOF) para representar de manera óptima las señales MER. A través de este método, se logran extraer características relevantes combinando diccionarios de wavelet y coseno, lo que mejora significativamente los resultados en comparación con métodos tradicionales como la transformada wavelet discreta (DWT). La precisión alcanzada supera el 97,6%, y lo más importante, se logra reducir la tasa de falsos positivos a menos del 2%, un factor crucial en cirugías de alta precisión como esta. Este avance permite una localización más segura del STN, reduciendo riesgos y mejorando la eficacia terapéutica para los pacientes que se someten a DBS. [28].

4.5.2. El aprendizaje multipaciente aumenta la precisión para la identificación del núcleo subtalámico en la estimulación cerebral profunda

En este estudio, los autores exploran una técnica innovadora de aprendizaje multitarea para mejorar la identificación del STN durante la DBS en pacientes con Parkinson. La idea principal es aprovechar la información compartida entre diferentes pacientes para aumentar la precisión en la localización del STN. A diferencia de los métodos tradicionales, que tratan a cada paciente de manera independiente, este enfoque multitarea permite que el sistema "aprenda" de varios pacientes a la vez, lo que mejora la capacidad del algoritmo para identificar correctamente el STN en nuevas cirugías. Los resultados obtenidos en dos conjuntos de datos distintos muestran que esta estrategia no solo es viable, sino que supera considerablemente a los enfoques convencionales, lo que sugiere que compartir información entre pacientes puede ser clave para mejorar la precisión en estos procedimientos quirúrgicos [29].

4.5.3. Representación dispersa de señales MER para la localización del núcleo subtalámico en la cirugía de la Enfermedad de Parkinson

La investigación destaca un enfoque innovador para mejorar la identificación del STN durante la cirugía de DBS en pacientes con EP. Utilizando la codificación dispersa de las señales MER, se emplearon tres métodos: el Método de Marcos (MOF), la Mejor Base Ortogonal (BOB) y la Búsqueda de Base (BP). Estos métodos, que combinan los diccionarios Wavelet Packet y Cosine Packet, mostraron un desempeño superior en comparación con enfoques clásicos como la Transformada Wavelet (WT) y la Adaptativa Wavelet con Esquemas de Elevación (AW-LS). Los resultados de clasificación obtenidos con clasificadores supervisados simples alcanzaron precisiones superiores al 98%. Este enfoque demuestra que la codificación dispersa permite extraer características discriminantes de las señales MER, lo que mejora significativamente la identificación del STN durante la cirugía de DBS, destacando su potencial para el procesamiento de señales en aplicaciones clínicas avanzadas [30].

4.5.4. Representación óptima de señales MER mediante el método de Frames aplicado a la cirugía de la Enfermedad de Parkinson

En esta investigación se propone un enfoque de representación óptima de señales MER mediante el método de Frames, obteniendo coeficientes que minimizan la norma euclidiana. Utilizando una combinación de diccionarios Wavelet Packet y coseno, el objetivo fue identificar con precisión el STN, estructura crítica en la cirugía de DBS. La metodología fue validada con una base de datos clínica real, empleando clasificadores como K-Nearest Neighbors (K-NN) y Clasificadores Bayesianos Lineal (LDC) y Cuadrático (QDC), logrando una tasa de identificación positiva del STN del 97.6%. Este trabajo resalta la efectividad del método de Frames para describir el comportamiento dinámico de las señales MER y su capacidad para mejorar la caracterización entre clases, contribuyendo a un enfoque terapéutico más preciso [31].

4.5.5. Localización automática de electrodos de estimulación cerebral profunda en imágenes de tomografía computarizada: aplicación a la Enfermedad de Parkinson

En la investigación se aborda el problema de la correcta localización de los electrodos durante la DBS, un aspecto crucial para evitar efectos secundarios en los pacientes con Parkinson. Este estudio propone una metodología automatizada de segmentación de imágenes de tomografía computarizada (CT) mediante umbrales adaptativos, logrando localizar con precisión los electrodos DBS y minimizar la exposición a la radiación. Este enfoque permite reducir la repetición de procedimientos de toma de imágenes y mejorar la tolerancia al ruido en las imágenes. La metodología propuesta no solo es eficiente en términos de tiempo de procesamiento, sino que también tiene implicaciones clínicas importantes para asegurar una correcta estimulación de las áreas neuronales objetivo [32].

4.5.6. Algoritmos de clasificación lineal para la identificación de zonas cerebrales

En el presente trabajo, se propone la utilización de algoritmos de clasificación lineal basados en regresión logística para la identificación del STN durante la estimulación cerebral profunda en pacientes con Parkinson. A través del uso de vectores de características obtenidos de señales MER, se implementaron tres clasificadores lineales: Clasificador Bayesiano basado en múltiples clases, Método de Mínimos Cuadrados y Regresión Logística. La investigación destaca la simplicidad algorítmica de estos métodos, que, aunque menos explorados, ofrecen un enfoque eficaz para la clasificación de zonas cerebrales durante las cirugías de DBS [33].

4.5.7. Software para análisis de actividad neuronal en tiempo real durante neurocirugía

Este proyecto desarrolla un software innovador para la localización en tiempo real del target anatómico durante neurocirugías basadas en la DBS. Este software permite realizar análisis automatizados de microregistro neuronal, aplicando un análisis wavelet de 10 dimensiones y algoritmos de spike sorting basados en genética, lo que reduce el problema del overfitting. A pesar del desafío en el tiempo de procesamiento, el software ofrece una herramienta de apoyo para la toma de decisiones en tiempo real, con futuras posibilidades de integración con sistemas de adquisición de datos y adaptación a plataformas móviles [34].

4.5.8. Selección de características mediante un conjunto de paquetes de ondículas óptimas y máquina de aprendizaje: aplicación a señales de registro de microelectrodos (MER)

Se propone un novedoso método de selección de características basado en paquetes de ondículas óptimas y una máquina de aprendizaje (OWLM). Esta técnica utiliza algoritmos genéticos para optimizar las fases de descomposición de la Transformada Wavelet Packet (WPT), maximizando la separabilidad de clases mediante el criterio de Davies-Bouldin. Aplicado a señales MER, el método OWLM demostró ser superior a enfoques tradicionales como WT y WPT, logrando una tasa de clasificación del 99.14%, destacándose como una herramienta eficiente para el análisis de señales no estacionarias en aplicaciones neurológicas [35].

Las investigaciones revisadas como antecedentes se enfocan principalmente en la identificación del STN, haciendo especial énfasis en el procesamiento de señales mediante métodos matemáticos avanzados y, en su mayoría, utilizando modelos supervisados de machine learning. Sin embargo, la presente investigación introduce un enfoque novedoso al emplear modelos de Deep Learning para la identificación del STN. Este enfoque constituye el principal diferenciador con respecto a las investigaciones previas, ya que propone un análisis automático de las señales, lo que podría aportar mejoras significativas en la exactitud y eficiencia del proceso.

5. PREPARACIÓN DE LOS DATOS Y DEFINICIÓN DEL ENTORNO

5.1. CONFIGURACIÓN DEL ENTORNO DE TRABAJO

Para el desarrollo del proyecto se utilizó un computador portátil MSI THIN GF63 12VE con las siguientes características:

- Procesador (CPU): 12th Gen Intel(R) Core(TM) i5-12450H 2.00 GHz
- Memoria RAM: 16,0 GB
- Tarjeta gráfica (GPU): NVIDIA GeForce RTX 4050 Laptop GPU 6,0 GB
- Almacenamiento: SSD 477 GB

El lenguaje utilizado para desarrollar el código fue Python 3.10 (64 -bit) y las librerías principales fueron las siguientes:

- Pandas 2.0.0
- Numpy 1.24.2
- Matplotlib 3.7.1
- Tensorflow 2.10.0
- Keras 2.10.0

Para habilitar el uso de la GPU con TensorFlow, fue necesario instalar la última versión de los controladores de la tarjeta gráfica, así como el NVIDIA GPU Computing Toolkit v11.2 y NVIDIA CUDA v11.2.

El entorno de desarrollo integrado (IDE) utilizado para desarrollar los modelos fue Visual Studio Code, en su versión 1.93.0.

5.2. DESCRIPCIÓN DEL CONJUNTO DE DATOS

Para el desarrollo del proyecto, se utilizó una base de datos compuesta por 600 registros de actividad cerebral, tanto del STN como de otras estructuras cerebrales, provenientes de 6 pacientes. Cada registro tiene una duración de 1 segundo y consta de 25,000 muestras (frecuencia de muestreo de 25 kHz). De cada paciente, se obtuvieron 50 registros del STN y 50 registros de otras estructuras cerebrales, los cuales fueron almacenados en archivos con extensiones .mat o .bin. Es importante señalar que la base de datos fue obtenida de la Universidad Tecnológica de Pereira [36] y es de acceso público. En la Figura 2 se muestra un ejemplo de la señal MER del STN de uno de los pacientes, mientras que en la Figura 3 se presenta la señal MER de otra estructura cerebral del mismo paciente.

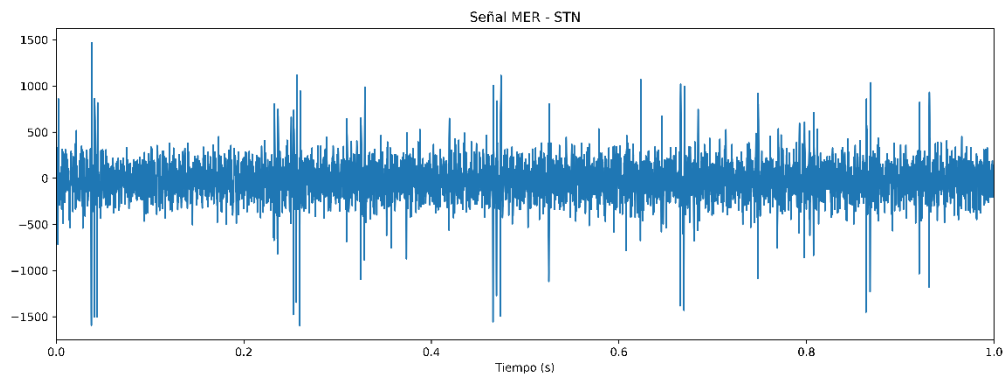


Figura 2. Señal MER del STN.

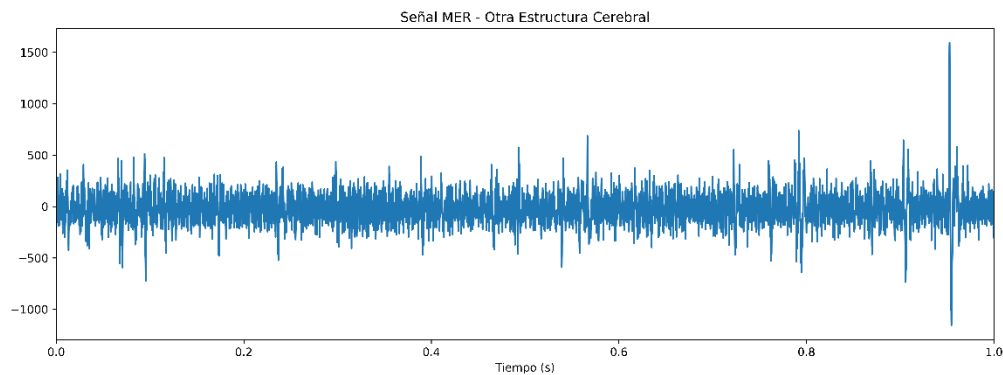


Figura 3. Señal MER de otra estructura cerebral.

5.3. PREPROCESAMIENTO DE DATOS

Para procesar los datos, el primer paso fue extraer el contenido de los archivos. Para los archivos con extensión `.mat`, se utilizó la función `loadmat` de la biblioteca `scipy`. Sin embargo, no fue posible extraer los datos de los archivos con extensión `.bin` debido a que se desconocía la estructura con la que habían sido almacenados. Esto impidió obtener los datos de 2 pacientes, reduciendo el número total de registros de 600 a 400 (200 registros del STN y 200 registros de otras estructuras cerebrales). Debido a esta reducción, se decidió dividir cada señal original de 1 segundo de duración en 10 partes iguales, resultando en nuevos registros de 100 ms de duración con un total de 2,500 muestras por registro. Como resultado, la cantidad de registros se incrementó de 400 a 4,000.

Para cambiar la forma de los registros, se creó una función que recibía como parámetro la ruta al archivo a procesar. La función abría el archivo, extraía los datos, cambiaba la forma de los datos con la función `reshape` de la biblioteca `numpy`, y finalmente retornaba un dataframe de `pandas` con los datos reestructurados. Esta función fue utilizada dentro de dos bucles `for`. El primer bucle leía todos los archivos del STN, aplicaba la función para cambiar la forma de los datos y concatenaba cada dataframe devuelto en un dataframe destinado a almacenar únicamente la información del STN. Luego de procesar todos los archivos del STN, se añadió una columna llamada "estructura" en el dataframe, donde se asignaba la etiqueta 1 para indicar que todos los registros correspondían al STN.

Para procesar la información de las demás estructuras cerebrales, se siguió el mismo procedimiento, con la única diferencia de que la etiqueta de la columna "estructura" en el dataframe creado para almacenar esta información fue 0. Posteriormente, ambos dataframes se concatenaron en uno solo y se mezclaron los registros para asegurar una distribución equitativa. Esto resultó en un dataframe final con 4,000 filas y 2,501 columnas (2,500 columnas correspondientes a las muestras y una columna para las etiquetas), el cual se guardó en un archivo CSV. La Figura 4 muestra una parte del contenido del dataframe generado.

muestra 1	muestra 2	muestra 3	muestra 4	muestra 5	muestra 6	muestra 7	muestra 8	muestra 9	muestra 10	muestra 2492	muestra 2493	muestra 2494	muestra 2495	muestra 2496	muestra 2497	muestra 2498	muestra 2499	muestra 2500	estructura
-89	196	90	264	54	191	-150	-2	-108	87	26	318	185	420	65	181	-17	148	180	1
-103	-45	-12	-26	-13	53	161	217	84	-125	-48	124	173	189	119	10	-33	11	37	0
-393	-395	-362	-330	-353	-308	-233	-53	83	191	-116	-176	-250	-181	-95	-66	-97	-87	-140	1
-9	13	1	39	-37	-140	-157	-162	-107	-63	144	249	265	204	84	-29	-60	-44	-94	1
-171	-87	-437	-270	-230	-141	-102	-204	-93	-296	252	404	238	152	70	-214	-221	-511	-358	1
...
-106	-55	-20	-40	-143	-233	-250	-258	-301	-284	-267	-246	-298	-370	-425	-390	-380	-387	-290	0
193	228	136	52	-41	-81	-147	-198	-234	-293	145	82	-29	-52	-34	-28	-34	-58	-43	1
118	298	186	189	5	125	-82	-148	-350	-500	-175	-78	-128	-51	-123	-133	-100	-115	-60	1
-56	-110	-61	133	-76	-2	-120	11	-16	291	-82	-212	-291	-129	-178	-65	-272	-41	-156	0
352	-13	-39	-193	61	-49	-244	-385	-323	-58	295	236	448	349	339	260	324	324	161	0

Figura 4. Dataframe generado.

