UNIVERSIDAD ESAN

FACULTAD DE INGENIERÍA

INGENIERÍA DE TECNOLOGÍAS DE INFORMACIÓN Y SISTEMAS

Propuesta de modelo de clasificación automatizada de deterioros en pavimentos de Lima Metropolitana, Perú utilizando Redes Neuronales Convolucionales

Tesis de investigación para el curso Trabajo de Tesis I

Hernan Marecos

Asesor: Marks Calderón

Lima, 28 de octubre de 2024

Esta tesis denominada:

PROPUESTA DE MODELO DE CLASIFICACIÓN AUTOMATIZADA DE DETERIOROS EN PAVIMENTOS DE LIMA METROPOLITANA, PERÚ UTILIZANDO REDES NEURONALES CONVOLUCIONALES

ha sido aprobada.

. . . . . .

Jurado 1

. . . . . .

Jurado 2

. . . . . .

Jurado 3

Universidad ESAN

2024

PROPUESTA DE MODELO DE CLASIFICACIÓN AUTOMATIZADA DE DETERIOROS EN PAVIMENTOS DE LIMA METROPOLITANA, PERÚ UTILIZANDO REDES NEURONALES CONVOLUCIONALES

Dedicatoria

A mi madre.

Sin ti nada de esto hubiera sido posible te debo todo.

Índice general

Resumen **1**

Introducción **2**

Capítulo I: Planteamiento del Problema **3**

1.1 Descripción de la Realidad Problemática 3

1.2 Formulación del Problema 3

1.2.1 Problema General 3

1.2.2 Problemas Específicos 3

1.3 Objetivos de la Investigación 4

1.3.1 Objetivo General 4

1.3.2 Objetivos Específicos 4

1.4 Hipótesis 4

1.4.1 Hipótesis General 4

1.4.2 Hipótesis Específicos 4

1.5 Justificación de la Investigación 5

1.5.1 Teórica 5

1.5.2 Práctica 5

1.5.3 Metodológica 5

1.6 Delimitación del Estudio 5

1.6.1 Espacial 5

1.6.2 Temporal 5

1.6.3 Conceptual 5

Capítulo II: Marco Teórico **6**

2.1 Antecedentes de la investigación 6

2.2 Bases Teóricas 9

2.2.1 Deep Learning 9

2.3 Marco Conceptual 10

2.3.1 Inteligencia Artficial 10

2.3.2 Perceptrón 11

2.3.3 Perceptrón Multicapa 11

2.3.4 Machine Learning 12

2.4 Hipótesis 13

2.4.1 Hipótesis General 13

2.4.2 Hipótesis Específicas 13

Capítulo III: Metodología de la Investigación **14**
3.1 Diseño de la investigación 14
3.1.1 Tipo de la investigación 14
3.1.2 Enfoque de la investigación 14
3.1.3 Población 14
3.1.4 Muestra 15
3.2 Metodología de Implementación de la Solución 15
3.3 Metodología para la Medición de Resultados de la Implementación 16
3.4 Cronograma de actividades y presupuesto 19

Capítulo IV: Desarrollo de la Solución **21**
4.1 Determinación y evaluación de alternativas de solución 21
4.2 Propuesta solución 24
4.2.1 Planeamiento y descripción de Actividades 24
4.2.2 Desarrollo de actividades 27
4.3 Medición de la solución 70
4.3.1 Análisis de Indicadores cuantitativo 70
4.3.2 Simulación de solución 72

Capítulo V: Conclusiones y Recomendaciones **75**

Referencias **78**

Anexos **80**
A Árbol de Problemas 81
B Árbol de Objetivos 82
C Matriz de Consistencia 83

Índice de Figuras

Figura 1. Perceptrón 11
Figura 2. Perceptrón Multicapa 12
Figura 3. Enfoque del Machine Learning 13
Figura 4. Metodología de implementación 15
Figura 5. Matriz de Confusión 17
Figura 6. Cronograma de actividades 19
Figura 7. Arquitectura híbrida CNN + ViT 23
Figura 8. Ejemplo de imágenes del conjunto de datos de la Universidad de Colombia 28
Figura 9. Ejemplo de archivo XML del conjunto de datos de la Universidad de Colombia 28
Figura 10. Ejemplo de imágenes del conjunto de datos de Zhujiang Hospital of Southem Medical University 29
Figura 11. Ejemplo de archivo CSV del conjunto de datos de Zhujiang Hospital of Southem Medical University 30
Figura 12. Gráfica circular de conjunto de datos de entrenamiento 31
Figura 13. Gráfica circular de conjunto de datos de prueba 32
Figura 14. Nueva distribución de ficheros para los datos 33
Figura 15. Arquitectura de Red Generadora 36
Figura 16. Arquitectura de Red Discriminadora 36
Figura 17. Parte final del entrenamietno del DCGAN 37
Figura 18. Plot de pérdidas de las red Generadora y Discriminadora 38
Figura 19. Imágenes de ultrasonido de nódulos tiroideos generados y reales 38
Figura 20. Parte final del entrenamietno del DCGAN con imágenes de nódulos malignos 40
Figura 21. Plot de pérdidas de las red Generadora y Discriminadora de imágenes de nódulos malignos 40
Figura 22. Imágenes de ultrasonido de nódulos tiroideos generados y reales (DCGAN de imágenes de nódulos malignos) 41
Figura 23. Gráfica circula de data de entrenamiento balanceada 42
Figura 24. Proceso de entrenamiento del modelo n°1 44
Figura 25. Proceso de entrenamiento del modelo n°2 45
Figura 26. Proceso de entrenamiento del modelo n°3 46
Figura 27. Proceso de entrenamiento del modelo n°4 47
Figura 28. Proceso de entrenamiento del modelo n°5 48
Figura 29. Proceso de entrenamiento del modelo n°6 49
Figura 30. Proceso de entrenamiento del modelo n°7 50

Figura 31. Proceso de entrenamiento del modelo n°8 51
Figura 32. Proceso de entrenamiento del modelo n°9 52
Figura 33. Proceso de entrenamiento del modelo n°10 53
Figura 34. Proceso de entrenamiento del modelo n°11 54
Figura 35. Proceso de entrenamiento del modelo n°12 55
Figura 36. Proceso de entrenamiento del modelo n°13 56
Figura 37. Proceso de entrenamiento del modelo n°14 57
Figura 38. Proceso de entrenamiento del modelo n°15 57
Figura 39. Proceso de entrenamiento del modelo n°16 58
Figura 40. Proceso de entrenamiento del modelo n°17 59
Figura 41. Proceso de entrenamiento del modelo n°18 60
Figura 42. Proceso de entrenamiento del modelo n°19 61
Figura 43. Proceso de entrenamiento del modelo n°20 62
Figura 44. Proceso de entrenamiento del modelo n°21 63
Figura 45. Proceso de entrenamiento del modelo n°22 64
Figura 46. Proceso de entrenamiento del modelo n°23 65
Figura 47. Proceso de entrenamiento del modelo n°24 66
Figura 48. Proceso de entrenamiento del modelo híbrido ViT + ResNet50 68
Figura 49. Proceso de entrenamiento del modelo híbrido ViT + VGG16 69
Figura 50. Proceso de entrenamiento del modelo híbrido de Google 70
Figura 51. Cargado de imágenes de ultrasonido para prueba del modelo 72
Figura 52. Prueba de modelo con imágen de ultrasonido de nódulo maligno 73
Figura 53. Prueba de modelo con imágen de ultrasonido de nódulo benigno 74

Índice de Tablas

Tabla 1. Presupuesto 20
Tabla 2. Accuracy, Recall y Precision del modelo n°1 44
Tabla 3. Accuracy, Recall y Precision del modelo n°2 45
Tabla 4. Accuracy, Recall y Precision del modelo n°3 46
Tabla 5. Accuracy, Recall y Precision del modelo n°4 47
Tabla 6. Accuracy, Recall y Precision del modelo n°5 48
Tabla 7. Accuracy, Recall y Precision del modelo n°6 49
Tabla 8. Accuracy, Recall y Precision del modelo n°7 50
Tabla 9. Accuracy, Recall y Precision del modelo n°8 51
Tabla 10. Accuracy, Recall y Precision del modelo n°9 52
Tabla 11. Accuracy, Recall y Precision del modelo n°10 53
Tabla 12. Accuracy, Recall y Precision del modelo n°11 54
Tabla 13. Accuracy, Recall y Precision del modelo n°12 55
Tabla 14. Accuracy, Recall y Precision del modelo n°13 56
Tabla 15. Accuracy, Recall y Precision del modelo n°14 57
Tabla 16. Accuracy, Recall y Precision del modelo n°15 58
Tabla 17. Accuracy, Recall y Precision del modelo n°16 58
Tabla 18. Accuracy, Recall y Precision del modelo n°17 59
Tabla 19. Accuracy, Recall y Precision del modelo n°18 60
Tabla 20. Accuracy, Recall y Precision del modelo n°19 61
Tabla 21. Accuracy, Recall y Precision del modelo n°20 62
Tabla 22. Accuracy, Recall y Precision del modelo n°21 63
Tabla 23. Accuracy, Recall y Precision del modelo n°22 64
Tabla 24. Accuracy, Recall y Precision del modelo n°23 65
Tabla 25. Accuracy, Recall y Precision del modelo n°24 66
Tabla 26. Accuracy, Recall y Precision del modelo híbrido ViT + ResNet50 68
Tabla 27. Accuracy, Recall y Precision del modelo híbrido ViT + VGG16 69
Tabla 28. Accuracy, Recall y Precision del modelo híbrido de Google 70
Tabla 29. Resultados de los modelos entrenados con los dos conjuntos de datos, técnicas de entrenamiento y pesos 71