Dado que uno de los objetivos del proyecto es identificar el STN mediante modelos convolucionales en una y dos dimensiones, se utilizó el dataframe con las señales etiquetadas para generar sus representaciones en imágenes conocidas como escalogramas. Para realizar esto, se aplicó la Transformada Wavelet Continua (CWT, Continuous Wavelet Transform), que es una herramienta matemática que descompone una señal en diferentes escalas y tiempos, permitiendo un análisis detallado de sus componentes frecuenciales y temporales simultáneamente [37]. Estos componentes son representados gráficamente en una imagen denominada escalograma, que muestra cómo la escala de una señal varía a lo largo del tiempo [38]. La CWT fue aplicada utilizando tres funciones madre: Wavelet Derivada Gaussiana (CGAU, Gaussian Derivative Wavelet), Wavelet de Morlet Compleja (CMOR, Complex Morlet Wavelet) y Wavelet Sombrero Mexicano (MEXH, Mexican Hat Wavelet). Para generar los escalogramas de las señales, el dataframe generado anteriormente se dividió en dos: uno que contenía los registros del STN y otro con los registros de las otras estructuras cerebrales. Posteriormente, se desarrolló una función encargada de aplicar la CWT y generar los escalogramas. Para calcular la CWT se utilizó la función *cwt* de la librería PyWavelets y la generación de los escalogramas se realizó utilizando la función *imshow* de Matplotlib. La función desarrollada recibía como parámetros la función madre, el registro o señal a transformar, la ruta de almacenamiento y un índice numérico para nombrar los escalogramas. la Figura 5 presenta la estructura de directorios creada para almacenar los escalogramas generados.

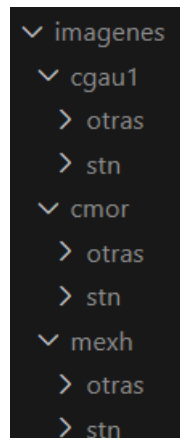


Figura 5. Estructura de directorios para almacenar escalogramas

Donde "imagenes" es el directorio raíz que contiene todos los escalogramas generados. El subdirectorio "cgau1" almacena los escalogramas generados con la función madre CGAU; el subdirectorio "cmor" contiene los escalogramas generados con la función madre CMOR; y el subdirectorio "mexh" almacena los escalogramas generados con la función madre MEXH. Las carpetas "otras" contienen los escalogramas correspondientes a otras estructuras cerebrales, mientras que las carpetas "stn" almacenan los escalogramas del STN. En las Figuras 6, 7 y 8 se muestran los escalogramas generados por las funciones madre CGAU, CMOR y MEXH, respectivamente, para una señal proveniente del STN.

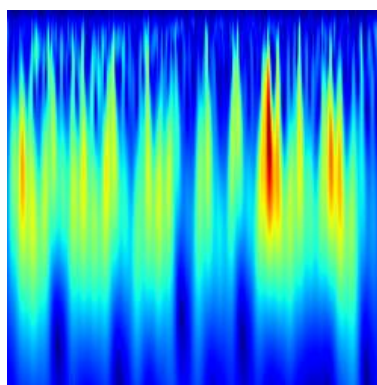


Figura 6. Escalograma del STN generado con la función madre CGAU.

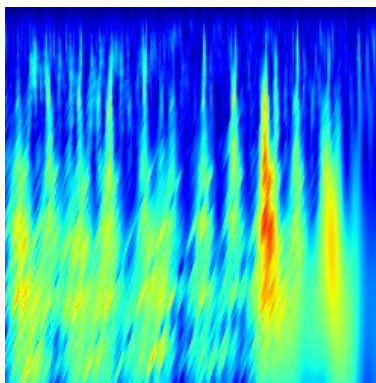


Figura 7. Escalograma del STN generado con la función madre CMOR.

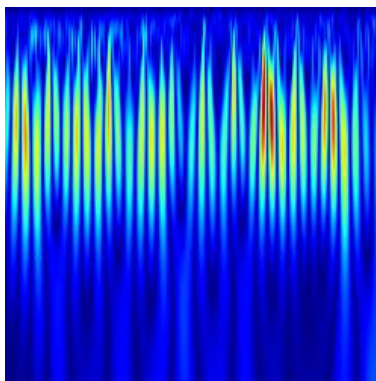


Figura 8. Escalograma del STN generado con la función madre MEXH.

Es importante destacar que se generaron un total de 12,000 escalogramas: 4,000 escalogramas utilizando la función madre CGAU, 4,000 con la función madre CMOR, y 4,000 con la función madre MEXH. Cada escalograma corresponde a una señal del STN o de alguna otra estructura cerebral. En total, la base de datos de escalogramas tiene un tamaño de 533 MB.

6. DESARROLLO DE MODELOS

En esta sección se describe el diseño, la implementación y el entrenamiento de los modelos desarrollados. La creación de los modelos se dividió en dos categorías principales: modelos CNN 1D y modelos CNN 2D. Se desarrollaron cinco modelos CNN 1D, los cuales fueron entrenados y validados utilizando el *dataframe* generado durante el procesamiento de las señales. La base de datos creada a partir de los escalogramas de las señales se empleó para entrenar y validar los modelos CNN 2D. Los modelos CNN 2D se subdividieron en dos categorías: modelos CNN 2D sin TL y modelos CNN 2D con TL. Las arquitecturas preentrenadas seleccionadas fueron ResNet50 y VGG19. En total, se desarrollaron 27 modelos CNN 2D. La Figura 9 presenta un resumen de todos los modelos creados.

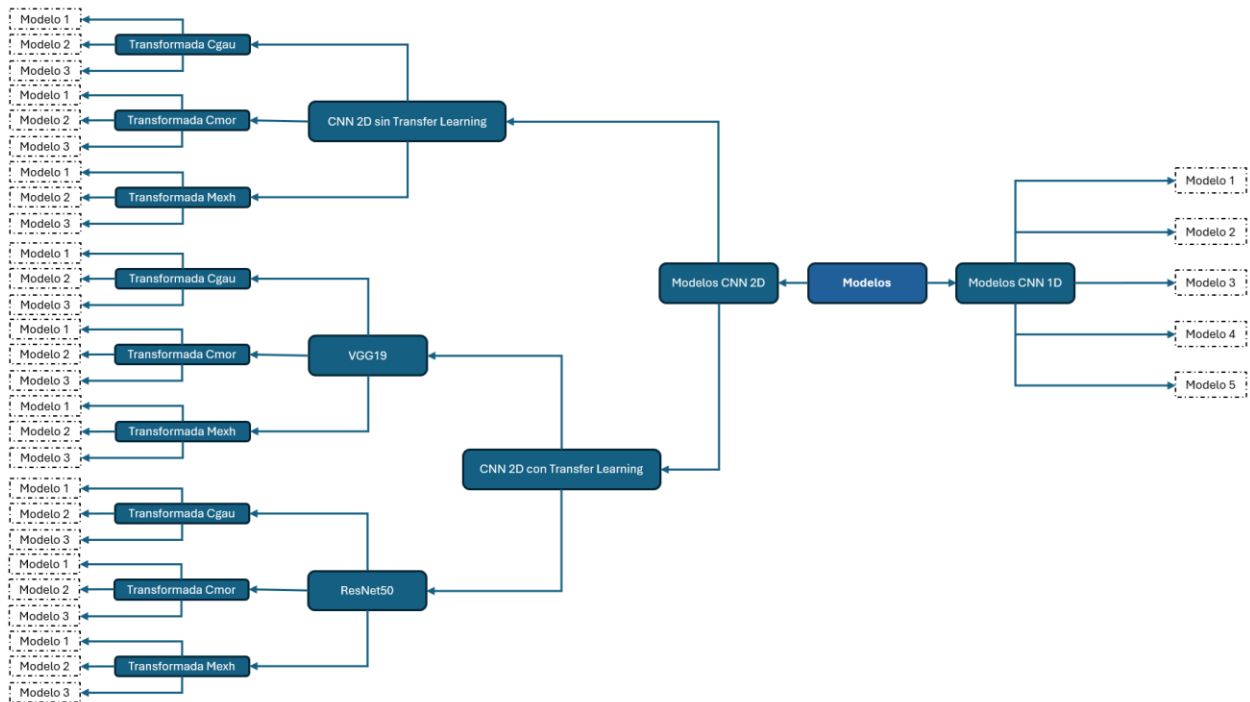


Figura 9. Resumen de los modelos creados.

6.1. CREACIÓN DE LOS MODELOS

6.1.1. MODELOS CNN 1D

El primer paso en la creación de los modelos CNN 1D fue cargar el dataframe generado durante el procesamiento de las señales, el cual se dividió en conjuntos de entrenamiento y validación utilizando la función *train_test_split* de la librería Scikit-learn. Se asignó el 80% de los datos al entrenamiento y el 20% a la validación. Dado el número limitado de datos, no se realizó una división adicional para pruebas. Las dimensiones de los conjuntos quedaron de la siguiente manera:

Dimensiones X_train: (3200, 2500)

Dimensiones y_train: (3200,)

Dimensiones X_val: (800, 2500)

Dimensiones y_val: (800,)

El siguiente paso fue escalar los datos y modificar su forma para adaptarlos a la entrada de los modelos CNN 1D. En Keras y la mayoría de los frameworks de aprendizaje profundo, la entrada de los modelos CNN 1D debe tener tres dimensiones: el tamaño del lote (batch size), la longitud de la secuencia (input length), y el número de características (input channels) [39]. Por lo tanto, al cambiar la forma de los datos de entrada, las dimensiones resultantes fueron:

Dimensiones X_train: (3200, 2500, 1)

Dimensiones X_val: (800, 2500, 1)

Con los datos preparados, se procedió a crear y probar diferentes estructuras para los modelos CNN 1D. Todos los modelos desarrollados incluyeron capas convolucionales, capas de Max Pooling, una capa de Flatten, capas densas intermedias y una capa de salida densa con función de activación sigmoide. Para el entrenamiento, se utilizó el optimizador Adam, la función de pérdida de entropía cruzada y, como métrica de evaluación, el accuracy. Además, se implementó un callback para guardar el modelo que lograra el mayor accuracy en la validación. Durante el desarrollo, se probaron diversas configuraciones, variando el número de capas convolucionales, capas de Max Pooling, capas densas intermedias, y capas de Dropout. También se exploraron diferentes valores de hiperparámetros, como la cantidad de filtros por capa, el tamaño del kernel y del pool, valores de regularización, tasas de aprendizaje, número de épocas de entrenamiento y diferentes tamaños de batch, hasta encontrar las estructuras que ofrecían los mejores resultados. En el Anexo A se presentan las estructuras de los modelos CNN 1D desarrollados.

6.1.2. MODELOS CNN 2D SIN TRANSFER LEARNING

Para la creación de los modelos CNN 2D sin TL, se siguió el mismo enfoque que en los modelos CNN 1D, diferenciándose únicamente en la forma de cargar los datos y generar los conjuntos de entrenamiento y validación. Es importante destacar que, mientras que en los modelos CNN 1D se trabajó con los valores de las señales, en los modelos CNN 2D se utilizó la base de datos de los escalogramas de las señales, generados mediante la CWT y sus diferentes funciones madre. Por lo tanto, el procesamiento de los datos para que fueran reconocidos por los modelos CNN 2D fue distinto al empleado en los modelos CNN 1D.

El primer paso consistió en definir con cuales escalogramas trabajaría el modelo, para lo cual se seleccionó la función madre correspondiente. A continuación, se estableció la ruta al directorio donde se encuentran almacenados los escalogramas y la ruta del directorio donde se almacenaría la división de los escalogramas para entrenamiento y validación. Estas rutas se generaron dinámicamente utilizando el nombre de la función madre seleccionada.

Antes de realizar la división de las imágenes, el código verifica si el directorio de destino ya existe. Si es así, se elimina el directorio junto con todo su contenido, para garantizar que la nueva división se realice desde cero y no sobre datos antiguos.

Finalmente, se empleó la función *ratio* de la librería Splitfolders para dividir las imágenes en conjuntos de entrenamiento y validación. Es relevante destacar que la función *ratio* funciona de manera similar a *train_test_split* de Scikit-learn, pero está específicamente diseñada para particionar conjuntos de datos compuestos por imágenes. Para realizar la partición, se estableció que el 80% de los escalogramas se utilizara para el entrenamiento y el 20% restante para la validación. A continuación, se presenta el código que ejecuta las acciones previamente descritas:

```
# Se selecciona la función madre que generó los escalogramas
transformada = "cgau1"

# Ruta al directoria que contiene los escalogramas
ruta_imagenes = Path(f"../.././imagenes/{transformada}")
# Ruta al directoria que almacenará la división de los escalogramas
ruta_split_imagenes = Path(f"../.././split_imagenes/{transformada}")

# Si el directorio ya existe, eliminarlo junto con su contenido
if ruta_split_imagenes.exists():
    shutil.rmtree(ruta_split_imagenes)

# Crear el split de los escalogramas
splitfolders.ratio(input=ruta_imagenes, output=ruta_split_imagenes, ratio=(.8,.2))
```

A continuación, la Figura 10 muestra la estructura de directorios creada para almacenar la división de los escalogramas en los grupos de entrenamiento y validación.

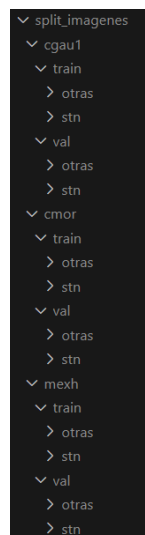


Figura 10. Estructura de directorios para la división de los escalogramas en los conjuntos de entrenamiento y validación.

La carpeta "split_imagenes" es la raíz que contiene los escalogramas generados, y dentro de ella se encuentran subcarpetas correspondientes a las tres funciones madre utilizadas para su generación: "cgau1", "cmor" y "mexh". Cada una de estas subcarpetas contiene dos directorios principales, "train" y "val", que representan los conjuntos de entrenamiento y validación, respectivamente. A su vez, dentro de estas carpetas se encuentran las subcarpetas "otras" y "stn," que contienen los escalogramas correspondientes a las otras estructuras cerebrales (subcarpeta "otras") y al STN (subcarpeta "stn").

Posterior a la división de los datos en los conjuntos de entrenamiento y validación, se definieron varios parámetros clave que fueron utilizados en todos los modelos CNN 2D, tanto en los no preentrenados como en los preentrenados. Los valores de ancho y alto de las imágenes se establecieron en 250x250 píxeles (img_width y img_height), y el formato de entrada (input_shape) se configuró como (250, 250, 3) para representar imágenes en color. El número de épocas de entrenamiento se fijó en 100, y el batch_size fue de 32, lo que indicaba que se procesarían 32 imágenes en cada iteración durante el entrenamiento de los modelos.

A continuación, se crearon los generadores de datos de imágenes, que permitieron cargar las imágenes en lotes y aplicar una normalización en los valores de los píxeles en tiempo real. Para el conjunto de entrenamiento y validación, se utilizó la función *ImageDataGenerator* con el parámetro `rescale=1./255`, que normalizó los valores de los píxeles en un rango de 0 a 1.

El generador de imágenes para el entrenamiento cargó las imágenes desde el subdirectorio "train" dentro de la ruta previamente definida. Las imágenes se redimensionaron a 250x250 píxeles y se agruparon en lotes de 32 imágenes. Se activó el parámetro `shuffle=True`, lo que permitió que las imágenes se mezclaran aleatoriamente en cada época, evitando que el modelo aprendiera patrones específicos del orden de los datos.