Índice de Ecuaciones

Ecuación 1 Fórmula para calcular el accuracy 17
Ecuación 2 Fórmula para calcular el recall 18
Ecuación 3 Fórmula para calcular el precision 18

Índice de Algoritmos

1 Proceso de organización y traslado de imágenes 33
2 Preparación de datos para entrenamiento, validación y prueba 34

Resumen

Introducción

Capítulo I: Planteamiento del Problema

1.1 Descripción de la Realidad Problemática

Lima Metropolitana, con una población superior a los 10 millones de habitantes, enfrenta un grave problema de deterioro en sus pavimentos debido al tráfico vehicular intenso y las condiciones climáticas. La falta de un sistema de monitoreo eficiente ha resultado en un elevado costo de mantenimiento y en la proliferación de daños en las vías, como grietas y baches. Estos deterioros, no gestionados de manera adecuada, afectan la movilidad, la seguridad vial y la calidad de vida de los ciudadanos. A pesar de los esfuerzos del Ministerio de Transportes y Comunicaciones (MTC) para destinar fondos al mantenimiento, las intervenciones no son suficientes ni oportunas. (Organización Mundial de la Salud, 2022)

A continuación, se presentan la Figura **??** y la Figura **??** que muestran los distintos índices de incidencia y mortalidad de cáncer de tiroides en mujeres, donde se puede resaltar la presencia de Perú en los rangos más altos de cada uno de estos. El avance de la inteligencia artificial (IA) y la visión por computadora ofrecen la posibilidad de desarrollar sistemas de monitoreo automatizados que puedan detectar daños en tiempo real, optimizando el mantenimiento de las carreteras y disminuyendo los costos asociados a la rehabilitación vial.(Haugen et al., 2016).

1.2 Formulación del Problema

Con el objetivo de formular los objetivos de esta investigación, se propusieron las siguientes preguntas.

1.2.1 Problema General

PG: ¿Cómo se podría detectar y gestionar de manera automatizada y eficiente el desgaste y deterioro de los pavimentos en Lima Metropolitana para reducir los costos de mantenimiento y mejorar la seguridad vial?

1.2.2 Problemas Específicos

- PE1: ¿Cómo podría implementarse un sistema automatizado para monitorear y clasificar en tiempo real el deterioro de los pavimentos?
- PE2: ¿De qué manera podría reducirse el costo económico derivado de la rehabilitación tardía de carreteras en mal estado?
- PE3: ¿Cómo se podría minimizar el riesgo de accidentes de tránsito ocasionados por el

mal estado de las vías urbanas?

- PE4: ¿Qué estrategias permitirían priorizar la reparación de las áreas más deterioradas para optimizar la distribución de los recursos?

1.3 Objetivos de la Investigación

A continuación, se presentan el objetivo general y los objetivos específicos.

1.3.1 Objetivo General

OG: Desarrollar un modelo de clasificación automatizada utilizando redes neuronales convolucionales (CNN) para detectar y clasificar el deterioro de los pavimentos en Lima Metropolitana, optimizando el mantenimiento y mejorando la seguridad vial.

1.3.2 Objetivos Específicos

- OE1: Implementar un sistema de monitoreo y clasificación basado en IA y visión por computadora para la identificación de grietas y baches en tiempo real.
- OE2: Analizar el impacto económico de la detección temprana de daños en la infraestructura vial de Lima.
- OE3: Priorizar áreas críticas de deterioro para optimizar el uso de los recursos en las intervenciones de mantenimiento.
- OE4: Evaluar la precisión y eficiencia del modelo propuesto frente a métodos tradicionales de inspección vial.

1.4 Hipótesis

1.4.1 Hipótesis General

La implementación de un modelo automatizado de clasificación de deterioros en pavimentos, basado en redes neuronales convolucionales, mejorará la detección y el mantenimiento de las vías en Lima Metropolitana, reduciendo costos y aumentando la seguridad vial.

1.4.2 Hipótesis Específicos

- H1: El uso de redes neuronales convolucionales permitirá una clasificación más precisa del estado de los pavimentos respecto a métodos tradicionales.
- H2: La detección automatizada y en tiempo real reducirá el tiempo de respuesta para las reparaciones, minimizando los costos a largo plazo.

- H3: La priorización de las zonas más deterioradas mejorará la eficiencia en la asignación de recursos.

1.5 Justificación de la Investigación

1.5.1 Teórica

La investigación aportará al campo de la inteligencia artificial aplicada a la infraestructura urbana, ampliando el conocimiento sobre el uso de redes neuronales convolucionales para la clasificación de imágenes en un contexto de mantenimiento vial.

1.5.2 Práctica

La implementación del sistema propuesto contribuirá a mejorar la gestión del mantenimiento de carreteras en Lima Metropolitana, reduciendo los costos asociados a reparaciones y mejorando la seguridad de los ciudadanos.

1.5.3 Metodológica

El estudio aplicará metodologías de redes neuronales convolucionales y técnicas de visión por computadora, lo que permitirá validar la efectividad de modelos de IA para tareas de clasificación en escenarios reales de infraestructura vial.

1.6 Delimitación del Estudio

A continuación, se presentará la delimitación espacial, temporal y conceptual.

1.6.1 Espacial

El estudio se llevará a cabo en las principales vías urbanas de Lima Metropolitana, donde se concentra el mayor flujo vehicular y deterioro de pavimentos.

1.6.2 Temporal

La investigación se desarrollará durante el período de 2024, con recolección de datos de campo y pruebas en un entorno controlado para evaluar el rendimiento del modelo propuesto.

1.6.3 Conceptual

El estudio se centrará en la aplicación de redes neuronales convolucionales para la clasificación de imágenes de pavimentos en estado de deterioro, considerando las variables clave de desgaste, como grietas y baches.

Capítulo II: Marco Teórico

2.1 Antecedentes de la investigación

En esta parte de la investigación se presentan algunos antecedentes relacionados a la detección y pre-diagnóstico de nódulos en distintos órganos y a través de diversas metodologías. Estos ayudarán a entender el enfoque y obtener bases para un correcto desarrollo del proyecto en cuestión.

Bicbic et al. (2023) En este estudio, Jasmin Bicbic, Thomas Emmanuel Gabriel Macatangay y su equipo desarrollaron un sistema de detección automática de daños en pavimentos para Filipinas, empleando el modelo YOLOv5n6 junto con un dispositivo Jetson Nano. Este enfoque permite la detección en tiempo real de daños como grietas, fisuras de cocodrilo y baches, eliminando la necesidad de inspecciones manuales en carreteras de alto tráfico y condiciones ambientales variables.

La selección del modelo YOLOv5n6 responde a su capacidad para equilibrar precisión y velocidad en el procesamiento, mientras que Jetson Nano permite una implementación eficiente de YOLOv5n6, proporcionando un sistema ágil y adaptable. Durante las pruebas, el modelo alcanzó una precisión superior al 90 % en la clasificación de daños, mostrando una notable robustez ante variaciones en la luz y el clima. La adaptabilidad del sistema, unida a su precisión en clasificación, representa un avance considerable para países con redes viales extensas y recursos limitados.

Resultados y Rendimiento: El modelo YOLOv5n6 en Jetson Nano demostró altas tasas de precisión, superando el 90 % en pruebas controladas de campo. Su capacidad de análisis en tiempo real permite identificar y clasificar rápidamente daños en el pavimento, facilitando la priorización de mantenimiento.

Conclusiones y Futuras Aplicaciones: Este sistema contribuye significativamente a la eficiencia en el mantenimiento vial, minimizando costos y mejorando la seguridad. Los autores proponen mejoras futuras para maximizar la precisión y extender su uso a otros tipos de pavimento y áreas rurales, anticipando su utilidad para un mantenimiento preventivo de alcance nacional. Ren et al. (2024) Este trabajo, liderado por Miao Ren y su equipo, desarrolla un recurso robusto para la detección automática de daños en pavimentos llamado SVRDD (Street View Road Damage Dataset), un conjunto de datos de imágenes de calle en Beijing. SVRDD fue creado para mejorar el rendimiento de modelos de detección de objetos mediante el entrenamiento en datos reales de grietas longitudinales, transversales, fisuras de cocodrilo y baches. Utilizando imágenes georreferenciadas y anotadas de Baidu Maps, SVRDD abarca más de

20,000 instancias de daños, proporcionando una base sólida para el desarrollo de algoritmos precisos.

Al evaluar el conjunto SVRDD, se utilizaron modelos de detección de objetos como YOLOv5, YOLOX y Cascade R-CNN. YOLOv5 se destacó en términos de velocidad y precisión, alcanzando una precisión promedio (mAP@0.5) de 0.733 y un F1-Score de 0.709, lo que lo convierte en una elección óptima para la clasificación de daños en múltiples condiciones ambientales.

Resultados y Rendimiento: El rendimiento de YOLOv5 en el conjunto de datos SVRDD superó las expectativas en términos de precisión y eficiencia, destacando su potencial en la identificación de diferentes tipos de daños de pavimento en entornos variados.

Conclusiones y Futuras Aplicaciones: El SVRDD facilita avances en la visión por computadora aplicada al mantenimiento vial. La disponibilidad pública del dataset fomenta investigaciones futuras en diversas áreas urbanas y climas, ampliando la aplicabilidad de modelos de detección en imágenes de vista callejera y promoviendo la gestión eficiente de infraestructuras viales.

Ha et al. (2022) La investigación realizada por este equipo se centra en el uso de segmentación basada en aprendizaje profundo para evaluar la severidad de grietas en pavimentos, utilizando modelos de segmentación que combinan redes neuronales convolucionales (CNN) con técnicas avanzadas de detección. El enfoque está diseñado para detectar y clasificar daños de pavimento en diferentes categorías de severidad, proporcionando una evaluación detallada y precisa.

El estudio evaluó varios modelos de segmentación en diferentes condiciones de luz y resolución, y la implementación de CNN mostró ser particularmente eficaz para identificar grietas, baches y otros tipos de daños, alcanzando niveles de precisión superiores al 90 % en la segmentación de daños severos y leves. Este modelo demostró ser capaz de clasificar eficientemente grietas en tiempo real, facilitando la priorización de las reparaciones de acuerdo con la severidad.

Resultados y Rendimiento: Las pruebas controladas mostraron que el modelo CNN podía clasificar y segmentar daños con una precisión superior al 90 %, adaptándose a diversas condiciones y resoluciones de imagen, lo que facilita su implementación en sistemas de monitoreo vial.

Conclusiones y Futuras Aplicaciones: Este enfoque de segmentación profunda permite una evaluación proactiva de los daños en pavimentos, optimizando el mantenimiento al categorizar la severidad de los daños. Los autores planean integrar más datos y mejorar la precisión

para implementar esta solución en redes viales más amplias, aportando a la conservación preventiva de infraestructura vial en diversas condiciones urbanas y rurales.

Roman-Garay et al. (2023) La investigación de Mario Alberto Roman-Garay, Luis Alberto Morales-Rosales, Héctor Rodríguez-Rangel y Sofía Isabel Fernández-Gregorio se centra en un enfoque automatizado para la detección de deterioros superficiales en pavimentos, empleando redes neuronales avanzadas de tipo Transformer. La necesidad de identificar de manera temprana y precisa grietas y baches en pavimentos flexibles es crítica para la seguridad vial y la eficiencia en el mantenimiento, permitiendo acciones preventivas que pueden reducir costos y evitar deterioros graves en infraestructuras clave. El modelo utilizado, llamado Segformer, aborda el problema mediante segmentación semántica, adaptándose a condiciones ambientales diversas gracias a la robustez de esta arquitectura de red.

Para entrenar el modelo, los autores crearon un conjunto de datos inicial de 245 imágenes tomadas en diversas calles de Culiacán, México, bajo diferentes condiciones de iluminación y presencia de sombras. Con el fin de mejorar el rendimiento y la precisión del modelo, el conjunto se expandió mediante técnicas de aumento de datos, como el contraste y la rotación, alcanzando un total de 2052 imágenes. Estas imágenes fueron etiquetadas y preprocesadas, utilizando una cámara GoPro 8 Black a una altura de 1.20 metros para capturar de manera óptima la superficie de los pavimentos.

Modelo y Metodología: La arquitectura Segformer está diseñada específicamente para tareas de segmentación y detección en entornos no controlados. Utiliza un codificador que extrae características gruesas y finas de las imágenes, seguido de un decodificador que combina estas características para generar una máscara de segmentación precisa. Esto permite al modelo identificar de forma confiable daños específicos, como grietas y baches, independientemente de variaciones en las condiciones de luz o sombras. Además, se implementaron técnicas de preprocesamiento de imágenes, como el estiramiento de contraste y la conversión a escala de grises, con el objetivo de resaltar detalles críticos en las imágenes y reducir el impacto de elementos externos que pudieran interferir en el análisis.

Resultados y Rendimiento: Los resultados del modelo Segformer fueron evaluados en términos de precisión, recall y F1 Score, alcanzando métricas significativas: una precisión del 82.35 %, un F1 Score de 88.89 % y un recall de 96.55 %. Esto indica que el modelo logró detectar correctamente la mayoría de los daños superficiales en pavimentos, con un mínimo de falsos positivos, superando a otros métodos de procesamiento de imágenes y redes convolucionales en situaciones similares. El modelo mostró una alta efectividad al identificar grietas y baches incluso en condiciones desafiantes, lo que demuestra su robustez en entornos reales.

Conclusiones y Futuras Aplicaciones: Esta investigación contribuye con una metodo-

logía eficaz y aplicable para la conservación de infraestructuras viales, donde el uso de redes Transformer permite un análisis de alta precisión en diferentes tipos de pavimentos y condiciones ambientales. La implementación de este modelo en el ámbito de la gestión vial podría facilitar una monitorización continua y detallada del estado de los pavimentos, optimizando la planificación de reparaciones y asignación de recursos. Los autores sugieren ampliar el conjunto de datos y explorar mejoras en el modelo para incluir medidas de severidad en el análisis, como la longitud, anchura y profundidad de las grietas, para una evaluación más exhaustiva de los daños y su impacto en la infraestructura vial.

2.2 Bases Teóricas

2.2.1 Deep Learning

El Deep Learning o Aprendizaje Profundo es una rama del Machine Learning que conforma varios tipos de algoritmos y enfoques mucho más amplios. Este involucra tratar a los problemas que requiere mayor generalización como lo es la visión por computadora y el reconocimiento de voz. Es decir, estos problemas generales se refieren a problemas que los humanos pueden resolver con facilidad. El Deep Learning involucra al Aprendizaje Supervisado, No Supervisado y el Aprendizaje por Refuerzo. El concepto de “profundidad” se refiere a la característica de sus enfoques en usar muchas capas de redes neuronales artificiales, donde cada una de estas realiza una operación especial, para que en conjunto su complejidad y potencia de resolución de problemas sea mayor. (Hurbans, 2020)

El aprendizaje profundo (Deep Learning) ha revolucionado el campo de la inteligencia artificial y, en particular, el de la visión por computadora. Se basa en el uso de redes neuronales artificiales con múltiples capas, que permiten aprender representaciones jerárquicas de datos. Este enfoque ha demostrado ser particularmente efectivo en tareas de clasificación y detección de objetos debido a su capacidad para procesar grandes volúmenes de datos y extraer características complejas de imágenes.

El aprendizaje profundo se inspira en la estructura y funcionamiento del cerebro humano, donde las neuronas procesan información y transmiten señales. Las redes neuronales profundas (DNNs, por sus siglas en inglés) se componen de capas de neuronas, donde cada capa transforma la entrada en una representación más abstracta. Por ejemplo, en una red convolucional, las capas iniciales pueden detectar bordes y texturas, mientras que las capas más profundas pueden reconocer formas y objetos completos.

Un modelo popular en el campo de la detección de objetos es la red neuronal convolucional (CNN), que utiliza operaciones de convolución para extraer características espaciales de las imágenes. Según Goodfellow, Bengio y Courville (2016), ”las redes neuronales convo-

lucionales han sido utilizadas exitosamente para el reconocimiento de imágenes, superando a otros enfoques anteriores en términos de precisión y eficiencia”(Goodfellow et al., 2016).

En la detección de objetos, los modelos de aprendizaje profundo han permitido avances significativos. Por ejemplo, la arquitectura YOLO (You Only Look Once) es un modelo de detección en tiempo real que divide la imagen en una cuadrícula y predice las cajas delimitadoras y las clases de los objetos simultáneamente. Este enfoque ha demostrado ser rápido y preciso, lo que lo hace ideal para aplicaciones en tiempo real.

Un estudio de Redmon et al. (2016) sobre YOLO destaca que ”la capacidad de YOLO para procesar imágenes en tiempo real sin sacrificar la precisión lo convierte en una herramienta valiosa para tareas de detección de objetos en diversas aplicaciones, desde vehículos autónomos hasta monitoreo de infraestructuras”(Redmon et al., 2016).

Con el auge de arquitecturas más avanzadas, como las redes Transformer, el aprendizaje profundo ha continuado evolucionando. Modelos como DETR (DEtection TRansformer) han mostrado un enfoque novedoso al tratar la detección de objetos como un problema de secuencia a secuencia, mejorando la precisión en tareas complejas de detección y localización.

Investigaciones recientes han demostrado que las arquitecturas basadas en Transformers, combinadas con técnicas de aprendizaje profundo, ofrecen mejoras en la precisión y capacidad de generalización en comparación con las redes convolucionales tradicionales. Según Carion et al. (2020), ”DETR logra una alta precisión en la detección de objetos al abordar el problema como una tarea de atención, permitiendo una mejor captura de relaciones espaciales en las imágenes”(Carion et al., 2020).

El aprendizaje profundo ha transformado la manera en que se aborda la detección de objetos, proporcionando herramientas poderosas para la automatización y el análisis en diversos dominios. La continua evolución de las arquitecturas de redes neuronales promete aún más avances en la precisión y eficiencia de estas tecnologías, lo que es fundamental para aplicaciones críticas como el mantenimiento de infraestructuras viales y la seguridad pública.