De manera similar, se configuró el generador de validación para cargar imágenes desde el subdirectorio "val", manteniendo las mismas dimensiones de 250x250 píxeles y un lote de 32 imágenes. Sin embargo, en este caso, el parámetro `shuffle=False` fue empleado para que las imágenes se procesaran en el orden original, facilitando una evaluación consistente del modelo durante la validación. A continuación, se presenta el código utilizado para procesar los datos y asegurarse de que fueran compatibles con la entrada de los modelos:

```
# Definición de parámetros
img_width, img_height = 250, 250
input_shape            = (img_width, img_height, 3)
epochs                 = 100
batch_size             = 32

# Configurar el generador de datos de imágenes
train_datagen = ImageDataGenerator(rescale=1./255)
valid_datagen = ImageDataGenerator(rescale=1./255)

# Cargar imágenes de entrenamiento
generador_entrenamiento = train_datagen.flow_from_directory(
    ruta_split_imagenes/"train",
    target_size = (img_width, img_height),
    batch_size  = batch_size,
    class_mode  = 'binary',
    shuffle     = True
)

# Cargar imágenes de validación
generador_validacion = valid_datagen.flow_from_directory(
    ruta_split_imagenes/"val",
    target_size = (img_width, img_height),
    batch_size  = batch_size,
    class_mode  = 'binary',
    shuffle     = False
)
```

La metodología empleada para desarrollar las diferentes estructuras de los modelos fue similar a la utilizada en los modelos CNN 1D. Se realizaron pruebas con diferentes cantidades de capas convolucionales, capas de maxpooling, y capas densas, entre otras. Además, se probaron distintas combinaciones de hiperparámetros con una variedad de valores. Las mejores estructuras obtenidas para los modelos CNN 2D no preentrenados se detallan en el **Anexo B**.

6.1.3. MODELOS CNN 2D CON TRANSFER LEARNING

La creación de los modelos CNN 2D con TL siguió el mismo proceso que los modelos CNN 2D sin TL, diferenciándose únicamente en el desarrollo de la estructura. En este proceso, se importó el modelo preentrenado VGG19 o ResNet50, dependiendo del caso.

Primero, se cargó el modelo preentrenado utilizando los pesos del conjunto de datos ImageNet [40]. El modelo se cargó sin las capas superiores (`include_top=False`), lo que permitió personalizar la parte final de la arquitectura para adaptarla a la tarea específica de clasificación. Además, se definió la forma de entrada mediante el parámetro `input_shape`.

A continuación, se congelaron las capas del modelo preentrenado para evitar la actualización de sus pesos, estableciendo el atributo `trainable=False` en cada una de ellas. De este modo, el modelo desarrollado pudo aprovechar las características preentrenadas sin modificarlas.

Posteriormente, se completó la construcción del modelo probando diferentes cantidades de capas densas y variando el número de neuronas en cada capa. Finalmente, el modelo se completó con una capa densa final que actuó como la salida del modelo. El siguiente código es un ejemplo que resume el proceso descrito anteriormente para la creación de uno de los modelos, utilizando el modelo preentrenado VGG19.

```
# Cargar el modelo VGG19 pre-entrenado sin la parte superior (include_top = False)
vgg19_model = VGG19(weights='imagenet', include_top=False, input_shape=input_shape)

# Construir el modelo
model = Sequential()
model.add(vgg19_model)
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(1, activation='sigmoid'))

# Compilar el modelo
model.compile(optimizer=Adam(learning_rate=0.001),
              loss='binary_crossentropy',
              metrics=['accuracy'])
```

Las estructuras de los modelos creados con VGG19 se presentan en el Anexo C, mientras que las estructuras de los modelos creados con ResNet50 se encuentran en el Anexo D.

6.1.4. ENTRENAMIENTO DE LOS MODELOS

Como se mencionó en la sección 6.1.1, debido a la cantidad limitada de datos disponibles, no se realizó la triple partición (entrenamiento, validación y prueba); por lo tanto, solo se trabajó con conjunto de entrenamiento y de validación tanto para los modelos CNN 1D como para los modelos CNN 2D. Para asegurar una evaluación robusta del rendimiento, se realizó un entrenamiento con aleatoriedad no controlada, ejecutando cinco veces cada notebook donde se definían las estructuras. Dado que no se definió una semilla de reproducibilidad (el parámetro *random_state* en *train_test_split* y *ratio*), en cada ejecución se generaron de forma aleatoria diferentes conjuntos de datos de entrenamiento y validación, lo que permitió evaluar los modelos con distintas proporciones de las bases de datos.

Durante cada ejecución de los notebooks, se monitoreó que no presentara sobreajuste o subajuste, graficando el comportamiento de la exactitud (accuracy) y la pérdida (loss) para cada época de entrenamiento en los conjuntos de entrenamiento y validación. En la Figura 11 se muestra el rendimiento de una de las CNN 1D durante el entrenamiento para ambos conjuntos. En la gráfica se observa cómo la exactitud del conjunto de entrenamiento aumenta a lo largo de las épocas, alcanzando un nivel cercano al 90%, lo que indica que la red está aprendiendo patrones relevantes de los datos. La exactitud en el conjunto de validación sigue una tendencia similar, aunque con mayores fluctuaciones. La pequeña brecha entre la exactitud de los conjuntos de entrenamiento y validación sugiere que el modelo está generalizando bien. En la gráfica de pérdida (a la derecha), se observa que la pérdida en el conjunto de entrenamiento comienza en un valor alto y disminuye rápidamente durante las primeras 20-30 épocas, lo que indica un aprendizaje rápido de los patrones básicos. La pérdida en el conjunto de validación también disminuye de manera similar, demostrando una mejora en la capacidad de generalización. Para obtener información detallada sobre el rendimiento de las diferentes arquitecturas desarrolladas durante el entrenamiento, se puede consultar el Anexo E, donde se encuentra el enlace al repositorio del proyecto.

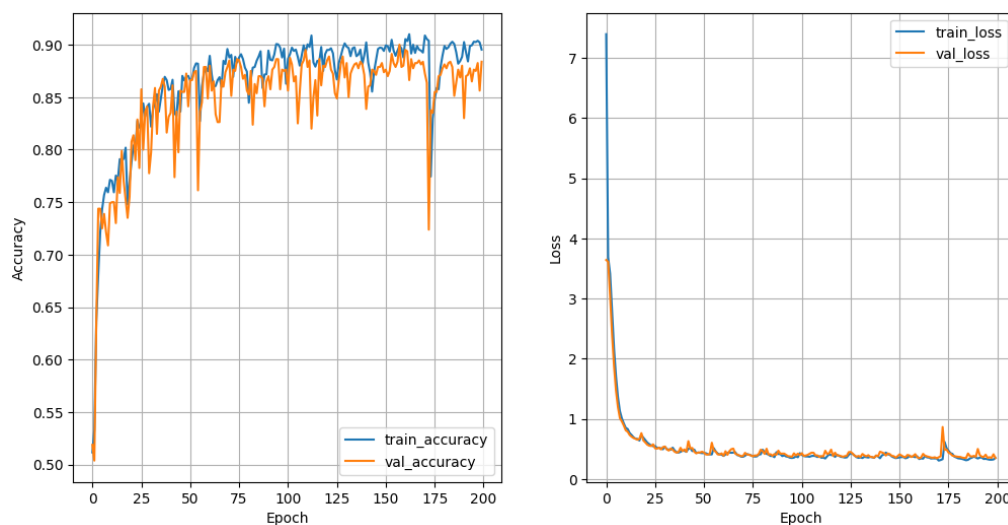


Figura 11. Curva de evaluación de la exactitud y perdida durante el entrenamiento.

Además de monitorear el comportamiento de los modelos durante el entrenamiento, en cada ejecución de los notebooks se registraron los resultados de las diferentes métricas de evaluación en un archivo de Excel. Las métricas registradas fueron: exactitud (accuracy), sensibilidad (sensitivity), especificidad (specificity), precisión (precision), puntuación F1 (F1 score) y el área bajo la curva (AUC). Una vez obtenidos todos los resultados, se calculó el promedio y la desviación estándar para determinar el desempeño general de los modelos. En la Tabla 2 se presentan los resultados del modelo que generó la Figura 11. En los Anexos F, G, H e I se presentan los resultados de todos los modelos desarrollados.

Modelo 1						
# Rep. Exp.	Accuracy	Sensitivity	Specificity	Precision	F1	Auc
1	0,91	0,99	0,82	0,86	0,92	0,95
2	0,87	0,97	0,78	0,80	0,88	0,93
3	0,91	0,93	0,88	0,88	0,91	0,96
4	0,90	0,96	0,84	0,85	0,90	0,96
5	0,90	0,98	0,81	0,83	0,90	0,93
Promedio	0,90	0,97	0,83	0,84	0,90	0,94
Desv	0,01	0,02	0,04	0,03	0,01	0,02

Tabla 2. Resultados de las métricas de evaluación

7. ANÁLISIS DE RESULTADOS

Este capítulo presenta un análisis detallado del rendimiento de los modelos desarrollados durante el proyecto. En las siguientes secciones, se evaluará el desempeño de los modelos, se compararán sus resultados, se interpretarán los hallazgos obtenidos, y se discutirán las limitaciones y consideraciones que surgieron durante la investigación.

7.1. EVALUACIÓN DEL RENDIMIENTO DE LOS MODELOS

Dado que el objetivo principal de esta investigación es identificar automáticamente el STN a través de diversas arquitecturas de CNN, se han seleccionado como métricas principales para evaluar el rendimiento la precisión, la sensibilidad y la exactitud. La precisión es prioritaria porque refleja la confianza del sistema al identificar correctamente el STN, minimizando los falsos positivos, lo cual es crucial en este contexto clínico. La sensibilidad, por su parte, asegura la detección de la mayor cantidad posible de casos de STN, reduciendo el riesgo de falsos negativos. Finalmente, la exactitud proporciona una visión global del desempeño. Al combinar estas métricas, se logra una evaluación más completa del rendimiento, permitiendo seleccionar aquella configuración que optimice tanto la detección del STN como la minimización de errores críticos, garantizando así su eficacia como herramienta de apoyo a los profesionales de la salud.

7.1.1. RENDIMIENTO DE LOS MODELOS CNN 1D

La Tabla 3 presenta los valores promedio de cada métrica para todos los modelos CNN 1D desarrollados.

	Modelos CNN 1D											
	Accuracy		Sensitivity		Specificity		Precision		F1		Auc	
	Pro (%)	Desv (%)	Pro (%)	Desv (%)	Pro (%)	Desv (%)	Pro (%)	Desv (%)	Pro (%)	Desv (%)	Pro (%)	Desv (%)
Modelo 1	90,15	0,45	94,90	4,39	85,42	4,46	86,57	2,98	90,43	0,82	95,21	1,31
Modelo 2	91,08	1,77	98,81	0,70	83,30	3,32	85,69	3,06	91,76	1,71	96,61	0,86
Modelo 3	90,90	0,83	97,47	2,26	84,10	1,88	86,46	1,50	91,61	0,83	95,77	0,44
Modelo 4	88,10	2,99	97,55	0,55	78,47	6,47	82,25	3,47	89,22	2,14	94,15	0,77
Modelo 5	83,65	5,49	93,91	4,83	73,44	7,44	78,12	4,71	85,26	4,42	87,90	6,22

Tabla 3. Valores promedio de métricas para los modelos CNN 1D.

Al observar la Tabla 3, si se considerara únicamente la precisión, el Modelo 1 sería seleccionado como el mejor, con un valor de 86,57%. Sin embargo, al analizar el Modelo 2, se observa que, aunque su precisión es un 1% menor, su sensibilidad aumenta significativamente del 94,90% (en el Modelo 1) al 98,81%, lo que implica una mayor capacidad para detectar la clase positiva. Además, la exactitud del Modelo 2 es del 91,08%, superando en un 1% a la del Modelo 1. Al comparar el Modelo 2 con el Modelo 3, este último podría competir en rendimiento, pero el Modelo 2 sigue siendo superior en términos de sensibilidad. Por lo tanto, para los modelos CNN 1D desarrollados, el Modelo 2 se presenta como la mejor opción general, equilibrando de manera efectiva precisión, sensibilidad y exactitud.

7.1.2. RENDIMIENTO DE LOS MODELOS CNN 2D SIN TRANSFER LEARNING

La Tabla 4 presenta los valores promedio de cada métrica para todos los modelos CNN 2D sin TL desarrollados.

		Modelos CNN 2D SIN TRANSFER LEARNING											
		Accuracy		Sensitivity		Specificity		Precision		F1		Auc	
		Pro (%)	Desv (%)	Pro (%)	Desv (%)	Pro (%)	Desv (%)	Pro (%)	Desv (%)	Pro (%)	Desv (%)	Pro (%)	Desv (%)
CGAU	Modelo 1	65,00	8,41	53,25	29,88	78,75	11,95	57,09	31,92	54,50	30,48	72,66	3,77
	Modelo 2	68,57	2,76	72,80	3,15	63,95	3,05	67,29	2,47	69,68	2,88	73,05	4,46
	Modelo 3	81,33	0,47	84,65	2,10	78,20	1,57	79,60	0,76	82,29	0,83	88,56	0,64
CMOR	Modelo 1	79,93	0,43	84,00	0,87	75,65	1,65	77,69	0,99	80,98	0,27	86,67	0,52
	Modelo 2	78,15	2,24	82,40	6,19	72,45	3,89	75,01	1,38	78,50	2,28	82,87	0,25
	Modelo 3	80,62	0,54	84,25	3,15	76,95	4,14	78,69	2,63	81,24	0,28	86,84	0,66
MEXH	Modelo 1	80,65	0,54	82,50	2,42	78,75	2,59	80,19	1,71	80,98	0,62	87,21	0,53
	Modelo 2	70,33	1,17	73,85	5,04	68,75	6,16	70,13	2,40	72,01	1,05	76,34	1,05
	Modelo 3	80,83	1,24	86,50	2,27	75,00	1,29	77,77	0,62	81,80	1,00	87,53	0,98

Tabla 4. Valores promedio de métricas para los modelos CNN 2D sin TL.

Al observar la Tabla 4, es evidente que el Modelo 3, desarrollado con los escalogramas de la función madre CGAU, es superior a los otros dos modelos generados con la misma función madre. De manera similar, para la función madre CMOR, el Modelo 3 también se destaca, superando a los otros dos modelos en las métricas clave. En cuanto a la función madre MEXH, aunque el Modelo 1 presenta la mayor precisión (80,19%) entre los modelos desarrollados con esta función, al comparar con el Modelo 3, su precisión es solo 2,42% inferior, pero su combinación de exactitud (80,83%) y sensibilidad (86,50%) lo convierten en la opción más sólida. Por lo tanto, el Modelo 3 es el mejor entre los modelos desarrollados con la función madre MEXH.

Al comparar los mejores modelos de cada función madre, se observa que el Modelo 3 de la función madre CGAU se destaca como el mejor entre los modelos CNN 2D sin TL. Aunque los tres modelos comparten una exactitud similar, el Modelo 3 con CGAU presenta la mayor precisión con un 79,60%. En cuanto a la sensibilidad, solo es superado ligeramente por el modelo desarrollado con la función madre MEXH, con una diferencia de apenas un 1,85%.

7.1.3. RENDIMIENTO DE LOS MODELOS CNN 2D CON VGG19

La Tabla 5 presenta los valores promedio de cada métrica para todos los modelos CNN 2D con preentrenamiento desarrollados con VGG19.