2.3 Marco Conceptual

2.3.1 Inteligencia Artficial

Definir a la Inteligencia Artificial llega a ser complicado, esto debido a la dificultad que se tiene en definir lo que en verdad es inteligencia. Algunos grandes personajes pensaban de la inteligencia como la ambición, mientras que otros mencionaban a la imaginación como gran impulsador de la inteligencia. También lo consideraban como la capacidad de adaptarse a los cambios. Estas definiciones hacen pensar que no existe un consenso claro de lo que es

la inteligencia, menos aun de la Inteligencia Artificial; sin embargo, en términos generales, se puede calificar de "Inteligencia Artificial" a todo aquel sistema sintético que muestra algún tipo de comportamiento "inteligente". Esto quiere decir que se puede considerar como IA a todo a aquellos que simule los sentidos naturales como el visión o audición, incluso a la capacidad de algunos sistemas de aprender de forma autónoma según su entorno lo requiera. (Hurbans, 2020)

2.3.2 Perceptrón

Inventado por Frank Rosenblatt en 1957, el Perceptrón es la unidad básica de una red neuronal e incluso es considerado como la más simple red neuronal artificial (ANN). (Géron, 2022)

El Perceptrón se muestra en la Figura 1.

Figura 1. Perceptrón.

Fuente: Géron (2022). *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow*.

2.3.3 Perceptrón Multicapa

Un Multi-Layer Perceptron (MLP) consiste en la unión de varios perceptrones, con la característica de estar organizados por capas. Existe una capa de entrada y otra de salida, todas las capas, a excepción de la salida, contiene una neurona adicional llamada bias. (Géron, 2022)

Una representación gráfica de un MLP se muestra en la Figura 2.

Figura 2. Perceptrón Multicapa.

Fuente: Géron (2022). *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow*.

2.3.4 Machine Learning

El Aprendizaje Automático o Machine Learning es considerado como una ciencia y arte de dar instrucciones a una computadora para que pueda aprender por sí sola de una data. (Géron, 2022)

En la Figura 3 se presenta el enfoque tradicional del Machine Learning.

Figura 3. Enfoque del Machine Learning.

Fuente: Géron (2022). *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow*.

2.4 Hipótesis

2.4.1 Hipótesis General

HG: A través del diseño de un modelo de Deep Learning se pre-diagnóstica los nódulos tiroideos a través de imágenes de ultrasonido.

2.4.2 Hipótesis Específicas

- HE1: Las mejores características del conjunto de datos usadas para el proceso de entrenamiento y prueba permiten un mejor desarrollo del modelo de Deep Learning.
- HE2: Las técnicas de preprocesamiento a aplicar a las imágenes de ultrasonido del conjunto de datos mejora el desempeño del modelo de Deep Learning.
- HE3: Las métricas de evaluación de rendimiento de los modelos de Deep Learning logra una correcta comparación de modelos en la tarea de clasificación de imágenes de ultrasonido.
- HE4: Las arquitecturas de Deep Learning con más alto desempeño en la clasificación de imágenes de ultrasonido demuestran superioridad frente a los demás modelos.

Capítulo III: Metodología de la Investigación

3.1 Diseño de la investigación

Según Sampieri et al. (2014) el manipular y encontrar una relación de causa-efecto entre las variables independientes y dependientes son propias de los diseños experimentales.

En el caso de la presente investigación se tiene un diseño experimental porque se pretende establecer la relación entre modelos y técnicas de Deep Learning con el pre-diagnóstico de nódulos tiroideos, esto a través de imágenes de ultrasonido.

3.1.1 Tipo de la investigación

Ya definido el diseño que tendrá la investigación, se consideró al tipo experimental puro para seguir en la presente investigación, esto ya que la variable independiente (técnicas, modelos y herramientas de Deep Learning) será manipulada constantemente con el fin de encontrar una relación de causalidad ideal con la variable dependiente. Es decir, todo lo relacionado con el Deep Learning será manipulado reiteradas veces con el objetivo de encontrar el impacto en el pre-diagnóstico de nódulos tiroideos. La manipulación en las técnicas o modelos de Deep Learning también incluye a la de la data a usar, ya que se aplicarán diversas técnicas para realizar una limpieza y extracción de características que ayudarán a la posterior elaboración de un algoritmo de diagnóstico asistido.

3.1.2 Enfoque de la investigación

Según Sampieri et al. (2014) el enfoque que sigue una forma secuencial donde no se puede evitar ningún paso, y se usan herramientas estadísticas para realizar mediciones, es el enfoque cuantitativo.

La presente investigación tiene un enfoque cuantitativo, esto dado que la variable independiente usa valores numéricos y/o estadísticos para su medición. Los resultados de la variable de Deep Learning deberán ser medidos a través de valores numéricos y, en mayor medida, estadísticos.

3.1.3 Población

La población consiste en imágenes de ultrasonido de la glándula de tiroides con presencia de nódulos. Estas imágenes fueron recolectadas de 2421 pacientes por el Zhujiang Hospital of Southem Medical University y debidamente validadas por los comités de ética institucionales en China.

3.1.4 Muestra

La muestra consiste en 3493 imágenes de ultrasonido de la glándula de tiroides recopiladas de 2421 pacientes.

3.2 Metodología de Implementación de la Solución

La metodología por seguir para implementar un modelo de Deep Learning está basado en gran medida en el conocido ciclo de vida de desarrollo de modelos de Inteligencia Artificial, del cual gran parte de los antecedentes mostrados anteriormente siguen con algunas variaciones.

En el presente caso, se pretende desarrollar un modelo de Deep Learning capaz de brindar ayuda en el diagnóstico médico; por tal motivo, se optó por basar metodología de esta investigación en la de Monroy Malca (2021), modificándolo en cierta medida en el contexto de nódulos tiroideos e imágenes de ultrasonido. En la Figura 4 se presenta de forma gráfica la metodología a seguir.

Figura 4. Metodología de implementación.

Fuente: Elaboración propia.

La adquisición del conjunto de datos, es decir, imágenes de ultrasonido, consiste en la búsqueda y revisión de las bases de datos sobre imágenes de nódulos tiroideos, donde finalmente se seleccionará y descargará la de mayor utilidad. La base de datos debe cumplir con ciertos requerimientos para realizar un posterior entrenamiento del modelo que otorgue buenos

resultados. Las imágenes deben ser debidamente etiquetadas según el carácter del nódulo al que representen, es decir, se debe indicar si cada una de las imágenes pertenece a la categoría benigno o maligno. Además, para evitar discrepancias entre calidad de imágenes en futuras evaluaciones, el conjunto de datos debe poseer imágenes de distintas instituciones de salud y de distintas calidades. Esto permitirá que el modelo entrenado no dependa de la una alta o baja calidad de las imágenes para realizar una correcta predicción. Finalmente, la cantidad de datos debe ser relativamente alta con el fin de aumentar la capacidad de generalización del modelo y evitar posibles sobreajustes o bajo rendimiento.

En la etapa de preprocesamiento, se realizará en primera instancia una exploración del conjunto de datos con el fin de entender su composición y las características a mejorar; por ejemplo, un posible desbalanceo de clases o presencia de imágenes corruptas o sin etiquetar. Posteriormente, se realizará limpieza datos en caso de imágenes corruptas o de nula utilidad. Además, se usarán técnicas de Aumento de Datos en la situación de desbalanceo de datos, con el fin de evitar una baja generalización y sobreajuste en la clase mayoritaria. Además, se aplicará un redimensionamiento y normalización en las imágenes destinadas al entrenamiento del modelo con el objetivo de reducir la complejidad computacional y así obtener menor tiempo de ejecución.

Una vez se tenga la data ya preprocesada, una parte de este se usará para el entrenamiento de modelos de Deep Learning. Inicialmente, se planea usar algunos de los diversos tipos y arquitecturas de Redes Neuronales Convolucionales (CNN), específicamente los más utilizados en este tipo de tareas como lo son VGG y ResNet, pues son ideales para la extracción de características de las imágenes, facilitando así el proceso final de clasificación. Además, se desarrollarán modelos basados en arquitecturas de Vision Transformer debido al potencial que han demostrado en anteriores investigaciones. Finalmente, también se realizará el desarrollo de modelos híbridos de CNN con Vision Transformer.

Cada uno de los modelos será probado en la parte restante de la data, específicamente en la data de prueba o test. De aquí se obtendrán las predicciones de los modelos clasificando las imágenes en benigno (0) o maligno (1).

3.3 Metodología para la Medición de Resultados de la Implementación

Los resultados obtenidos de la clasificación de los modelos previamente entrenados deberán ser evaluados para una correcta elección final.

Antes de presentar las métricas a usar, es necesario conocer a la matriz de confusión y las partes que lo conforman, pues servirá como base para entenderlas.

Según Izco (2018) la matriz de confusión es una herramienta que permite ver de forma

más clara el rendimiento de nuestro modelo. Este se presenta en la Figura 5.

	Actual Positive	Actual Negative
Predicted Positive	TP	FP
Predicted Negative	FN	TN

Figura 5. Matriz de Confusión.

Fuente: Elaboración propia.

La matriz consta de cuatro partes importantes: TP, FP, FN y TN. Estos serán usados para presentar las fórmulas de las métricas más adelante. El primero (TP o true positive) se refiere a la cantidad de observaciones que se han predicho como positivos y que en verdad sí son positivos; por el contrario, FP (false positive) se refiere a aquellas predicciones dadas como positivos, pero en verdad son negativos. FN (false negative) es la cantidad de observaciones predichas como negativas; sin embargo, estas en realidad son positivas. Finalmente, TN (true negative) es la cantidad de observaciones predichas como negativas y que en realidad son también negativas.

A continuación, se presenta las métricas para medir el desempeño de la clasificación del modelo.

El accuracy representa aquella proporción del total de predicciones que se ha obtenido correctamente (Izco, 2018). Este se calcula a través de la siguiente fórmula 1.