Modelos CNN 2D CON TRANSFER LEARNING (VGG19)													
		Accuracy		Sensitivity		Specificity		Precision		F1		Auc	
		Pro (%)	Desv (%)	Pro (%)	Desv (%)	Pro (%)	Desv (%)	Pro (%)	Desv (%)	Pro (%)	Desv (%)	Pro (%)	Desv (%)
CGAU	Modelo 1	86,40	0,82	88,20	3,37	84,90	2,90	85,68	2,02	87,09	0,81	92,03	2,32
	Modelo 2	86,92	0,31	92,25	2,13	80,85	1,93	83,45	1,28	87,64	0,45	94,44	0,15
	Modelo 3	86,62	0,28	90,40	2,36	82,66	2,08	84,05	1,27	87,19	0,30	92,80	1,88
CMOR	Modelo 1	83,18	0,82	89,50	1,98	77,40	1,52	80,23	0,94	84,62	0,34	88,98	2,19
	Modelo 2	84,15	0,30	89,81	1,03	78,84	1,21	80,93	0,74	85,09	0,11	91,92	0,24
	Modelo 3	84,30	0,44	90,48	0,20	78,44	0,46	80,76	0,33	85,35	0,21	92,13	0,22
MEXH	Modelo 1	85,27	5,33	95,29	1,31	78,90	1,14	81,94	0,85	88,14	0,92	89,02	0,87
	Modelo 2	87,62	0,62	93,95	1,84	81,25	2,46	83,43	1,64	88,48	0,37	90,63	2,23
	Modelo 3	88,05	0,45	94,25	3,11	82,35	4,35	84,31	2,71	88,80	0,08	91,20	2,67

Tabla 5. Valores promedio de métricas para los modelos CNN 2D con TL (VGG19).

Para los modelos desarrollados con la función madre CGAU, el Modelo 1 presenta la mayor precisión (85,68%), lo que lo convierte en una buena opción si se busca minimizar los falsos positivos. Sin embargo, el Modelo 2 se destaca por su sensibilidad (92,25%), siendo el mejor para detectar correctamente los casos positivos. Dado que tanto la precisión como la sensibilidad son métricas clave en este análisis, el Modelo 3 es la opción más equilibrada, con una precisión del 84,05% y una sensibilidad del 90,40%.

En el caso de la función madre CMOR, el Modelo 2 y el Modelo 3 son prácticamente idénticos en precisión y sensibilidad, lo que los convierte en opciones igualmente adecuadas cuando se busca un equilibrio entre ambas métricas.

Para los modelos desarrollados con la función madre MEXH, el Modelo 3 es el mejor en términos generales, con la mayor precisión (84,31%) y una sensibilidad también muy alta (94,25%). Aunque el Modelo 1 tiene una sensibilidad ligeramente superior (95,29%), su precisión es menor (81,94%). Dado que ambas métricas son críticas en este análisis, el Modelo 3 es la opción más equilibrada y robusta.

En resumen, el mejor modelo entre las tres funciones madre (CGAU, CMOR, MEXH) es el Modelo 3 desarrollado con la función madre MEXH. Este modelo ofrece un excelente equilibrio entre las métricas de precisión y sensibilidad, lo que lo convierte en la opción más sólida para minimizar tanto los falsos positivos como los falsos negativos.

7.1.4. RENDIMIENTO DE LOS MODELOS CNN 2D CON RESNET50

La Tabla 6 presenta los valores promedio de cada métrica para todos los modelos CNN 2D con TL desarrollados con ResNet50.

Modelos CNN 2D CON TRANSFER LEARNING (ResNet50)													
		Accuracy		Sensitivity		Specificity		Precision		F1		Auc	
		Pro (%)	Desv (%)	Pro (%)	Desv (%)	Pro (%)	Desv (%)	Pro (%)	Desv (%)	Pro (%)	Desv (%)	Pro (%)	Desv (%)
CGAU	Modelo 1	75,76	0,39	83,38	1,31	68,09	1,07	72,41	0,34	77,68	0,40	79,53	0,20
	Modelo 2	74,78	0,46	82,44	0,73	67,07	0,23	71,43	0,29	76,56	0,47	78,59	0,29
	Modelo 3	74,44	0,41	82,31	2,29	66,28	1,67	71,06	0,43	76,32	0,81	78,31	0,10
CMOR	Modelo 1	72,38	0,20	81,97	2,32	62,88	2,60	68,85	0,91	74,87	0,50	78,56	0,22
	Modelo 2	72,16	0,34	77,99	2,03	66,71	1,30	70,08	0,35	73,71	0,73	78,08	0,18
	Modelo 3	72,03	0,37	78,30	1,20	65,75	1,65	69,58	0,74	73,68	0,30	77,83	0,42
MEXH	Modelo 1	73,58	0,07	85,60	0,52	61,55	0,51	69,01	0,16	76,41	0,13	78,60	0,30
	Modelo 2	72,25	0,58	83,55	1,20	60,95	0,96	68,15	0,48	75,07	0,61	77,73	0,30
	Modelo 3	71,50	0,42	85,85	3,41	57,15	4,20	66,77	1,36	75,06	0,51	77,47	0,12

Tabla 6. Valores promedio de métricas para los modelos CNN 2D con TL (ResNet50).

El Modelo 1 se destaca como el mejor entre los modelos desarrollados con la función madre CGAU, gracias a su mayor sensibilidad (83,38%) y exactitud (75,76%). Aunque los Modelos 2 y 3 presentan resultados muy similares, su precisión y sensibilidad son ligeramente inferiores a las del Modelo 1, lo que lo convierte en la opción más sólida.

Para la función madre CMOR, el Modelo 1 es ligeramente inferior en términos de precisión (68,85%) en comparación con los Modelos 2 y 3, pero se destaca por su sensibilidad superior (81,97%). Por lo cual, el Modelo 1 es la mejor opción, ya que compensa la leve diferencia en precisión con una mejor capacidad para detectar casos positivos.

Con la función madre MEXH, el Modelo 1 destaca por su combinación de precisión de 69.01% y sensibilidad de 85.60%, además de alcanzar una exactitud del 73.58%, la más alta entre los modelos evaluados. Aunque el Modelo 3 presenta una sensibilidad similar, su precisión y exactitud son ligeramente menores, consolidando al Modelo 1 como la opción más robusta.

En general, el Modelo 1 desarrollado con la función madre MEXH se posiciona como el mejor entre todas las funciones madre evaluadas, debido a su combinación sólida de sensibilidad, precisión y exactitud, lo que lo convierte en la opción más equilibrada y eficaz.

7.2. COMPARACIÓN DE MODELOS

En la Tabla 7 se presenta un resumen del mejor modelo obtenido en cada metodología implementada para su desarrollo. Al analizar los resultados, se observa que el Modelo 2 desarrollado con CNN 1D es el mejor modelo del proyecto. Este modelo destaca por su precisión (85,69%), sensibilidad (98,81%) y exactitud (91,08%), superando a los demás en todas las métricas clave.

El segundo mejor modelo es el Modelo 3, desarrollado con CNN 2D con VGG19 utilizando la función madre MEXH, que presenta una precisión del 84,31%, una sensibilidad del 94,25% y una exactitud del 88,05%, lo que lo convierte en una opción muy sólida, aunque ligeramente inferior al Modelo 2.

En tercer lugar, se encuentra el Modelo 3 de CNN 2D sin TL con la función madre CGAU, que ofrece una precisión del 79,60%, una sensibilidad del 84,65% y una exactitud del 81,33%. Aunque es un modelo competitivo, su desempeño es más modesto en comparación con los otros dos.

Por último, el Modelo 1 desarrollado con CNN 2D con ResNet50 y la función madre MEXH ocupa la cuarta posición, con una precisión del 69,01%, una sensibilidad del 85,60% y una exactitud del 73,58%, mostrando resultados aceptables pero inferiores en general.

Identificación Modelos	Accuracy		Sensitivity		Specificity		Precision		F1		Auc	
	Pro (%)	Desv (%)	Pro (%)	Desv (%)	Pro (%)	Desv (%)	Pro (%)	Desv (%)	Pro (%)	Desv (%)	Pro (%)	Desv (%)
Modelo 2 - CNN 1D	91,08	1,77	98,81	0,70	83,30	3,32	85,69	3,06	91,76	1,71	96,61	0,86
Modelo 3 - CNN 2D MEXH (VGG19)	88,05	0,45	94,25	3,11	82,35	4,35	84,31	2,71	88,80	0,08	91,20	2,67
Modelo 3 - CNN 2D CGAU (sin preentrenamiento)	81,33	0,47	84,65	2,10	78,20	1,57	79,60	0,76	82,29	0,83	88,56	0,64
Modelo 1 - CNN 2D MEXH (ResNet50)	73,58	0,07	85,60	0,52	61,55	0,51	69,01	0,16	76,41	0,13	78,60	0,30

Tabla 7. Mejores modelos obtenidos.

7.3. INTERPRETACIÓN DE RESULTADOS

Los resultados obtenidos a lo largo del proyecto han permitido obtener un buen modelo para la tarea de apoyo en la identificación automática del STN a partir de señales MER, destacando el Modelo 2 de CNN 1D. Este modelo presentó un desempeño notable en términos de precisión (85,69%), sensibilidad (98,81%) y exactitud (91,08%), lo que refuerza su potencial como herramienta de soporte, minimizando tanto los falsos positivos como los falsos negativos.

El alto valor de sensibilidad demuestra que el modelo es eficaz para identificar correctamente los casos positivos de STN, aspecto clave para optimizar la colocación de los microelectrodos de registro durante la DBS. Esto es coherente con los objetivos iniciales del proyecto, que buscaban minimizar los errores en la identificación de esta estructura.

Por otro lado, la precisión del modelo (85,69%) asegura que la tasa de falsos positivos se mantiene en niveles controlados, lo cual es esencial en la práctica clínica. Un elevado número de falsos positivos podría llevar a la estimulación incorrecta de áreas no deseadas, disminuyendo la efectividad del tratamiento. La combinación de alta precisión y sensibilidad sugiere que el Modelo 2 logra un equilibrio sólido, posicionándolo como una gran herramienta de apoyo en el proceso de toma de decisiones para los especialistas.

7.4. LIMITACIONES Y CONSIDERACIONES

Durante el desarrollo del proyecto se tuvieron principalmente dos limitaciones, las cuales se describen a continuación:

- La primera limitación surgió al intentar leer los registros de los pacientes 1 y 2, que estaban guardados en formato .bin. Aunque se probaron diferentes métodos para leer estos archivos, no fue posible extraer los datos. Como resultado, se perdieron 100 registros asociados al STN y 100 registros de otras estructuras cerebrales para el desarrollo del proyecto.

- La segunda limitación estuvo relacionada con los requerimientos computacionales. Debido a estas restricciones, no fue posible probar más estructuras en los modelos CNN 2D, ya que el *kernel* fallaba principalmente por la falta de memoria RAM y potencia de procesamiento. Aunque el equipo utilizado contaba con una GPU, no fue suficiente para implementar estructuras más robustas. Además, el sistema de ventilación del PC no lograba mantener una temperatura adecuada durante el uso intensivo de la GPU, lo que ocasionaba que el equipo se apagara en algunas ocasiones, dejando el entrenamiento de los modelos incompleto.

8. CONCLUSIONES Y TRABAJOS FUTUROS

8.1. CONCLUSIONES

- El Modelo 2, desarrollado utilizando una CNN 1D, ha demostrado ser el más efectivo, obteniendo los mejores resultados en las métricas clave evaluadas en este proyecto. Al revisar los resultados de este modelo (precisión del 85,69%, sensibilidad del 98,81%, exactitud del 91,08% y AUC del 96,61%), es importante destacar que su rendimiento es comparable al de los modelos reportados en el estado del arte. Sin embargo, es relevante señalar que las investigaciones previas se centraron en gran medida en el procesamiento avanzado de señales, aplicando métodos matemáticos sofisticados. En contraste, el procesamiento de señales para los modelos CNN 1D fue más simple: las señales originales de 1s de duración se dividieron en fragmentos de 100 ms. A pesar de esta simplificación, la capacidad de las CNN para extraer automáticamente características ha sido clave para alcanzar un rendimiento tan notable.
- Es interesante notar que los modelos CNN 2D desarrollados a partir de los escalogramas generados con diferentes funciones madre no obtuvieron los resultados esperados. Esto podría deberse a dos factores principales: la reducción de la duración de las señales y las limitaciones computacionales.

Al reducir la duración de las señales, existe el riesgo de perder información crítica que se manifiesta a lo largo de intervalos de tiempo más largos. En los modelos del estado del arte, donde las señales tienen una duración de 1s, es probable que se capturen patrones temporales más complejos que no se perciben en fragmentos más cortos, como los de 100 ms. Esta pérdida de contexto temporal puede afectar la capacidad del modelo para identificar correctamente las características relevantes.

Por otro lado, las limitaciones computacionales restringieron el desarrollo de estructuras CNN 2D más robustas, las cuales probablemente habrían mejorado el desempeño en la identificación del STN. Al analizar los resultados, se observa que uno de los modelos más destacados fue el que aprovechó el preentrenamiento de VGG19, ocupando el segundo lugar en desempeño. Esto sugiere que, al aumentar la robustez de las arquitecturas CNN 2D, sería posible obtener mejores resultados, incluso utilizando señales de 100 ms de duración. Además, un incremento en la capacidad de las redes podría explotar más eficazmente las características presentes en las señales, lo que contribuiría a mejorar significativamente el desempeño de los modelos CNN 2D.

8.2. TRABAJOS FUTUROS

- Uno de los desafíos más notables en esta investigación fue la limitación de los recursos computacionales disponibles. Futuras investigaciones podrían enfocarse en mejorar dichos recursos o explorar alternativas como la computación en la nube, lo que permitiría entrenar modelos más complejos y robustos sin las restricciones actuales.
- Otra línea de trabajo que merece atención es el uso de señales de mayor duración. En este proyecto se utilizó una segmentación de 100 ms, pero futuros estudios podrían investigar si trabajar con señales más largas mejora la identificación de patrones complejos en las señales MER.
- Aunque los modelos preentrenados para CNN 1D son menos comunes, resultaría interesante explorar el uso de arquitecturas diseñadas específicamente para series temporales, como WaveNet, InceptionTime, ResNet 1D o incluso Autoencoders. Estas opciones ofrecen un enfoque prometedor para aprovechar características relevantes en señales unidimensionales y podrían mejorar el rendimiento en diversas aplicaciones.

9. REFERENCIAS BIBLIOGRÁFICAS

1. Michelli Federico E (2006). Enfermedad de Parkinson y trastornos relacionados. Ed. Médica Panamericana.
2. Marín, D. S., Carmona, H., Ibarra, M. y Gámez, M. (2018). Enfermedad de Parkinson: fisiopatología, diagnóstico y tratamiento. Revista de La Universidad Industrial de Santander. Salud, 50(1), 79-92.
3. Hurtado, F., Cárdenas, M. A. N., Cárdenas, F., y León, L. (2016). La enfermedad de Parkinson: Etiología, tratamientos y factores preventivos. Universitas Psychologica, 15(5).
4. Díaz Maroto, I., Fernández Díaz, E., Palazón García, E., Perona Moratalia, A. S., y García Muñozguren, S. (2012). Estimulación cerebral profunda en la enfermedad de Parkinson. Rev. neurol. (Ed. impr.), s1-s8.
5. Berganza, K., Tijero, B., González-Eizaguirre, A., Somme, J., Lezcano, E., Gabilondo, I., y Gómez-Esteban, J. C. (2016). Síntomas no motores y motores en la enfermedad de Parkinson y su relación con la calidad de vida y los distintos subgrupos clínicos. Neurología, 31(9), 585-591.
6. Álvarez, M. A., y Orozco Gutiérrez, Á. (2015). Representación óptima de señales MER aplicada a la identificación de estructuras cerebrales durante la estimulación cerebral profunda. Tecnura, 19(45), 15-27.
7. B. Y. Tapia Martillo y G. V. Gómez Toro, "La Enfermedad de Parkinson", MJ, vol. 5, n.º 14, pp. 1-11, feb. 2023.
8. H. D. V. Cardona, J. B. Padilla, R. Arango, H. Carmona, M. A. Alvarez, E. G. Estellés y A. A. Orozco, "NEURO- ZONE: On-line recognition of brain structures in stereotactic surgery-application to Parkinson's disease, en 2012 Annual International Conference of the IEEE Engineering in Medicine and Biology Society, Agosto 2012, pp. 2219-2222.

9. H. D. V. Cardona, A. A. Orozco y M. A. Alvarez, "Unsupervised learning applied in MER and ECG signals through Gaussians mixtures with the Expectation-Maximization Algorithm and Variational Bayesian Inference," en 2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), Julio 2013, pp. 4326-4329.
10. Díaz Maroto, I., Fernández Díaz, E., Palazón García, E., Perona Moratalia, A. S., Y García Muñozguren, S. (2012). Estimulación cerebral profunda en la enfermedad de Parkinson. *Rev. neurol.* (Ed. impr.), s1-s8.
11. Martínez-Fernández, C. Gasca-Salas, Á. Sánchez-Ferro, J. Á. Obeso, "Actualización en la enfermedad de Parkinson," *Revista Médica Clínica Las Condes*, vol. 27, no. 3, pp. 363-379, 2016.
12. G. A. Neri-Nani, "Síntomas motores de la enfermedad de Parkinson," *Rev Neurol Neurocir Psiquiat*, vol. 45, no. 2, pp. 45-50, 2017.
13. B. Frades-Payo, M. J. Forjaz, P. Martínez-Martín, "Situación actual del conocimiento sobre calidad de vida en la enfermedad de Parkinson: I. Instrumentos, estudios comparativos y tratamientos," *Rev Neurol*, vol. 49, no. 11, pp. 594, 2009.
14. R. Figueiras-Méndez, C. Magariños-Ascone, I. Regidor, M. Del Álamo-De Pedro, L. Cabañes-Martínez, M. Gómez-Galán, "Estimulación cerebral profunda: 12 años de experiencia y 250 pacientes intervenidos con un seguimiento de más de un año," *Rev Neurol*, vol. 49, no. 10, pp. 511-516, 2009.
15. P. Krack, A. Batir, E. Van Blercom, V. Chabardes, A. Fraix, A. Ardouin y A. L. Benabid, "Five-year follow-up of bilateral stimulation of the subthalamic nucleus in advanced Parkinson's disease," *New England Journal of Medicine*, vol. 349, no. 20, pp. 1925-1934, 2003.
16. C. CIREN, "Conceptos actuales sobre la función de los ganglios basales y el papel del núcleo subtalámico (STN) en trastornos del movimiento," *Revista Mexicana de Neurociencia*, vol. 2, no. 2, pp. 77-85, 2001.

17. A. Castro-García, A. Sesar-Ignacio, B. Ares-Pensado, J. L. Relova-Quinteiro, M. Gelabert-González, R. Martínez-Rumbo y M. Noya-García, "Complicaciones psiquiátricas y cognitivas de la estimulación subtalámica en la enfermedad de Parkinson", *Rev Neurol*, vol. 43, n.º 4, pp. 218-222, 2006.
18. I. Aguado López, "Deep Learning: Redes Neuronales Convolucionales en R", Universidad de Sevilla, Departamento de Ciencias de la Computación e Inteligencia Artificial, Grado en Matemáticas, 2020-09-01.
19. F. J. Núñez Sánchez-Agustino, S. Kanaan Izquierdo, C. Ventura Royo, "Diseño de un sistema de reconocimiento automático de matrículas de vehículos mediante una red neuronal convolucional", Máster Universitario en Ingeniería Informática, Área de Inteligencia Artificial, 01/06/2016.
20. M. Massiris, C. Delrieux, y J. Á. Fernández Muñoz, "Detección de equipos de protección personal mediante red neuronal convolucional YOLO", en XXXIX Jornadas de Automática, Área de Ingeniería de Sistemas y Automática, Universidad de Extremadura, pp. 1022-1029, 2018.
21. F. López Saca, "Clasificación de imágenes usando redes neuronales convolucionales," Maestría en Ciencias de la Computación, Unidad Azcapotzalco, Universidad Autónoma Metropolitana, México, 2019.
22. P. Loncomilla, "Deep learning: Redes convolucionales," 2016. [Online]. Available: <https://ccc.inaoep.mx/pgomez/deep/presentations>
23. I. Rodríguez Martínez, "Efecto de la sustitución de la función de pooling en redes neuronales convolucionales," 2020.
24. F. J. Núñez Sánchez-Agustino, "Diseño de un sistema de reconocimiento automático de matrículas de vehículos mediante una red neuronal convolucional," 2016.

25. A. Géron, *Aprende machine learning con scikit-learn, keras y tensorflow*, España: Anaya, 2020.
26. K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *arXiv preprint arXiv:1409.1556*, 2014.
27. K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770-778.
28. H. D. Vargas Cardona, M. A. Álvarez, and Á. Orozco Gutiérrez, Representación óptima de señales MER aplicada a la identificación de estructuras cerebrales durante la estimulación cerebral profunda,"*Revista Tecnura*, vol. 19, no. 45, pp. 15-27, 2015. doi: 10.14483/udistrital.jour.tecnura. 2015.3.a 01
29. H. D. V. Cardona, A. A. Orozco, and M. A. Alvarez, "Multi-patient learning increases accuracy for Subthala- mic nucleus identification in deep brain stimulation, in *2012 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pp. 4341-4344, August 2012.
30. H. D. V. Cardona, M. A. Alvarez, and A. A. Orozco, "Sparse representation of MER signals for localizing the Subthalamic Nucleus in Parkinson's disease surgery, in *2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, pp. 950-953, August 2014.
31. C. M. González Arbeláez, Representación óptima de señales MER mediante el método de Frames aplicado a la cirugía de la enfermedad de Parkinson,"2014.
32. R. Gómez Nieto, "Localización automática de electrodos de estimulación cerebral profunda en imágenes de tomografía computarizada: Aplicación a la enfermedad de Parkinson,"2016.
33. A. F. Londoño, .Algoritmos de clasificación lineal para la identificación de zonas cerebrales, Universidad Tecnológica de Pereira, 2016, pp. 1-32.

34. S. O. Villafañe, "Software para análisis de actividad neuronal en tiempo real durante neurocirugía, Universidad Nacional de San Martín (UNSAM), 2017, pp. 1-37.
35. R. D. Pinzon-Morales, A. A. Orozco-Gutierrez and C. G. Castellanos-Dominguez, "Feature selection using an ensemble of optimal wavelet packet and learning machine: Application to MER signals," 2010 7th International Symposium on Communication Systems, Networks y Digital Signal Processing (CSNDSP 2010), Newcastle Upon Tyne, UK, 2010, pp. 25-30, doi: 10.1109/CSNDSP16145.2010.5580465
36. <https://biblioteca.utp.edu.co/informacion-de-servicios/328/bases-de-datos/>
37. J. A. Cortés, H. B. Cano Garzón, and J. A. Chaves O., "Del análisis de Fourier a las Wavelets – Transformada Continua Wavelet (CWT)," *Scientia et Technica*, vol. XIII, no. 37, pp. 133–138, Dec. 2007.
38. V. J. Bolós and R. Benítez, "The Wavelet Scalogram in the Study of Time Series," in *Advances in Differential Equations and Applications*, Springer, Cham, 2014, pp. 133-138. DOI: https://doi.org/10.1007/978-3-319-06953-1_15.
39. "Conv1D layer," Keras, Accessed: Sep. 2024. [Online]. Available: https://keras.io/api/layers/convolution_layers/convolution1d/.
40. J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 2009, pp. 248-255, doi: 10.1109/CVPR.2009.5206848.

10. ANEXOS

Anexo A: Estructura de los Modelos 1D

En este anexo se presentan las configuraciones de los modelos CNN 1D desarrollados en el proyecto.

A.1. Modelo 1

La Tabla 8 muestra la estructura del primer modelo CNN 1D generado.

Layer (type)	Output Shape	Param #
conv1d (Conv1D)	(None, 2498, 64)	256
max_pooling1d (MaxPooling1D)	(None, 499, 64)	0
conv1d_1 (Conv1D)	(None, 497, 64)	12352
max_pooling1d_1 (MaxPooling1D)	(None, 98, 64)	0
flatten (Flatten)	(None, 6272)	0
dense (Dense)	(None, 128)	802944
dropout (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 1)	129
Total params:		815681
Trainable params:		815681
Non-trainable params:		0

Tabla 8. Estructura modelo 1 – CNN 1D.

A.2. Modelo 2

La Tabla 9 muestra la estructura del segundo modelo CNN 1D generado.

Layer (type)	Output Shape	Param #
conv1d (Conv1D)	(None, 2498, 64)	256
max_pooling1d (MaxPooling1D)	(None, 499, 64)	0
conv1d_1 (Conv1D)	(None, 497, 64)	12352
max_pooling1d_1 (MaxPooling1D)	(None, 98, 64)	0
conv1d_2 (Conv1D)	(None, 96, 64)	12352
max_pooling1d_2 (MaxPooling1D)	(None, 18, 64)	0
flatten (Flatten)	(None, 1152)	0
dense (Dense)	(None, 128)	147584
dropout (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 1)	129
Total params:		172673
Trainable params:		172673
Non-trainable params:		0

Tabla 9. Estructura modelo 2 – CNN 1D.

A.3. Modelo 3

La Tabla 10 muestra la estructura del tercer modelo CNN 1D generado.

Layer (type)	Output Shape	Param #
conv1d_4 (Conv1D)	(None, 2498, 64)	256
max_pooling1d_4 (MaxPooling1D)	(None, 499, 64)	0
conv1d_5 (Conv1D)	(None, 497, 64)	12352
max_pooling1d_5 (MaxPooling1D)	(None, 98, 64)	0
conv1d_6 (Conv1D)	(None, 96, 64)	12352
max_pooling1d_6 (MaxPooling1D)	(None, 18, 64)	0
conv1d_7 (Conv1D)	(None, 16, 64)	12352
max_pooling1d_7 (MaxPooling1D)	(None, 2, 64)	0
flatten_1 (Flatten)	(None, 128)	0
dense_2 (Dense)	(None, 128)	16512
dropout_1 (Dropout)	(None, 128)	0
dense_3 (Dense)	(None, 1)	129
Total params:		53953
Trainable params:		53953
Non-trainable params:		0

Tabla 10. Estructura modelo 3 – CNN 1D.

A.4. Modelo 4

La Tabla 11 muestra la estructura del cuarto modelo CNN 1D generado.

Layer (type)	Output Shape	Param #
conv1d (Conv1D)	(None, 2498, 64)	256
max_pooling1d (MaxPooling1D)	(None, 499, 64)	0
conv1d_1 (Conv1D)	(None, 497, 64)	12352
max_pooling1d_1 (MaxPooling1D)	(None, 98, 64)	0
conv1d_2 (Conv1D)	(None, 96, 64)	12352
max_pooling1d_2 (MaxPooling1D)	(None, 18, 64)	0
conv1d_3 (Conv1D)	(None, 16, 64)	12352
max_pooling1d_3 (MaxPooling1D)	(None, 2, 64)	0
flatten (Flatten)	(None, 128)	0
dense (Dense)	(None, 128)	16512
dropout (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 128)	16512
dropout_1 (Dropout)	(None, 128)	0
dense_2 (Dense)	(None, 1)	129
Total params:		70465
Trainable params:		70465
Non-trainable params:		0

Tabla 11. Estructura modelo 4 – CNN 1D.

A.5. Modelo 5

La Tabla 12 muestra la estructura del quinto modelo CNN 1D generado.

Layer (type)	Output Shape	Param #
conv1d (Conv1D)	(None, 2498, 64)	256
max_pooling1d (MaxPooling1D)	(None, 499, 64)	0
conv1d_1 (Conv1D)	(None, 497, 64)	12352
max_pooling1d_1 (MaxPooling1D)	(None, 98, 64)	0
conv1d_2 (Conv1D)	(None, 96, 64)	12352
max_pooling1d_2 (MaxPooling1D)	(None, 18, 64)	0
conv1d_3 (Conv1D)	(None, 16, 64)	12352
max_pooling1d_3 (MaxPooling1D)	(None, 2, 64)	0
flatten (Flatten)	(None, 128)	0
dense (Dense)	(None, 128)	16512
dropout (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 128)	16512
dropout_1 (Dropout)	(None, 128)	0
dense_2 (Dense)	(None, 128)	16512
dropout_2 (Dropout)	(None, 128)	0
dense_3 (Dense)	(None, 1)	129
Total params:		86977
Trainable params:		86977
Non-trainable params:		0

Tabla 12. Estructura modelo 5 – CNN 1D.

Anexo B: Estructura de los Modelos 2D sin TL

En este anexo se presentan las configuraciones de los modelos CNN 2D sin TL.

B.1. Modelos desarrollados con los escalogramas de la función madre CGAU

B.1.1 Modelo 1

La Tabla 13 muestra la estructura del primer modelo CNN 2D sin TL generado con la función madre CGAU.

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 248, 248, 128)	3584
max_pooling2d (MaxPooling2D)	(None, 49, 49, 128)	0
conv2d_1 (Conv2D)	(None, 47, 47, 128)	147584
max_pooling2d_1 (MaxPooling2D)	(None, 9, 9, 128)	0
conv2d_2 (Conv2D)	(None, 7, 7, 128)	147584
max_pooling2d_2 (MaxPooling2D)	(None, 1, 1, 128)	0
flatten (Flatten)	(None, 128)	0
dense (Dense)	(None, 128)	16512
dropout (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 1)	129
Total params:		315393
Trainable params:		315393
Non-trainable params:		0

Tabla 13. Modelo 1: CNN 2D con transformada CGAU.

B.1.2 Modelo 2

La Tabla 14 muestra la estructura del segundo modelo CNN 2D sin TL generado con la función madre CGAU.

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 248, 248, 64)	1792
conv2d_1 (Conv2D)	(None, 246, 246, 64)	36928
max_pooling2d (MaxPooling2D)	(None, 123, 123, 64)	0
flatten (Flatten)	(None, 968256)	0
dense (Dense)	(None, 64)	61968448
dropout (Dropout)	(None, 64)	0
dense_1 (Dense)	(None, 1)	65
Total params:		62007233
Trainable params:		62007233
Non-trainable params:		0

Tabla 14. Modelo 2: CNN 2D con transformada CGAU.

B.1.3 Modelo 3

La Tabla 15 muestra la estructura del tercer modelo CNN 2D sin TL generado con la función madre CGAU.

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 248, 248, 32)	896
max_pooling2d (MaxPooling2D)	(None, 124, 124, 32)	0
conv2d_1 (Conv2D)	(None, 122, 122, 16)	4624
max_pooling2d_1 (MaxPooling2D)	(None, 61, 61, 16)	0
conv2d_2 (Conv2D)	(None, 59, 59, 8)	1160
max_pooling2d_2 (MaxPooling2D)	(None, 29, 29, 8)	0
flatten (Flatten)	(None, 6728)	0
dense (Dense)	(None, 32)	215328
dropout (Dropout)	(None, 32)	0
dense_1 (Dense)	(None, 1)	33
Total params:		222041
Trainable params:		222041
Non-trainable params:		0

Tabla 15. Modelo 3: CNN 2D con transformada CGAU.