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

El recall representa la proporción de solo los positivos reales predichos de manera acertada (Izco, 2018). Se calcula con la siguiente fórmula.

$$recall = \frac{TP}{TP + FN} \tag{2}$$

Precision representa aquella proporción de lo predicho positivamente que es positiva (Izco, 2018). Se calcula con la fórmula a continuación.

$$precision = \frac{TP}{TP + FP} \tag{3}$$

3.4 Cronograma de actividades y presupuesto

Se propuso un cronograma para la investigación. Conforma desde el inicio hasta ser terminada con la sustentación final planeada para mediados del año 2024. Este se presneta en la Figura 6.

Figura 6. Cronograma de actividades.

Fuente: Elaboración propia.

Además, se determinó el presupuesto necesario para la elaboración completa de la investigación. Este se presenta en la Tabla 1.

Tabla 1
Presupuesto.

Grupo	Item	Costo (soles)	Subtotal
Recursos materiales	Laptop	S/ 6,500.00	
	Materiales de escritorio	S/ 100.00	S/ 6,600.00
Software y trámites	Matrícula de Trabajo de Tesis II	S/ 375.00	
	Cuotas de Trabajo de Tesis II	S/ 1,044.00	
	Software	S/ 50.00	
	Renta de servidor en la nube	S/ 224.15	S/ 1,693.15
Extras	Consultorías	S/ 100.00	
	Movilidad	S/ 200.00	S/ 300.00
	Total		S/ 8,593.15

Fuente: Elaboración propia.

Capítulo IV: Desarrollo de la Solución

En el presnete capítulo se describe las herramientas y procesos de desarrollo del modelo de Deep Learning.

4.1 Determinación y evaluación de alternativas de solución

En esta sección se presentarán las arquitecturas base CNN y ViT que se usarán en el entrenamiento de los modelos de Deep Learning para la tarea de clasificación de nódulos tiroideos a través de imágenes de ultrasonido. Además, también se introducirá la idea de modelo híbrido que junta las capacidades de extracción de características de los CNN con la actualmente popular arquitectura ViT.

En el grupo de modelos CNN, se tiene en primera instancia a la arquitectura VGG16, ya presentada anteriormente en el Capítulo 2.Este modelo posee 16 capas de profundidad junto con un tamaño de 3 x 3 en los filtros. Se ha reconocido que VGG16, y las arquitecturas VGGNet en general, poseen una alta capacidad de generalización, lo cuál lo vuelve ideal para tareas de clasificación distintas a los que originalmente fue entrenado. (Simonyan & Zisserman, 2015)

En segundo lugar, también parte del grupo de modelos CNN, se tiene a ResNet 50, también ya presentado inicialmente en el Capítulo 2. Esta arquitectura introdujo el concepto de redes residuales que permitieron obtener de manera más optimizada un mayor desempeño en arquitecturas de alta profundidad. (He et al., 2016) La versión de ResNet usada en esta investigación fue la de 50 capas de profundidad.

En el conjunto de modelos ViT, el usado en la presente investigación fue la ViT-Base con patch de tamaño 16 x 16.

Los modelos ViT también ya fueron presentados en el Capítulo 2.

Esta arquitectura, basada en los originales Transformers, ha demostrado tener mejor desempeño en tareas de clasificación comparado a los modelos basados en CNNs, aunque ciertamente con una mayor necesidad de datos de entrenamiento.

En la Figura **??**, ya se muestra la forma de la arquitectura del modelo ViT. Cabe volver a resaltar que la versión del modelo ViT utilizado es el Base que consta de 86 millones de parámetros y recibe patch de tamaño 16 x 16.

Finalmente, el último modelo presente en la investigación es uno que combina la capacidad de los CNN y los ViT para obtener un modelo híbrido.

Este modelo utiliza al CNN como backbone que alimentará los features extraídos en

forma de patch de tamaño 1 x 1 al modelo ViT. Para lograr esto se es necesario quitar el cabezal clasificador de la arquitectura CNN, pues esta tarea ya será desarrollada por el propio ViT.

Debido a las limitaciones de tiempo y recursos, solamente se desarrollará dos modelos híbridos con aquellos CNN de mejor desempeño en los entrenamientos iniciales.

Cabe destacar que este modelo híbrido fue inspirado por el trabajo presentado por JERBI et al. (2023). A continuación se presenta la Figura 7, donde se muestra la arquitectura híbrida.

Figura 7. Arquitectura híbrida CNN + ViT.

Fuente: JERBI et al. (2023). *Automatic classification of ultrasound thyroids images using vision transformers and generative adversarial networks*.

4.2 Propuesta solución

4.2.1 Planeamiento y descripción de Actividades

La metodología descrita en el capítulo anterior consta de tres etapas principales: adquisición de datos, preprocesamiento de los datos y la aplicación de Deep Learning; es decir, el entrenamiento de los diferentes tipos de algoritmos. En la siguiente parte de esta investigación se procederá describir las actividades necesarias para desarrollar y completar cada una de estas etapas para, finalmente, obtener los resultados de cada uno de los modelos entrenados.

La metodología inicia con la adquisición del conjunto de datos necesario para realizar el entrenamiento de los modelos de Deep Learning. Este consta de las siguientes actividades.

Actividad 1: Buscar y revisar bases de datos sobre imágenes de ultrasonido de nódulos tiroideos
En esta actividad se investigarán y analizarán aquellos conjunto de datos que cumplan con las características básicas requeridas: poseer imágenes de ultrasonido de nódulos tiroideos benignos y malignos.

Entregable: Diversos conjuntos de datos de imágenes de ultrasonido de nódulos tiroideos benignos y malignos.

Actividad 2: Filtrado de los conjuntos de datos
El conjunto de datos seleccionado debe cumplir con los siguientes requerimientos: las imágenes deben poseer información del carácter del nódulo; en otras palabras, se debe tener información sobre la imagen cada imagen de ultrasonido donde se indique si se está ante un nódulo de carácter benigno o maligno. Otra importante característica, necesaria para evitar obtener completamente distintos resultados en futuras investigaciones, es la fuente de origen de las imágenes del conjunto de datos, ya que este debe poseer imágenes de diferentes instituciones y herramientas. Esta característica permitirá a los modelos a entrenar tener la capacidad de afrontar distintas calidades y tipos de imágenes de ultrasonido. La última característica radica en la cantidad de las imágenes, ya que el conjunto de datos debe poseer una moderada cantidad de datos que permitan a los modelos interpretar mejor las imágenes de ultrasonido, y así finalmente aumentar la capacidad de generalización del modelo.

Entregable: Conjunto de datos de imágenes de ultrasonido de nódulos tiroideos benignos y malignos que cumple con todas las características requeridas.
En la etapa de preprocesamiento, necesaria para realizar un correcto entrenamiento de los modelos de Deep Learning, se realizará un análisis y limpieza de los datos, además, se pasará por

un proceso de preparamiento de los datos para facilitar el flujo de los datos a través de los de los modelos a entrenar. Esta etapa consta de dos flujos, donde uno de estos incluye el Aumento de Datos con el modelo DCGAN; mientras que el otro, evita el paso por este proceso debido a la fiabilidad de este tipo de red para generar datos que puedan ser útiles para el entrenamiento de los modelos. Ambos flujos constan de las tres primeras actividades descritas a continuación.

Actividad 1: Explorar características del conjunto de datos

Se inicia con la exploración del conjunto de datos seleccionado, extrayendo gráficas y tablas que muestren su composición y características, permitiendo así determinar qué técnicas se podrían aplicar para mejorar las flaquezas del conjunto de datos. Aquí se puede obtener información sobre la necesidad o no de aplicar el Aumento de Datos a una clase minoritaria si lo hubiera.

Entregable: Gráficas y tablas de características y composición del conjunto de datos.

Actividad 2: Filtrar y organizar imágenes poco útiles del conjunto de datos

Se realizará un filtrado de aquellas imágenes corruptas, atípicas o sin etiqueta, dejando solo en el conjunto de datos a aquellas de mayor utilidad, al mismo tiempo que se reorganiza en un nuevo formato de ficheros para facilitar su posterior carga a los modelos de Deep Learning.

Entregable: Conjunto de datos organizados y sin imágenes corruptas, atípicas o sin etiquetas.

Actividad 3: Redimensionar, normalizar y dividir los datos según los requerimientos para cada modelo de Deep Learning

Inicialmente se aplicará a cada imagen un redimensionamiento y normalización según se vea conveniente, esto con el fin de evitar mayor complejidad computacional y reducir los tiempos de ejecución de los modelos. Luego se hará una división del conjunto de datos total en data de entrenamiento, validación y prueba.

Entregable: Conjunto de datos de entrenamiento, validación y prueba preprocesados.

El segundo flujo de la metodología, al evitar el proceso de Aumento de Datos, obvia la siguiente actividad y continúa directamente en la siguiente etapa.

Actividad 4: Mejorar composición del conjunto de datos

El Aumento de Datos se debe utilizar en caso de desbalance de clases en el conjunto de datos con el objetivo de aumentar la capacidad de generalización del modelo y evitar su sobreajuste.