B.2. Modelos desarrollados con los escalogramas de la función madre CMOR

B.2.1 Modelo 1

La Tabla 16 muestra la estructura del primer modelo CNN 2D sin TL generado con la función madre CMOR.

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 248, 248, 128)	3584
max_pooling2d (MaxPooling2D)	(None, 49, 49, 128)	0
conv2d_1 (Conv2D)	(None, 47, 47, 128)	147584
max_pooling2d_1 (MaxPooling2D)	(None, 9, 9, 128)	0
conv2d_2 (Conv2D)	(None, 7, 7, 128)	147584
max_pooling2d_2 (MaxPooling2D)	(None, 1, 1, 128)	0
flatten (Flatten)	(None, 128)	0
dense (Dense)	(None, 128)	16512
dropout (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 1)	129
Total params:		315393
Trainable params:		315393
Non-trainable params:		0

Tabla 16. Modelo 1: CNN 2D con transformada CMOR.

B.2.2 Modelo 2

La Tabla 17 muestra la estructura del segundo modelo CNN 2D sin TL generado con la función madre CMOR.

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 248, 248, 64)	1792
conv2d_1 (Conv2D)	(None, 246, 246, 64)	36928
max_pooling2d (MaxPooling2D)	(None, 123, 123, 64)	0
flatten (Flatten)	(None, 968256)	0
dense (Dense)	(None, 64)	61968448
dropout (Dropout)	(None, 64)	0
dense_1 (Dense)	(None, 1)	65
Total params:		62007233
Trainable params:		62007233
Non-trainable params:		0

Tabla 17. Modelo 2: CNN 2D con transformada CMOR.

B.2.3 Modelo 3

La Tabla 18 muestra la estructura del tercer modelo CNN 2D sin TL generado con la función madre CMOR.

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 248, 248, 32)	896
max_pooling2d (MaxPooling2D)	(None, 124, 124, 32)	0
conv2d_1 (Conv2D)	(None, 122, 122, 16)	4624
max_pooling2d_1 (MaxPooling2D)	(None, 61, 61, 16)	0
conv2d_2 (Conv2D)	(None, 59, 59, 8)	1160
max_pooling2d_2 (MaxPooling2D)	(None, 29, 29, 8)	0
flatten (Flatten)	(None, 6728)	0
dense (Dense)	(None, 32)	215328
dropout (Dropout)	(None, 32)	0
dense_1 (Dense)	(None, 1)	33
Total params		222041
Trainable params		222041
Non-trainable params		0

Tabla 18. Modelo 3: CNN 2D con transformada CMOR.

B.3. Modelos desarrollados con los escalogramas de la función madre MEXH

B.3.1 Modelo 1

La Tabla 19 muestra la estructura del primer modelo CNN 2D sin TL generado con la función madre MEXH.

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 248, 248, 128)	3584
max_pooling2d (MaxPooling2D)	(None, 49, 49, 128)	0
conv2d_1 (Conv2D)	(None, 47, 47, 128)	147584
max_pooling2d_1 (MaxPooling2D)	(None, 9, 9, 128)	0
conv2d_2 (Conv2D)	(None, 7, 7, 128)	147584
max_pooling2d_2 (MaxPooling2D)	(None, 1, 1, 128)	0
flatten (Flatten)	(None, 128)	0
dense (Dense)	(None, 128)	16512
dropout (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 1)	129
Total params:		315393
Trainable params:		315393
Non-trainable params:		0

Tabla 19. Modelo 1: CNN 2D con transformada MEXH.

B.3.2 Modelo 2

La Tabla 20 muestra la estructura del segundo modelo CNN 2D sin TL generado con la función madre MEXH.

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 248, 248, 64)	1792
conv2d_1 (Conv2D)	(None, 246, 246, 64)	36928
max_pooling2d (MaxPooling2D)	(None, 123, 123, 64)	0
flatten (Flatten)	(None, 968256)	0
dense (Dense)	(None, 64)	61968448
dropout (Dropout)	(None, 64)	0
dense_1 (Dense)	(None, 1)	65
Total params:		62007233
Trainable params:		62007233
Non-trainable params:		0

Tabla 20. Modelo 2: CNN 2D con transformada MEXH.

B.3.3 Modelo 3

La Tabla 21 muestra la estructura del tercer modelo CNN 2D sin TL generado con la función madre MEXH.

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 248, 248, 32)	896
max_pooling2d (MaxPooling2D)	(None, 124, 124, 32)	0
conv2d_1 (Conv2D)	(None, 122, 122, 16)	4624
max_pooling2d_1 (MaxPooling2D)	(None, 61, 61, 16)	0
conv2d_2 (Conv2D)	(None, 59, 59, 8)	1160
max_pooling2d_2 (MaxPooling2D)	(None, 29, 29, 8)	0
flatten (Flatten)	(None, 6728)	0
dense (Dense)	(None, 32)	215328
dropout (Dropout)	(None, 32)	0
dense_1 (Dense)	(None, 1)	33
Total params:		222041
Trainable params:		222041
Non-trainable params:		0

Tabla 21. Modelo 3: CNN 2D con transformada MEXH.

Anexo C: Estructura de los modelos creados con VGG19

En este anexo se presentan las configuraciones de los modelos CNN 2D con TL, desarrollados a partir del modelo VGG19.

C.1. Modelos desarrollados con los escalogramas de la función madre CGAU

C.1.1 Modelo 1

La Tabla 22 presenta la estructura del primer modelo CNN 2D con TL, generado utilizando el modelo VGG19 y la función madre CGAU.

Layer (type)	Output Shape	Param #
vgg19 (Functional)	(None, 7, 7, 512)	20024384
flatten (Flatten)	(None, 25088)	0
dense (Dense)	(None, 128)	3211392
dropout (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 1)	129
Total params:		23235905
Trainable params:		3211521
Non-trainable params:		20024384

Tabla 22. Modelo 1: CNN 2D – VGG19 utilizando la transformada CGAU.

C.1.2 Modelo 2

La Tabla 23 presenta la estructura del segundo modelo CNN 2D con TL, generado utilizando el modelo VGG19 y la función madre CGAU.

Layer (type)	Output Shape	Param #
vgg19 (Functional)	(None, 7, 7, 512)	20024384
flatten (Flatten)	(None, 25088)	0
dense (Dense)	(None, 512)	12845568
dropout (Dropout)	(None, 512)	0
dense_1 (Dense)	(None, 1)	513
Total params:		32870465
Trainable params:		12846081
Non-trainable params:		20024384

Tabla 23. Modelo 2: CNN 2D – VGG19 utilizando la transformada CGAU.

C.1.3 Modelo 3

La Tabla 24 presenta la estructura del tercer modelo CNN 2D con TL, generado utilizando el modelo VGG19 y la función madre CGAU.

Layer (type)	Output Shape	Param #
vgg19 (Functional)	(None, 7, 7, 512)	20024384
flatten (Flatten)	(None, 25088)	0
dense (Dense)	(None, 256)	6422784
dropout (Dropout)	(None, 256)	0
dense_1 (Dense)	(None, 1)	257
Total params:		26447425
Trainable params:		6423041
Non-trainable params:		20024384

Tabla 24. Modelo 3: CNN 2D – VGG19 utilizando la transformada CGAU.

C.2. Modelos desarrollados con los escalogramas de la función madre CMOR

C.2.1 Modelo 1

La Tabla 25 presenta la estructura del primer modelo CNN 2D con TL, generado utilizando el modelo VGG19 y la función madre CMOR.

Layer (type)	Output Shape	Param #
vgg19 (Functional)	(None, 7, 7, 512)	20024384
flatten (Flatten)	(None, 25088)	0
dense (Dense)	(None, 64)	1605696
dropout (Dropout)	(None, 64)	0
dense_1 (Dense)	(None, 1)	65
Total params:		21630145
Trainable params:		1605761
Non-trainable params:		20024384

Tabla 25. Modelo 1: CNN 2D – VGG19 utilizando la transformada CMOR.

C.2.2 Modelo 2

La Tabla 26 presenta la estructura del segundo modelo CNN 2D con TL, generado utilizando el modelo VGG19 y la función madre CMOR.

Layer (type)	Output Shape	Param #
vgg19 (Functional)	(None, 7, 7, 512)	20024384
flatten (Flatten)	(None, 25088)	0
dense (Dense)	(None, 128)	3211392
dropout (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 1)	129
Total params:		23235905
Trainable params:		3211521
Non-trainable params:		20024384

Tabla 26. Modelo 2: CNN 2D – VGG19 utilizando la transformada CMOR.

C.2.3 Modelo 3

La Tabla 27 presenta la estructura del tercer modelo CNN 2D con TL, generado utilizando el modelo VGG19 y la función madre CMOR.

Layer (type)	Output Shape	Param #
vgg19 (Functional)	(None, 7, 7, 512)	20024384
flatten (Flatten)	(None, 25088)	0
dense (Dense)	(None, 256)	6422784
dropout (Dropout)	(None, 256)	0
dense_1 (Dense)	(None, 1)	257
Total params:		26447425
Trainable params:		6423041
Non-trainable params:		20024384

Tabla 27. Modelo 3: CNN 2D – VGG19 utilizando la transformada CMOR.

C.3. Modelos desarrollados con los escalogramas de la función madre MEXH

C.3.1 Modelo 1

La Tabla 28 presenta la estructura del primer modelo CNN 2D con TL, generado utilizando el modelo VGG19 y la función madre MEXH.

Layer (type)	Output Shape	Param #
vgg19 (Functional)	(None, 7, 7, 512)	20024384
flatten (Flatten)	(None, 25088)	0
dense (Dense)	(None, 64)	1605696
dropout (Dropout)	(None, 64)	0
dense_1 (Dense)	(None, 1)	65
Total params:		21630145
Trainable params:		1605761
Non-trainable params:		20024384

Tabla 28. Modelo 1: CNN 2D – VGG19 utilizando la transformada MEXH.

C.3.2 Modelo 2

La Tabla 29 presenta la estructura del segundo modelo CNN 2D con TL, generado utilizando el modelo VGG19 y la función madre MEXH.

Layer (type)	Output Shape	Param #
vgg19 (Functional)	(None, 7, 7, 512)	20024384
flatten (Flatten)	(None, 25088)	0
dense (Dense)	(None, 128)	3211392
dropout (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 1)	129
Total params:		23235905
Trainable params:		3211521
Non-trainable params:		20024384

Tabla 29. Modelo 2: CNN 2D – VGG19 utilizando la transformada MEXH.

C.3.3 Modelo 3

La Tabla 30 presenta la estructura del tercer modelo CNN 2D con TL, generado utilizando el modelo VGG19 y la función madre MEXH.

Layer (type)	Output Shape	Param #
vgg19 (Functional)	(None, 7, 7, 512)	20024384
flatten (Flatten)	(None, 25088)	0
dense (Dense)	(None, 256)	6422784
dropout (Dropout)	(None, 256)	0
dense_1 (Dense)	(None, 1)	257
Total params:		26447425
Trainable params:		6423041
Non-trainable params:		20024384

Tabla 30. Modelo 3: CNN 2D – VGG19 utilizando la transformada MEXH.

Anexo D: Estructura de los modelos creados con ResNet50

En este anexo se presentan las configuraciones de los modelos CNN 2D con TL, desarrollados a partir del modelo ResNet50.

D.1. Modelos desarrollados con los escalogramas de la función madre CGAU

D.1.1 Modelo 1

La Tabla 31 presenta la estructura del primer modelo CNN 2D con TL, generado utilizando el modelo ResNet50 y la función madre CGAU.

Layer (type)	Output Shape	Param #
resnet50 (Functional)	(None, 8, 8, 2048)	23587712
global_average_pooling2d (GlobalAveragePooling2D)	(None, 2048)	0
dense (Dense)	(None, 1)	2049
Total params:		23589761
Trainable params:		2049
Non-trainable params:		23587712

Tabla 31. Modelo 1: CNN 2D – ResNet50 utilizando la transformada CGAU.

D.1.2 Modelo 2

La Tabla 32 presenta la estructura del segundo modelo CNN 2D con TL, generado utilizando el modelo ResNet50 y la función madre CGAU.

Layer (type)	Output Shape	Param #
resnet50 (Functional)	(None, 8, 8, 2048)	23587712
global_average_pooling2d (GlobalAveragePooling2D)	(None, 2048)	0
dense (Dense)	(None, 128)	262272
dense_1 (Dense)	(None, 1)	129
Total params:		23850113
Trainable params:		262401
Non-trainable params:		23587712

Tabla 32. Modelo 2: CNN 2D – ResNet50 utilizando la transformada CGAU.

D.1.3 Modelo 3

La Tabla 33 presenta la estructura del tercer modelo CNN 2D con TL, generado utilizando el modelo ResNet50 y la función madre CGAU.

Layer (type)	Output Shape	Param #
resnet50 (Functional)	(None, 8, 8, 2048)	23587712
global_average_pooling2d (GlobalAveragePooling2D)	(None, 2048)	0
dense (Dense)	(None, 256)	524544
dense_1 (Dense)	(None, 1)	257
Total params:		24112513
Trainable params:		524801
Non-trainable params:		23587712

Tabla 33. Modelo 3: CNN 2D – ResNet50 utilizando la transformada CGAU.

D.2. Modelos desarrollados con los escalogramas de la función madre CMOR

D.2.1 Modelo 1

La Tabla 34 presenta la estructura del primer modelo CNN 2D con TL, generado utilizando el modelo ResNet50 y la función madre CMOR.

Layer (type)	Output Shape	Param #
resnet50 (Functional)	(None, 8, 8, 2048)	23587712
global_average_pooling2d (GlobalAveragePooling2D)	(None, 2048)	0
dense (Dense)	(None, 1)	2049
Total params:		23589761
Trainable params:		2049
Non-trainable params:		23587712

Tabla 34. Modelo 1: CNN 2D – ResNet50 utilizando la transformada CMOR.

D.2.2 Modelo 2

La Tabla 35 presenta la estructura del segundo modelo CNN 2D con TL, generado utilizando el modelo ResNet50 y la función madre CMOR.

Layer (type)	Output Shape	Param #
resnet50 (Functional)	(None, 8, 8, 2048)	23587712
global_average_pooling2d (GlobalAveragePooling2D)	(None, 2048)	0
dense (Dense)	(None, 128)	262272
dense_1 (Dense)	(None, 1)	129
Total params:		23850113
Trainable params:		262401
Non-trainable params:		23587712

Tabla 35. Modelo 2: CNN 2D – ResNet50 utilizando la transformada CMOR.

D.2.3 Modelo 3

La Tabla 36 presenta la estructura del tercer modelo CNN 2D con TL, generado utilizando el modelo ResNet50 y la función madre CMOR.

Layer (type)	Output Shape	Param #
resnet50 (Functional)	(None, 8, 8, 2048)	23587712
global_average_pooling2d (GlobalAveragePooling2D)	(None, 2048)	0
dense (Dense)	(None, 256)	524544
dense_1 (Dense)	(None, 1)	257
Total params:		24112513
Trainable params:		524801
Non-trainable params:		23587712

Tabla 36. Modelo 3: CNN 2D – ResNet50 utilizando la transformada CMOR.

D.3. Modelos desarrollados con los escalogramas de la función madre MEXH

D.3.1 Modelo 1

La Tabla 37 presenta la estructura del primer modelo CNN 2D con TL, generado utilizando el modelo ResNet50 y la función madre MEXH.

Layer (type)	Output Shape	Param #
resnet50 (Functional)	(None, 8, 8, 2048)	23587712
global_average_pooling2d (GlobalAveragePooling2D)	(None, 2048)	0
dense (Dense)	(None, 1)	2049
Total params:		23589761
Trainable params:		2049
Non-trainable params:		23587712

Tabla 37. Modelo 1: CNN 2D – ResNet50 utilizando la transformada MEXH.

D.3.2 Modelo 2

La Tabla 38 presenta la estructura del segundo modelo CNN 2D con TL, generado utilizando el modelo ResNet50 y la función madre MEXH.

Layer (type)	Output Shape	Param #
resnet50 (Functional)	(None, 8, 8, 2048)	23587712
global_average_pooling2d (GlobalAveragePooling2D)	(None, 2048)	0
dense (Dense)	(None, 128)	262272
dense_1 (Dense)	(None, 1)	129
Total params:		23850113
Trainable params:		262401
Non-trainable params:		23587712

Tabla 38. Modelo 2: CNN 2D – ResNet50 utilizando la transformada MEXH.

D.3.3 Modelo 3

La Tabla 39 presenta la estructura del tercer modelo CNN 2D con TL, generado utilizando el modelo ResNet50 y la función madre MEXH.

Layer (type)	Output Shape	Param #
resnet50 (Functional)	(None, 8, 8, 2048)	23587712
global_average_pooling2d (GlobalAveragePooling2D)	(None, 2048)	0
dense (Dense)	(None, 256)	524544
dense_1 (Dense)	(None, 1)	257
Total params:		24112513
Trainable params:		524801
Non-trainable params:		23587712

Tabla 39. Modelo 3: CNN 2D – ResNet50 utilizando la transformada MEXH.

Anexo E: Enlace al repositorio del proyecto

En el siguiente enlace se puede acceder al repositorio del proyecto, que contiene los códigos y resultados generados durante su desarrollo. El repositorio proporciona información detallada sobre los modelos creados, los scripts de procesamiento y los archivos utilizados para el entrenamiento y validación de los modelos.

[Enlace al proyecto.](#)

Anexo F: Resultados de los modelos 1D

Este anexo presenta los resultados obtenidos durante la evaluación de los modelos CNN 1D desarrollados a lo largo del proyecto, proporcionando una visión detallada de su desempeño.

F.1. Modelo 1

La Tabla 40 presenta las métricas obtenidas tras ejecutar cinco veces el notebook donde se definió la estructura del primer modelo CNN 1D.

Modelo 1						
# Rep. Exp.	Accuracy	Sensitivity	Specificity	Precision	F1	Auc
1	0,91	0,99	0,82	0,86	0,92	0,95
2	0,91	0,93	0,88	0,88	0,91	0,96
3	0,90	0,96	0,84	0,85	0,90	0,96
4	0,90	0,98	0,81	0,83	0,90	0,93
5	0,90	0,88	0,92	0,91	0,89	0,96
Promedio	0,90	0,95	0,85	0,87	0,90	0,95
Desv	0,00	0,04	0,04	0,03	0,01	0,01

Tabla 40. Resultados modelo 1 – CNN 1D.

F.2. Modelo 2

La Tabla 41 presenta las métricas obtenidas tras ejecutar cinco veces el notebook donde se definió la estructura del segundo modelo CNN 1D.

Modelo 2						
# Rep. Exp.	Accuracy	Sensitivity	Specificity	Precision	F1	Auc
1	0,93	0,98	0,86	0,89	0,93	0,97
2	0,89	0,99	0,79	0,82	0,89	0,95
3	0,90	0,99	0,81	0,84	0,91	0,96
4	0,92	1,00	0,83	0,86	0,92	0,97
5	0,93	0,98	0,87	0,88	0,93	0,97
Promedio	0,91	0,99	0,83	0,86	0,92	0,97
Desv	0,02	0,01	0,03	0,03	0,02	0,01

Tabla 41. Resultados modelo 2 – CNN 1D.

F.3. Modelo 3

La Tabla 42 presenta las métricas obtenidas tras ejecutar cinco veces el notebook donde se definió la estructura del tercer modelo CNN 1D.

Modelo 3						
# Rep. Exp.	Accuracy	Sensitivity	Specificity	Precision	F1	Auc
1	0,90	0,94	0,86	0,88	0,91	0,95
2	0,92	0,99	0,83	0,87	0,92	0,96
3	0,92	0,98	0,86	0,88	0,93	0,96
4	0,90	0,98	0,83	0,85	0,91	0,96
5	0,91	0,99	0,82	0,85	0,91	0,96
Promedio	0,91	0,97	0,84	0,86	0,92	0,96
Desv	0,01	0,02	0,02	0,02	0,01	0,00

Tabla 42. Resultados modelo 3 – CNN 1D.

F.4. Modelo 4

La Tabla 43 presenta las métricas obtenidas tras ejecutar cinco veces el notebook donde se definió la estructura del cuarto modelo CNN 1D.

Modelo 4						
# Rep. Exp.	Accuracy	Sensitivity	Specificity	Precision	F1	Auc
1	0,90	0,98	0,82	0,85	0,91	0,94
2	0,83	0,97	0,67	0,76	0,86	0,93
3	0,89	0,97	0,79	0,83	0,90	0,95
4	0,89	0,98	0,81	0,83	0,90	0,94
5	0,90	0,97	0,83	0,84	0,90	0,95
Promedio	0,88	0,98	0,78	0,82	0,89	0,94
Desv	0,03	0,01	0,06	0,03	0,02	0,01

Tabla 43. Resultados modelo 4 – CNN 1D.

F.5. Modelo 5

La Tabla 44 presenta las métricas obtenidas tras ejecutar cinco veces el notebook donde se definió la estructura del quinto modelo CNN 1D.

Modelo 5						
# Rep. Exp.	Accuracy	Sensitivity	Specificity	Precision	F1	Auc
1	0,81	0,94	0,69	0,75	0,83	0,85
2	0,90	0,98	0,83	0,84	0,90	0,94
3	0,78	0,86	0,67	0,76	0,81	0,79
4	0,81	0,94	0,68	0,73	0,83	0,88
5	0,89	0,98	0,80	0,82	0,90	0,93
Promedio	0,84	0,94	0,73	0,78	0,85	0,88
Desv	0,05	0,05	0,07	0,05	0,04	0,06

Tabla 44. Resultados modelo 5 – CNN 1D.

Anexo G: Resultados de los modelos 2D sin TL

En este anexo se presentan los resultados obtenidos de la evaluación de los modelos CNN 2D desarrollados sin TL.

G.1. Modelos desarrollados con los escalogramas de la función madre CGAU.

G.1.1. Modelo 1

La Tabla 45 presenta los resultados de la evaluación del primer modelo CNN 2D sin TL, desarrollado con la función madre CGAU.

Modelo 1						
# Rep. Exp.	Accuracy	Sensitivity	Specificity	Precision	F1	Auc
1	0,50	0,00	1,00	0,00	0,00	0,66
2	0,70	0,67	0,73	0,71	0,69	0,74
3	0,69	0,70	0,72	0,72	0,68	0,75
4	0,68	0,67	0,74	0,71	0,69	0,74
5	0,69	0,63	0,75	0,72	0,67	0,74
Promedio	0,65	0,53	0,79	0,57	0,54	0,73
Desv	0,08	0,30	0,12	0,32	0,30	0,04

Tabla 45. Resultados del Modelo 1: CNN 2D con transformada CGAU.

G.1.1. Modelo 2

La Tabla 46 presenta los resultados de la evaluación del segundo modelo CNN 2D sin TL, desarrollado con la función madre CGAU.

Modelo 2						
# Rep. Exp.	Accuracy	Sensitivity	Specificity	Precision	F1	Auc
1	0,66	0,69	0,63	0,65	0,67	0,70
2	0,71	0,76	0,66	0,69	0,73	0,78
3	0,66	0,70	0,61	0,65	0,67	0,69
4	0,68	0,73	0,62	0,67	0,69	0,71
5	0,72	0,76	0,68	0,70	0,73	0,77
Promedio	0,69	0,73	0,64	0,67	0,70	0,73
Desv	0,03	0,03	0,03	0,02	0,03	0,04

Tabla 46. Resultados del Modelo 2: CNN 2D con transformada CGAU.

G.1.1. Modelo 3

La Tabla 47 presenta los resultados de la evaluación del tercer modelo CNN 2D sin TL, desarrollado con la función madre CGAU.

Modelo 3						
# Rep. Exp.	Accuracy	Sensitivity	Specificity	Precision	F1	Auc
1	0,82	0,83	0,80	0,80	0,82	0,88
2	0,82	0,88	0,76	0,78	0,83	0,89
3	0,81	0,84	0,79	0,80	0,82	0,88
4	0,81	0,85	0,78	0,80	0,83	0,89
5	0,81	0,83	0,79	0,79	0,81	0,88
Promedio	0,81	0,85	0,78	0,80	0,82	0,89
Desv	0,00	0,02	0,02	0,01	0,01	0,01

Tabla 47. Resultados del Modelo 3: CNN 2D con transformada CGAU.

G.2. Modelos desarrollados con los escalogramas de la función madre CMOR.

G.2.1. Modelo 1

La Tabla 48 presenta los resultados de la evaluación del primer modelo CNN 2D sin TL, desarrollado con la función madre CMOR.

Modelo 1						
# Rep. Exp.	Accuracy	Sensitivity	Specificity	Precision	F1	Auc
1	0,81	0,83	0,78	0,79	0,81	0,87
2	0,80	0,86	0,74	0,76	0,81	0,86
3	0,80	0,84	0,75	0,77	0,81	0,86
4	0,80	0,84	0,76	0,78	0,81	0,86
5	0,80	0,84	0,76	0,78	0,81	0,87
Promedio	0,80	0,84	0,76	0,78	0,81	0,87
Desv	0,00	0,01	0,02	0,01	0,00	0,01

Tabla 48. Resultados del Modelo 1: CNN 2D con transformada CMOR.

G.2.1. Modelo 2

La Tabla 49 presenta los resultados de la evaluación del segundo modelo CNN 2D sin TL, desarrollado con la función madre CMOR.

Modelo 2						
# Rep. Exp.	Accuracy	Sensitivity	Specificity	Precision	F1	Auc
1	0,76	0,76	0,75	0,75	0,76	0,83
2	0,78	0,88	0,69	0,74	0,80	0,83
3	0,81	0,88	0,70	0,74	0,81	0,83
4	0,79	0,85	0,71	0,75	0,80	0,83
5	0,77	0,75	0,78	0,77	0,76	0,83
Promedio	0,78	0,82	0,72	0,75	0,78	0,83
Desv	0,02	0,06	0,04	0,01	0,02	0,00

Tabla 49. Resultados del Modelo 2: CNN 2D con transformada CMOR.

G.2.1. Modelo 3

La Tabla 50 presenta los resultados de la evaluación del tercer modelo CNN 2D sin TL, desarrollado con la función madre CMOR.

Modelo 3						
# Rep. Exp.	Accuracy	Sensitivity	Specificity	Precision	F1	Auc
1	0,81	0,85	0,76	0,78	0,81	0,87
2	0,80	0,85	0,75	0,77	0,81	0,86
3	0,81	0,87	0,74	0,77	0,82	0,87
4	0,81	0,86	0,76	0,78	0,81	0,87
5	0,82	0,79	0,84	0,83	0,81	0,87
Promedio	0,81	0,84	0,77	0,79	0,81	0,87
Desv	0,01	0,03	0,04	0,03	0,00	0,01

Tabla 50. Resultados del Modelo 3: CNN 2D con transformada CMOR.

G.3. Modelos desarrollados con los escalogramas de la función madre MEXH.

G.3.1. Modelo 1

La Tabla 51 presenta los resultados de la evaluación del primer modelo CNN 2D sin TL, desarrollado con la función madre MEXH.

Modelo 1						
# Rep. Exp.	Accuracy	Sensitivity	Specificity	Precision	F1	Auc
1	0,81	0,79	0,83	0,82	0,81	0,88
2	0,80	0,85	0,76	0,78	0,81	0,87
3	0,80	0,83	0,77	0,80	0,81	0,87
4	0,81	0,85	0,79	0,82	0,82	0,88
5	0,80	0,82	0,79	0,80	0,80	0,87
Promedio	0,81	0,83	0,79	0,80	0,81	0,87
Desv	0,01	0,02	0,03	0,02	0,01	0,01

Tabla 51. Resultados del Modelo 1: CNN 2D con transformada MEXH.

G.3.1. Modelo 2

La Tabla 52 presenta los resultados de la evaluación del segundo modelo CNN 2D sin TL, desarrollado con la función madre MEXH.

Modelo 2						
# Rep. Exp.	Accuracy	Sensitivity	Specificity	Precision	F1	Auc
1	0,71	0,82	0,60	0,67	0,74	0,76
2	0,72	0,70	0,74	0,73	0,72	0,78
3	0,69	0,71	0,73	0,71	0,71	0,76
4	0,69	0,71	0,73	0,72	0,72	0,77
5	0,70	0,76	0,65	0,68	0,72	0,75
Promedio	0,70	0,74	0,69	0,70	0,72	0,76
Desv	0,01	0,05	0,06	0,02	0,01	0,01

Tabla 52. Resultados del Modelo 2: CNN 2D con transformada MEXH.

G.3.1. Modelo 3

La Tabla 53 presenta los resultados de la evaluación del tercer modelo CNN 2D sin TL, desarrollado con la función madre MEXH.

Modelo 3						
# Rep. Exp.	Accuracy	Sensitivity	Specificity	Precision	F1	Auc
1	0,80	0,84	0,77	0,78	0,81	0,88
2	0,81	0,88	0,74	0,77	0,82	0,87
3	0,79	0,85	0,76	0,78	0,81	0,87
4	0,82	0,87	0,74	0,78	0,82	0,87
5	0,82	0,89	0,75	0,78	0,83	0,89
Promedio	0,81	0,87	0,75	0,78	0,82	0,88
Desv	0,01	0,02	0,01	0,01	0,01	0,01

Tabla 53. Resultados del Modelo 3: CNN 2D con transformada MEXH.

Anexo H: Resultados de los modelos creados con VGG19

En este anexo se presentan los resultados obtenidos de la evaluación de los modelos CNN 2D con TL, desarrollados a partir del modelo VGG19.

H.1. Modelos desarrollados con los escalogramas de la función madre CGAU.

H.1.1. Modelo 1

La Tabla 54 presenta los resultados de la evaluación del primer modelo CNN 2D con TL, desarrollado utilizando el modelo VGG19 con la función madre CGAU.

Modelo 1						
# Rep. Exp.	Accuracy	Sensitivity	Specificity	Precision	F1	Auc
1	0,87	0,90	0,84	0,85	0,88	0,94
2	0,86	0,85	0,88	0,87	0,86	0,90
3	0,85	0,86	0,86	0,86	0,87	0,93
4	0,87	0,87	0,87	0,88	0,88	0,94
5	0,87	0,93	0,80	0,82	0,87	0,89
Promedio	0,86	0,88	0,85	0,86	0,87	0,92
Desv	0,01	0,03	0,03	0,02	0,01	0,02

Tabla 54. Resultados del Modelo 1: CNN 2D – VGG19 con transformada CGAU.