Debido a las características que determinan si una imagen de ultrasonido es benigno o

maligno (forma, orientación, tamaño y nivel de brillo), usar las técnicas comunes de Aumento de Datos de hacer diferentes transformaciones en las imágenes originales de forma aleatorio, no tendría sentido. Por este motivo, en esta actividad, se entrenará un modelo DCGAN personalizado que logre aprender la distribución de las imágenes reales y pueda generar posteriormente imágenes sintéticas. El entrenamiento de este modelo se realizará a través del conjunto de datos de entrenamiento obtenido de la anterior actividad, seleccionando aquellos pertenecientes a la clase de minoritaria.

Ya con el modelo entrenado, este será utilizado para generar y aumentar la cantidad de datos final para la clase minoritaria en la data de entrenamiento.

Entregable: Conjunto de datos con ambas clases balanceadas.
Una vez se termina el preprocesamiento y se obtiene la data final, se continúa con el entrenamiento y validación de modelos de Deep Learning, seguido de una prueba final con la data respectiva. Esta etapa se divide en las siguientes actividades.

Actividad 1: Seleccionar algoritmos a entrenar según usos y resultados de las investigaciones presentadas en los antecedentes
Se revisará aquellas arquitecturas utilizadas en las investigación mencionadas en el Capítulo 2. De todas las recolectadas, se seleccionarán las de mejor desempeño y de mayor presencia.

Entregable: Arquitecturas de Deep Learning ideales para la tarea de clasificación de imágenes de ultrasonido de nódulos tiroideos.

Actividad 2: Entrenar y probar los modelos de Deep Learning previamente definidos
Se realizará el entrenamiento de cada uno de las diferentes arquitecturas con configuraciones establecidas por anteriores investigaciones. También se aplicarán dos técnicas por separado para este proceso: Transfer Learning y Fine Tuning. Además, en cada época de entrenamiento de todos los modelos, se usará el conjunto de datos de validación para verificar el avance de la capacidad predictiva de los modelos.

Finalmente, cada uno de estos modelo será puesto a prueba en el conjunto de datos restantes (data de prueba), así finalmente se obtendrán las métricas correspondientes que permitirán luego realizar el análisis de desempeño.

Entregable: Modelos CNN y ViT entrenados. Métricas de cada uno de los modelos.

Actividad 3: Definir modelos híbridos entre ViT y CNN

Se desarrollarán modelos híbridos entre CNN y ViT. En cada uno de estos, la arquitectura CNN funcionará como backbone o modelo base que recibirá las imágenes y posteriormente alimentará al modelo ViT.

Debido a la limitación de tiempo y poder computacional al que se puede acceder, solamente se seleccionará dos modelo CNN de mejor desempeño como backbone para elaborar dos híbridos. Adicionalmente, también se utilizará un modelo híbrido totalmente preentrenado.

Entregable: Modelos híbridos CNN + ViT.

Actividad 4: Entrenar y probar modelos híbridos CNN + ViT
Al igual que los anteriores modelos, estos nuevos modelos híbridos serán entrenados con la misma data de entrenamiento y con las mismas configuraciones, a excepción de algunas variaciones.

En este caso solo se aplicará la técnica de Fine Tuning debido a la capacidad de procesamiento disponible.

En cada época del entrenamiento, se usará la data de validación para verificar su desempeño en este proceso.

Finalmente, al igual que los otros modelos, los híbridos también serán sometidos al conjunto de datos de prueba para obtener sus métricas correspondientes.

Entregable: Modelos híbridos CNN + ViT entrenandos. Métricas de los modelos híbridos.

4.2.2 Desarrollo de actividades

En esta sección de la investigación se procederá a desarrollar cada una de las actividades aplicando las técnicas y herramientas ya mencionadas en anteriores capítulos siguiendo la metodología de implementación previamente definida.

La primera etapa de Adquisición del conjunto de datos se desarrollaron las ya definidas actividades de la siguiente forma.

Actividad 1: Buscar y revisar bases de datos sobre imágenes de ultrasonido de nódulos tiroideos
En los antecedentes presentados en la sección 2.1, la mayoría de investigaciones relacionadas a imágenes de ultrasonido de nódulos tiroideos, no posee datos de acceso libre, esto conlleva a que se tenga una limitada cantidad de opciones de donde escoger y filtrar el conjunto de datos

a usar para entrenar y probar los modelos de Deep Learning.

La base de datos más popular y de acceso libre de imágenes de ultrasonido de nódulos tiroides es el dado por la Universidad de Colombia, CIM@LAB y IDIME (Instituto de Diagnóstico Médico). Este conjunto de datos consta de 472 imágenes, de las cuales, la amplia mayoría (397) son de nódulos malignos, mientras que los restantes pertenecen a aquellos de carácter benigno. Este conjunto de datos posee, además, anotaciones de expertos en archivos de formato XML, donde indica diversas características de las imágenes, además de datos personales de los pacientes. En la Figura 8 se muestra algunas imágenes de este conjunto de datos, mientras que en la Figura 9 se muestra un ejemplo de archivo XML también presente.

Figura 8. Ejemplo de imágenes del conjunto de datos de la Universidad de Colombia.

Fuente: Elaboración propia.


```
▼<case>
   <number>10</number>
   <age>74</age>
   <sex>F</sex>
   <composition>solid</composition>
   <echogenicity>hyperechogenicity</echogenicity>
   <margins>spiculated</margins>
   <calcifications>microcalcifications</calcifications>
   <tirads>4b</tirads>
   <reportbacaf/>
   <reporteco/>
  ▼<mark>
     <image>1</image>
     <svg>[{"points": [{"x": 326, "y": 172}, {"x": 332, "y": 172}, {"x": 341,
     "y": 167}, {"x": 364, "y": 165}, {"x": 364, "y": 160}, {"x": 371, "y": 15
     131}, {"x": 392, "y": 126}, {"x": 389, "y": 120}, {"x": 386, "y": 115}, {
     370, "y": 91}, {"x": 364, "y": 89}, {"x": 354, "y": 90}, {"x": 350, "y":
     {"x": 313, "y": 87}, {"x": 308, "y": 91}, {"x": 303, "y": 95}, {"x": 300,
     "y": 121}, {"x": 277, "y": 128}, {"x": 280, "y": 133}, {"x": 283, "y": 13
     162}, {"x": 296, "y": 162}, {"x": 300, "y": 162}, {"x": 306, "y": 164}, {
     "annotation": {}, "regionType": "freehand"}]</svg>
   </mark>
 </case>
```


Figura 9. Ejemplo de archivo XML del conjunto de datos de la Universidad de Colombia.

Fuente: Elaboración propia.

Otro conjunto de datos, también de acceso libre, aunque no tan difundido como el presentado anteriormente es el otorgado por Zhujiang Hospital of Southem Medical University que recopila imágenes de 2421 pacientes obteniendo un total de 3493 imágenes de ultrasonido, de los cuales, 2283 pertenecen a imágenes de glándulas de tiroides con nódulos de carácter benigno, mientras que 1210 pertenecen a nódulos malignos. Además, este conjunto de datos también posee archivos CSV donde especifica el nombre de la imagen y la etiqueta a la que pertenece: benigno (0) o maligno (1).

Cabe destacar que este conjunto de datos posee características particulares que lo diferencian del primero, y es que este, además de estar validado por distintos comités de ética en China, procede de distintas fuentes; es decir, las imágenes pueden variar de tamaño y/o calidad entre sí.

En la Figura 10 se muestran algunas imágenes de este conjunto de datos, mientras que en la Figura 11 se muestra una sección de los archivos CSV.

Figura 10. Ejemplo de imágenes del conjunto de datos de Zhujiang Hospital of Southem Medical University.

Fuente: Elaboración propia.

	A	B
1	0130.jpg	0
2	1882.jpg	0
3	2207.jpg	1
4	0690.jpg	0
5	1898.jpg	0
6	0100.jpg	0
7	1013.jpg	0
8	0695.jpg	0
9	1179.jpg	0
10	2305.jpg	0

Figura 11. Ejemplo de archivo CSV del conjunto de datos de Zhujiang Hospital of Southem Medical University.

Fuente: Elaboración propia.

Actividad 2: Filtrado de los conjuntos de datos

Los requerimientos para un buen conjunto de datos radica principalmente en la información del tipo o carácter del nódulo, el origen de las imágenes de ultrasonido y la cantidad de datos.

El conjunto de datos otorgado por la Universidad de Colombia, CIM@LAB y IDIME poseía buena información, superando el requerimiento base de tipo o carácter del nódulo, pues otorgaba datos del paciente que podrían ser útiles a la hora de realizar el proceso de pre-diagnóstico; sin embargo, la gran debilidad del este conjunto de datos es la poca cantidad de imágenes que posee. Además, el hecho de no definirse el origen específico de estas imágenes en relación a si se obtuvo de distintas instituciones con diferentes herramientas o no, conllevó finalmente a su descarte.

El conjunto de datos de Zhujiang Hospital of Southem Medical University cumple exactamente con otorgar información sobre el carácter de los nódulos de cada imagen que posee, además, también se especifica el origen variado de las imágenes, pues fueron recopiladas de distintas instituciones y con diferentes herramientas. Finalmente, la cantidad de datos, aunque con clases desbalanceadas, supera en gran medida al primer conjunto de datos.

Al cumplir con todos los requerimientos para el entrenamiento de un buen modelo Deep Learning, el conjunto de datos finalmente seleccionado fue el de Zhujiang Hospital of Southem Medical University.

Como se mencionó en la sección anterior, la etapa siguiente es la de preprocesamiento. En este se realizó un completo análisis, limpieza y preparación de los datos para el entrena-

miento de los modelos.