H.1.2. Modelo 2

La Tabla 55 presenta los resultados de la evaluación del segundo modelo CNN 2D con TL, desarrollado utilizando el modelo VGG19 con la función madre CGAU.

Modelo 2						
# Rep. Exp.	Accuracy	Sensitivity	Specificity	Precision	F1	Auc
1	0,87	0,94	0,80	0,82	0,88	0,94
2	0,87	0,89	0,84	0,85	0,87	0,94
3	0,87	0,92	0,80	0,83	0,87	0,94
4	0,87	0,92	0,81	0,85	0,88	0,95
5	0,87	0,95	0,80	0,83	0,88	0,94
Promedio	0,87	0,92	0,81	0,83	0,88	0,94
Desv	0,00	0,02	0,02	0,01	0,00	0,00

Tabla 55. Resultados del Modelo 2: CNN 2D – VGG19 con transformada CGAU.

H.1.3. Modelo 3

La Tabla 56 presenta los resultados de la evaluación del tercer modelo CNN 2D con TL, desarrollado utilizando el modelo VGG19 con la función madre CGAU.

Modelo 3						
# Rep. Exp.	Accuracy	Sensitivity	Specificity	Precision	F1	Auc
1	0,87	0,90	0,84	0,85	0,87	0,90
2	0,87	0,90	0,84	0,85	0,87	0,94
3	0,86	0,89	0,83	0,84	0,87	0,91
4	0,87	0,90	0,84	0,85	0,87	0,94
5	0,87	0,95	0,79	0,82	0,88	0,94
Promedio	0,87	0,90	0,83	0,84	0,87	0,93
Desv	0,00	0,02	0,02	0,01	0,00	0,02

Tabla 56. Resultados del Modelo 3: CNN 2D – VGG19 con transformada CGAU.

H.2. Modelos desarrollados con los escalogramas de la función madre CMOR.

H.2.1. Modelo 1

La Tabla 57 presenta los resultados de la evaluación del primer modelo CNN 2D con TL, desarrollado utilizando el modelo VGG19 con la función madre CMOR.

Modelo 1						
# Rep. Exp.	Accuracy	Sensitivity	Specificity	Precision	F1	Auc
1	0,84	0,88	0,80	0,81	0,84	0,87
2	0,84	0,91	0,77	0,80	0,85	0,91
3	0,83	0,89	0,78	0,81	0,85	0,88
4	0,82	0,88	0,78	0,81	0,84	0,87
5	0,84	0,92	0,76	0,79	0,85	0,91
Promedio	0,83	0,90	0,77	0,80	0,85	0,89
Desv	0,01	0,02	0,02	0,01	0,00	0,02

Tabla 57. Resultados del Modelo 1: CNN 2D – VGG19 con transformada CMOR.

H.2.2. Modelo 2

La Tabla 58 presenta los resultados de la evaluación del segundo modelo CNN 2D con TL, desarrollado utilizando el modelo VGG19 con la función madre CMOR.

Modelo 2						
# Rep. Exp.	Accuracy	Sensitivity	Specificity	Precision	F1	Auc
1	0,84	0,90	0,78	0,81	0,85	0,92
2	0,84	0,91	0,78	0,81	0,85	0,92
3	0,84	0,90	0,78	0,81	0,85	0,92
4	0,84	0,90	0,78	0,81	0,85	0,92
5	0,85	0,88	0,81	0,82	0,85	0,92
Promedio	0,84	0,90	0,79	0,81	0,85	0,92
Desv	0,00	0,01	0,01	0,01	0,00	0,00

Tabla 58. Resultados del Modelo 2: CNN 2D – VGG19 con transformada CMOR.

H.2.3. Modelo 3

La Tabla 59 presenta los resultados de la evaluación del tercer modelo CNN 2D con TL, desarrollado utilizando el modelo VGG19 con la función madre CMOR.

Modelo 3						
# Rep. Exp.	Accuracy	Sensitivity	Specificity	Precision	F1	Auc
1	0,84	0,91	0,78	0,81	0,85	0,92
2	0,85	0,91	0,78	0,81	0,85	0,92
3	0,84	0,90	0,78	0,81	0,85	0,92
4	0,84	0,90	0,78	0,81	0,85	0,92
5	0,85	0,91	0,79	0,81	0,86	0,92
Promedio	0,84	0,90	0,78	0,81	0,85	0,92
Desv	0,00	0,00	0,00	0,00	0,00	0,00

Tabla 59. Resultados del Modelo 3: CNN 2D – VGG19 con transformada CMOR.

H.3. Modelos desarrollados con los escalogramas de la función madre MEXH.

H.3.1. Modelo 1

La Tabla 60 presenta los resultados de la evaluación del primer modelo CNN 2D con TL, desarrollado utilizando el modelo VGG19 con la función madre MEXH.

Modelo 1						
# Rep. Exp.	Accuracy	Sensitivity	Specificity	Precision	F1	Auc
1	0,88	0,96	0,79	0,82	0,88	0,89
2	0,88	0,96	0,80	0,83	0,89	0,90
3	0,88	0,96	0,80	0,82	0,89	0,89
4	0,76	0,93	0,77	0,81	0,87	0,88
5	0,88	0,96	0,79	0,82	0,89	0,90
Promedio	0,85	0,95	0,79	0,82	0,88	0,89
Desv	0,05	0,01	0,01	0,01	0,01	0,01

Tabla 60. Resultados del Modelo 1: CNN 2D – VGG19 con transformada MEXH.

H.3.2. Modelo 2

La Tabla 61 presenta los resultados de la evaluación del segundo modelo CNN 2D con TL, desarrollado utilizando el modelo VGG19 con la función madre MEXH.

Modelo 2						
# Rep. Exp.	Accuracy	Sensitivity	Specificity	Precision	F1	Auc
1	0,88	0,96	0,80	0,82	0,89	0,89
2	0,89	0,92	0,85	0,86	0,89	0,95
3	0,87	0,93	0,81	0,83	0,88	0,90
4	0,87	0,94	0,82	0,84	0,88	0,90
5	0,87	0,96	0,79	0,82	0,88	0,89
Promedio.	0,88	0,94	0,81	0,83	0,88	0,91
Desv.	0,01	0,02	0,02	0,02	0,00	0,02

Tabla 61. Resultados del Modelo 2: CNN 2D – VGG19 con transformada MEXH.

H.3.3. Modelo 3

La Tabla 62 presenta los resultados de la evaluación del tercer modelo CNN 2D con TL, desarrollado utilizando el modelo VGG19 con la función madre MEXH.

Modelo 3						
# Rep. Exp.	Accuracy	Sensitivity	Specificity	Precision	F1	Auc
1	0,89	0,92	0,86	0,86	0,89	0,94
2	0,88	0,98	0,77	0,81	0,89	0,89
3	0,88	0,92	0,85	0,86	0,89	0,94
4	0,88	0,93	0,86	0,87	0,89	0,89
5	0,88	0,97	0,78	0,82	0,89	0,90
Promedio	0,88	0,94	0,82	0,84	0,89	0,91
Desv	0,00	0,03	0,04	0,03	0,00	0,03

Tabla 62. Resultados del Modelo 3: CNN 2D – VGG19 con transformada MEXH.

Anexo I: Resultados de los modelos creados con ResNet50

En este anexo se presentan los resultados obtenidos de la evaluación de los modelos CNN 2D con TL, desarrollados a partir del modelo ResNet50.

I.1. Modelos desarrollados con los escalogramas de la función madre CGAU.

I.1.1. Modelo 1

La Tabla 63 presenta los resultados de la evaluación del primer modelo CNN 2D con TL, desarrollado utilizando el modelo ResNet50 con la función madre CGAU.

Modelo 1						
# Rep. Exp.	Accuracy	Sensitivity	Specificity	Precision	F1	Auc
1	0,76	0,85	0,67	0,72	0,78	0,80
2	0,76	0,82	0,69	0,73	0,77	0,80
3	0,76	0,84	0,67	0,72	0,78	0,79
4	0,75	0,82	0,69	0,73	0,78	0,79
5	0,76	0,84	0,68	0,72	0,78	0,80
Promedio	0,76	0,83	0,68	0,72	0,78	0,80
Desv	0,00	0,01	0,01	0,00	0,00	0,00

Tabla 63. Resultados del Modelo 1: CNN 2D – ResNet50 con transformada CGAU.

I.1.2. Modelo 2

La Tabla 64 presenta los resultados de la evaluación del segundo modelo CNN 2D con TL, desarrollado utilizando el modelo ResNet50 con la función madre CGAU.

Modelo 2						
# Rep. Exp.	Accuracy	Sensitivity	Specificity	Precision	F1	Auc
1	0,74	0,82	0,67	0,71	0,76	0,78
2	0,75	0,83	0,67	0,71	0,77	0,78
3	0,75	0,82	0,67	0,71	0,76	0,79
4	0,75	0,83	0,67	0,72	0,77	0,79
5	0,75	0,83	0,67	0,72	0,77	0,79
Promedio	0,75	0,82	0,67	0,71	0,77	0,79
Desv	0,00	0,01	0,00	0,00	0,00	0,00

Tabla 64. Resultados del Modelo 2: CNN 2D – ResNet50 con transformada CGAU.

I.1.3. Modelo 3

La Tabla 65 presenta los resultados de la evaluación del tercer modelo CNN 2D con TL, desarrollado utilizando el modelo ResNet50 con la función madre CGAU.

Modelo 3						
# Rep. Exp.	Accuracy	Sensitivity	Specificity	Precision	F1	Auc
1	0,75	0,84	0,65	0,71	0,77	0,78
2	0,75	0,83	0,66	0,71	0,77	0,78
3	0,75	0,83	0,66	0,71	0,77	0,78
4	0,75	0,83	0,66	0,71	0,77	0,78
5	0,74	0,78	0,69	0,72	0,75	0,78
Promedio	0,74	0,82	0,66	0,71	0,76	0,78
Desv	0,00	0,02	0,02	0,00	0,01	0,00
Desv	0,00	0,01	0,00	0,00	0,00	0,00

Tabla 65. Resultados del Modelo 3: CNN 2D – ResNet50 con transformada CGAU.

I.2. Modelos desarrollados con los escalogramas de la función madre CMOR.

I.2.1. Modelo 1

La Tabla 66 presenta los resultados de la evaluación del primer modelo CNN 2D con TL, desarrollado utilizando el modelo ResNet50 con la función madre CMOR.

Modelo 1						
# Rep. Exp.	Accuracy	Sensitivity	Specificity	Precision	F1	Auc
1	0,72	0,84	0,61	0,68	0,75	0,78
2	0,72	0,79	0,66	0,70	0,74	0,78
3	0,73	0,84	0,61	0,68	0,75	0,79
4	0,73	0,80	0,66	0,70	0,75	0,79
5	0,72	0,84	0,61	0,68	0,75	0,78
Promedio	0,72	0,82	0,63	0,69	0,75	0,79
Desv	0,00	0,02	0,03	0,01	0,00	0,00

Tabla 66. Resultados del Modelo 1: CNN 2D – ResNet50 con transformada CMOR.

I.2.2. Modelo 2

La Tabla 67 presenta los resultados de la evaluación del segundo modelo CNN 2D con TL, desarrollado utilizando el modelo ResNet50 con la función madre CMOR.

Modelo 2						
# Rep. Exp.	Accuracy	Sensitivity	Specificity	Precision	F1	Auc
1	0,72	0,80	0,65	0,70	0,74	0,78
2	0,72	0,75	0,69	0,70	0,72	0,78
3	0,72	0,79	0,66	0,70	0,74	0,78
4	0,72	0,79	0,66	0,70	0,74	0,78
5	0,73	0,78	0,67	0,70	0,74	0,78
Promedio	0,72	0,78	0,67	0,70	0,74	0,78
Desv	0,00	0,02	0,01	0,00	0,01	0,00

Tabla 67. Resultados del Modelo 2: CNN 2D – ResNet50 con transformada CMOR.

I.2.3. Modelo 3

La Tabla 68 presenta los resultados de la evaluación del tercer modelo CNN 2D con TL, desarrollado utilizando el modelo ResNet50 con la función madre CMOR.

Modelo 3						
# Rep. Exp.	Accuracy	Sensitivity	Specificity	Precision	F1	Auc
1	0,72	0,80	0,65	0,70	0,74	0,78
2	0,72	0,77	0,68	0,70	0,74	0,78
3	0,72	0,77	0,68	0,70	0,74	0,78
4	0,72	0,79	0,64	0,69	0,74	0,77
5	0,72	0,79	0,64	0,69	0,74	0,77
Promedio	0,72	0,78	0,66	0,70	0,74	0,78
Desv	0,00	0,01	0,02	0,01	0,00	0,00

Tabla 68. Resultados del Modelo 3: CNN 2D – ResNet50 con transformada CMOR.

I.3. Modelos desarrollados con los escalogramas de la función madre MEXH.

I.3.1. Modelo 1

La Tabla 69 presenta los resultados de la evaluación del primer modelo CNN 2D con TL, desarrollado utilizando el modelo ResNet50 con la función madre MEXH.

Modelo 1						
# Rep. Exp.	Accuracy	Sensitivity	Specificity	Precision	F1	Auc
1	0,74	0,87	0,61	0,69	0,77	0,79
2	0,74	0,85	0,62	0,69	0,76	0,79
3	0,74	0,85	0,62	0,69	0,76	0,79
4	0,74	0,86	0,62	0,69	0,76	0,78
5	0,74	0,86	0,62	0,69	0,76	0,78
Promedio	0,74	0,86	0,62	0,69	0,76	0,79
Desv	0,00	0,01	0,01	0,00	0,00	0,00

Tabla 69. Resultados del Modelo 1: CNN 2D – ResNet50 con transformada MEXH.

I.3.2. Modelo 2

La Tabla 70 presenta los resultados de la evaluación del segundo modelo CNN 2D con TL, desarrollado utilizando el modelo ResNet50 con la función madre MEXH.

Modelo 2						
# Rep. Exp.	Accuracy	Sensitivity	Specificity	Precision	F1	Auc
1	0,72	0,85	0,59	0,68	0,75	0,77
2	0,73	0,84	0,62	0,69	0,76	0,78
3	0,73	0,84	0,62	0,69	0,76	0,78
4	0,72	0,82	0,61	0,68	0,74	0,78
5	0,72	0,82	0,61	0,68	0,74	0,78
Promedio	0,72	0,84	0,61	0,68	0,75	0,78
Desv	0,01	0,01	0,01	0,00	0,01	0,00

Tabla 70. Resultados del Modelo 2: CNN 2D – ResNet50 con transformada MEXH.

I.3.3. Modelo 3

La Tabla 71 presenta los resultados de la evaluación del tercer modelo CNN 2D con TL, desarrollado utilizando el modelo ResNet50 con la función madre MEXH.

Modelo 3						
# Rep. Exp.	Accuracy	Sensitivity	Specificity	Precision	F1	Auc
1	0,72	0,81	0,64	0,69	0,74	0,78
2	0,71	0,89	0,54	0,66	0,76	0,77
3	0,71	0,89	0,54	0,66	0,76	0,77
4	0,71	0,85	0,58	0,67	0,75	0,77
5	0,71	0,85	0,58	0,67	0,75	0,77
Promedio	0,72	0,86	0,57	0,67	0,75	0,77
Desv	0,00	0,03	0,04	0,01	0,01	0,00

Tabla 71. Resultados del Modelo 3: CNN 2D – ResNet50 con transformada MEXH.