Cabe recordar además la existencia de dos flujos en este proceso igualmente descritos en la sección anterior. Ambos inician de igual forma con las tres siguientes actividades.

Actividad 1: Explorar características del conjunto de datos
Ya con el conjunto de datos seleccionado, se utilizó la data almacenada en los archivos CSV (mostrado en la Figura 11) para contabilizar la cantidad de imágenes por cada clase y así determinar si existía o no la presencia de desbalance. La Figura 12 y Figura 13 muestran gráficas circulares donde se aprecia la proporción del total de datos por cada clase y por cada tipo de datos (entrenamiento y prueba).

Figura 12. Gráfica circular de conjunto de datos de entrenamiento.

Fuente: Elaboración propia.

Figura 13. Gráfica circular de conjunto de datos de prueba.

Fuente: Elaboración propia.

En ambas figuras se puede verificar la presencia de desbalance en clases, observando además como la clase de nódulos benignos duplica aproximadamente a la cantidad de imágenes de la clase de nódulos malignos. Este comportamiento se ve reflejado tanto para los datos de entrenamiento como para los de prueba.

Este descubrimiento afirmó la necesidad de aplicar una técnica de balanceo de datos.

Actividad 2: Filtrar y organizar imágenes poco útiles del conjunto de datos

A través del Algoritmo 1 se logró filtrar y obviar aquellas imágenes corruptas (en el caso de este conjunto de datos no se tuvo ninguno). Además, este también permitió la lectura de solo las imágenes con nombre presente en los archivos CSV, lo cual logró evitar el cargado de imágenes sin etiqueta alguna (en el caso de este conjunto de datos tampoco se tuvieron imágenes sin etiquetas). Finalmente, todas aquellos datos que cumplieran con los anteriores pasos serían trasladados a la nueva distribución de ficheros mostrada en la Figura 14.

Algoritmo 1 Proceso de organización y traslado de imágenes

1: **Entradas:** pd_train, pd_test, $source_dir_train$, $source_dir_test$, $train_dir$, $test_dir$
2: Obtener nombres de imágenes de entrenamiento de pd_train
3: Crear diccionario d de nombres a etiquetas de pd_train
4: **for** cada nombre de imagen im en nombres de imágenes de entrenamiento **do**
5: **Intentar:**
6: **if** etiqueta de im en d es 1 **then**
7: Copiar im a $train_dir/malignant$
8: **else if** etiqueta de im en d es 0 **then**
9: Copiar im a $train_dir/benign$
10: **end if**
11: **Excepción:**
12: **Imprimir:** La imagen im no se ha podido cargar...
13: **end for**
14: Obtener nombres de imágenes de prueba de pd_test
15: Crear diccionario dt de nombres a etiquetas de pd_test
16: **for** cada nombre de imagen im en nombres de imágenes de prueba **do**
17: **Intentar:**
18: **if** etiqueta de im en dt es 1 **then**
19: Copiar im a $test_dir/malignant$
20: **else if** etiqueta de im en dt es 0 **then**
21: Copiar im a $test_dir/benign$
22: **end if**
23: **Excepción:**
24: **Imprimir:** La imagen im no se ha podido cargar...
25: **end for**
26: **Imprimir:** SE TRASLADARON TODAS LAS IMÁGENES AL NUEVO DIRECTORIO!

```
./data/finalData/
./data/finalData/test
./data/finalData/test\benign
./data/finalData/test\malignant
./data/finalData/train
./data/finalData/train\benign
./data/finalData/train\malignant
```

Figura 14. Nueva distribución de ficheros para los datos.

Fuente: Elaboración propia.

Actividad 3: Redimensionar, normalizar y dividir los datos según los requerimientos para cada modelo de Deep Learning

A partir de esta actividad, los datos usados fueron aquellos anteriormente trasladados desde los ficheros originales a los nuevos directorios.

El tamaño a redimensionar de las imágenes del conjunto de datos dependió únicamente del modelo en el que este se usó.

La normalización de los valores de píxeles de cada imagen se hizo con un media y desviación estándar de 0.5 aplicadas a cada uno de los 3 canales de las imágenes. Esto llevó a obtener finalmente píxeles con valores dentro del rango de [-1, 1], lo cual se recomienda para realizar entrenamientos de los modelos de forma más eficiente.

Finalmente, el conjunto de datos de entrenamiento se dividió en dos partes: data de validación, el cual representa el 15 % del total, y data de entrenamiento, compuesta del 85 % restante. La distribución de los datos se hizo de forma aleatoria manteniendo los mismos porcentajes.

La cantidad de la data de prueba se mantuvo igual a como originalmente fue obtenida.

Todo este proceso se realizó a través del Algoritmo 2, donde finalmente se otorgaron los tres conjuntos de datos en tipo DataLoader de Pytorch que posteriormente facilitó la ingesta de datos a los modelos.

Algoritmo 2 Preparación de datos para entrenamiento, validación y prueba

1: **Entradas:** $train_dir$, $test_dir$, img_size, b_size
2: **Salidas:** $train_gen$, val_gen, $test_gen$
3: **procedure** DEFINIRTRANSFORMACIONES
4: Redimensionar imágenes a img_size
5: Recortar las imágenes a img_size
6: Convertir imágenes a tensores
7: Normalizar tensores con medias y desviaciones estándar $(0.5, 0.5, 0.5)$
8: **end procedure**
9: **procedure** CARGARDATOS
10: Cargar imágenes de entrenamiento de $train_dir$ con transformaciones
11: Cargar imágenes de prueba de $test_dir$ con transformaciones
12: **end procedure**
13: **procedure** DIVIDIRDATOSENTRENAMIENTO
14: Calcular val_size como 15 % de $train_data$
15: Calcular $train_size$ como $len(train_data) - val_size$
16: Dividir de forma aleatoria $train_data$ en $train_data$ y val_data usando $train_size$ y val_size
17: **end procedure**
18: **procedure** CREARDATALOADERS
19: Crear $train_gen$ para $train_data$ con b_size
20: Crear val_gen para val_data con b_size
21: Crear $test_gen$ para $test_data$ con b_size
22: **end procedure**
23: **return** $train_gen$, val_gen, $test_gen$

Actividad 4: Mejorar composición del conjunto de datos

Como se ha mencionado en la sección anterior de este capítulo, las características que determinan si un nódulo puede ser benigno o maligno en un imagen de ultrasonido impiden el uso de las técnicas convencionales de Aumento de Datos. Por este motivo, se construyó un modelo DCGAN con el objetivo de generar imágenes sintéticas.

Inicialmente, la arquitectura del modelo DCGAN siguió los mismos lineamientos definidos en la página oficial de Pytorch por Inkawhich (2024); sin embargo, debido a la posible pérdida de información al reducir considerablemente el tamaño de las imágenes (ya que se pedía como entrada imágenes de 64 x 64), se decidió por personalizar el modelo original para que pueda recibir imágenes de 128 x 128, de la misma manera como se hizo en la investigación presentada por JERBI et al. (2023). Para lograr esto se tuvo que modificar tanto la red generadora como la discriminadora.

En la red generadora se agregaron un bloque nuevo de capas similar a las que ya poseída, modificando únicamente el tamaño de canales de salida de la capa ConvTranspose2d al doble de las que poseía la capa inicial original. El tamaño de kernel, stride y padding de esta misma capa se mantuvieron del formato original. En la capa de BatchNorm2d también se duplicó la cantidad de features que recibía. La función de activación ReLU se mantuvo con la misma configuración inicial. La capas siguientes a este nuevo bloque, fueron adaptándose a la cambiada cantidad de canales de salida.

El tamaño del vector latente que ingresa a la red generadora se definió en 200, con 3 como número de canales y 128 como cantidad base de features map.

En el caso de la red discriminadora, se introdujo un bloque intermedio de capas en la parte final de todo el grupo de bloque de capas similares. La capa Conv2d de este nuevo bloque recibiría la cantidad de canales de salida del bloque anterior, mientras que sus canales de salida serían el doble de los tenía la capa Conv2d anterior. De igual forma, la capa BatchNorm2d duplicó la cantidad de features que recibía originalmente. Se mantuvo la misma configuración de la función de activación LeakyReLU. La capa siguiente a este bloque recién creado se modificó para recibir el doble de canales de los que recibía originalmente.

La cantidad de canales que recibía el modelo se estableció en 3, mientras que el número de features map base a generar se estableció a 32.

Todas las demás configuraciones en ambas redes se dejaron por defecto.

La Figura 15 muestra la arquitectura final de la red generadora.

```
Generador(
  (main): Sequential(
    (0): ConvTranspose2d(200, 2048, kernel_size=(4, 4), stride=(1, 1), bias=False)
    (1): BatchNorm2d(2048, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (2): ReLU(inplace=True)
    (3): ConvTranspose2d(2048, 1024, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (4): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (5): ReLU(inplace=True)
    (6): ConvTranspose2d(1024, 512, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (7): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (8): ReLU(inplace=True)
    (9): ConvTranspose2d(512, 256, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (10): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (11): ReLU(inplace=True)
    (12): ConvTranspose2d(256, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (13): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (14): ReLU(inplace=True)
    (15): ConvTranspose2d(128, 3, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (16): Tanh()
  )
)
```

Figura 15. Arquitectura de Red Generadora.

Fuente: Elaboración propia.

De igual forma, la Figura 16, muestra la arquitectura de la red discriminadora.

```
Discriminador(
  (main): Sequential(
    (0): Conv2d(3, 32, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (1): LeakyReLU(negative_slope=0.2, inplace=True)
    (2): Conv2d(32, 64, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (3): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (4): LeakyReLU(negative_slope=0.2, inplace=True)
    (5): Conv2d(64, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (6): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (7): LeakyReLU(negative_slope=0.2, inplace=True)
    (8): Conv2d(128, 256, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (9): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (10): LeakyReLU(negative_slope=0.2, inplace=True)
    (11): Conv2d(256, 512, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (12): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (13): LeakyReLU(negative_slope=0.2, inplace=True)
    (14): Conv2d(512, 1, kernel_size=(4, 4), stride=(1, 1), bias=False)
    (15): Sigmoid()
  )
)
```

Figura 16. Arquitectura de Red Discriminadora.

Fuente: Elaboración propia.

Para la función de pérdida se usó el Binary Cross Entropy Loss, mientras que la función de optimización se estableció en Adam con una tasa de aprendizaje de 0.0002 y betas de 0.5 para el primero y 0.999 para el segundo. Todo esto según lo establecido por la página oficial de Pytorch.

También se definió un tensor fijo de tamaño 200 (tamaño anteriormente definido de vector latente) inicializado con números aleatorios pero que siguen una distribución normal. Este tensor tenía un lote de tamaño 64.

Para ya iniciar con el entrenamiento del modelo DCGAN se procedió con la creación del DataLoader de Pytorch que contenía todas las imágenes del conjunto de datos de entrenamiento. En este proceso, se definió que las imágenes deberían ser redimensionadas a tamaño 128 y con leídas en un tamaño de lote 64.

El proceso de entrenamiento siguió las características normales del DCGAN original, estableciendo únicamente la cantidad de épocas a 200.

Se usó las capacidades de la GPU NVIDIA A100-SXM4-40GB obtenido a través de Google Colab para este proceso de entrenamiento.

En la Figura 17 se muestran las métricas de la parte final del entrenamiento.

```
[193/200][12/45]        Loss_D: 0.9329  Loss_G: 4.9711  D(x): 0.8268    D(G(z)): 0.3861 / 0.0154
[193/200][24/45]        Loss_D: 1.3177  Loss_G: 3.7289  D(x): 0.4678    D(G(z)): 0.0460 / 0.0918
[193/200][36/45]        Loss_D: 1.1743  Loss_G: 3.2154  D(x): 0.5488    D(G(z)): 0.1099 / 0.1011
[194/200][0/45] Loss_D: 0.7755  Loss_G: 6.3231  D(x): 0.8304    D(G(z)): 0.3657 / 0.0056
[194/200][12/45]        Loss_D: 0.3612  Loss_G: 4.7047  D(x): 0.8011    D(G(z)): 0.0835 / 0.0199
[194/200][24/45]        Loss_D: 0.5590  Loss_G: 4.0930  D(x): 0.7220    D(G(z)): 0.0862 / 0.0439
[194/200][36/45]        Loss_D: 0.7441  Loss_G: 5.7310  D(x): 0.8732    D(G(z)): 0.3288 / 0.0068
[195/200][0/45] Loss_D: 0.3181  Loss_G: 5.2338  D(x): 0.9074    D(G(z)): 0.1495 / 0.0154
[195/200][12/45]        Loss_D: 0.5000  Loss_G: 3.2941  D(x): 0.7293    D(G(z)): 0.0506 / 0.0875
[195/200][24/45]        Loss_D: 0.3884  Loss_G: 3.8248  D(x): 0.8192    D(G(z)): 0.1246 / 0.0390
[195/200][36/45]        Loss_D: 0.3102  Loss_G: 5.1148  D(x): 0.9061    D(G(z)): 0.1469 / 0.0122
[196/200][0/45] Loss_D: 0.2596  Loss_G: 5.6353  D(x): 0.9674    D(G(z)): 0.1730 / 0.0070
[196/200][12/45]        Loss_D: 0.2568  Loss_G: 5.7155  D(x): 0.9619    D(G(z)): 0.1732 / 0.0053
[196/200][24/45]        Loss_D: 0.2564  Loss_G: 4.6884  D(x): 0.9289    D(G(z)): 0.1434 / 0.0184
[196/200][36/45]        Loss_D: 0.1572  Loss_G: 4.6008  D(x): 0.9246    D(G(z)): 0.0659 / 0.0233
[197/200][0/45] Loss_D: 0.1648  Loss_G: 4.7549  D(x): 0.9327    D(G(z)): 0.0775 / 0.0190
[197/200][12/45]        Loss_D: 0.2026  Loss_G: 4.4653  D(x): 0.9404    D(G(z)): 0.1161 / 0.0199
[197/200][24/45]        Loss_D: 0.1449  Loss_G: 4.2249  D(x): 0.9231    D(G(z)): 0.0562 / 0.0249
[197/200][36/45]        Loss_D: 0.1430  Loss_G: 4.1032  D(x): 0.9278    D(G(z)): 0.0587 / 0.0300
[198/200][0/45] Loss_D: 0.2594  Loss_G: 5.9415  D(x): 0.9500    D(G(z)): 0.1662 / 0.0057
[198/200][12/45]        Loss_D: 0.2385  Loss_G: 3.6982  D(x): 0.8441    D(G(z)): 0.0517 / 0.0529
[198/200][24/45]        Loss_D: 0.1866  Loss_G: 4.9649  D(x): 0.9299    D(G(z)): 0.0914 / 0.0156
[198/200][36/45]        Loss_D: 0.2030  Loss_G: 4.1795  D(x): 0.9273    D(G(z)): 0.1074 / 0.0304
[199/200][0/45] Loss_D: 0.1119  Loss_G: 4.4837  D(x): 0.9500    D(G(z)): 0.0545 / 0.0211
[199/200][12/45]        Loss_D: 0.2235  Loss_G: 4.3183  D(x): 0.8310    D(G(z)): 0.0195 / 0.0336
[199/200][24/45]        Loss_D: 0.1580  Loss_G: 4.5593  D(x): 0.9082    D(G(z)): 0.0433 / 0.0192
[199/200][36/45]        Loss_D: 0.1250  Loss_G: 4.5769  D(x): 0.9679    D(G(z)): 0.0787 / 0.0158
```

Figura 17. Parte final del entrenamietno del DCGAN.

Fuente: Elaboración propia.

Además, se realizó un plot de las pérdidas de ambas redes a través de cada iteración del entrenamiento del modelo. Este se muestra en la Figura 18.

Figura 18. Plot de pérdidas de las red Generadora y Discriminadora.

Fuente: Elaboración propia.

En esta última gráfica se puede observar cómo ambas pérdidas van convergiendo a más iteraciones se tiene; sin embargo, ya en la parte final se logra observar la divergencia de ambos valores. Esto podría deberse a un sobreajuste por parte de la red discriminadora, al mismo tiempo que la red generadora deja de mejorar en la tarea de generar imágenes sintéticas.

Las imágenes reales y generadas se muestran en la Figura 19.

Figura 19. Imágenes de ultrasonido de nódulos tiroideos generados y reales.

Fuente: Elaboración propia.

Según un rápido análisis visual, la imágenes generadas sí tienen un poco de similitud a las originales, y fácilmente, para alguien no experto, estos podrían pasar como imágenes reales.

Ya que el entrenamiento se realizó con todas las imágenes la data de entrenamiento, todas las imágenes generadas podrían pertenecer a cualquiera de las clases (benigno o maligno) y se es difícil sino imposible determinar a qué tipo pertenecen (benigno o maligno). Por este motivo, y viendo el potencial del DCGAN, se volvió a entrenar el modelo ahora solo con imágenes de la clase maligno (clase minoritaria) del conjunto de datos de entrenamiento para que la red solo pueda generar imágenes de ultrasonido de esta clase en específico.

En este caso se modificó el tamaño de vector latente a 300 y los features map base a 256 en la red generadora. Las demás configuraciones se mantuvieron, incluyendo aquellas de la red discriminadora.

Se usaron la misma función de pérdida y optimización con los mismos parámetros definidos anteriormente.

El tensor fijo de números aleatorio de distribución normal tuvo para este entrenamiento un tamaño de 300 (igual al tamaño del vector latente).

Para el DataLoader se mantuvo el mismo tamaño de redimensionamiento y lote.

Debido a una prueba rápida inicial con una menor cantidad de épocas, donde se obtuvieron pobres resultados, se decidió aumentar la cantidad de iteraciones totales estableciendo el número de épocas a 370.

La Figura 20 muestra la parte final del entrenamiento.

```
[357/370][12/16]        Loss_D: 0.0833  Loss_G: 5.5940  D(x): 0.9683    D(G(z)): 0.0461 / 0.0082
[358/370][0/16] Loss_D: 0.2755  Loss_G: 9.7520  D(x): 0.9916    D(G(z)): 0.2037 / 0.0002
[358/370][12/16]        Loss_D: 0.0470  Loss_G: 4.3363  D(x): 0.9884    D(G(z)): 0.0339 / 0.0186
[359/370][0/16] Loss_D: 0.0375  Loss_G: 5.2440  D(x): 0.9854    D(G(z)): 0.0221 / 0.0124
[359/370][12/16]        Loss_D: 0.0677  Loss_G: 4.9654  D(x): 0.9722    D(G(z)): 0.0369 / 0.0126
[360/370][0/16] Loss_D: 0.9652  Loss_G: 15.0713 D(x): 0.9989    D(G(z)): 0.4774 / 0.0000
[360/370][12/16]        Loss_D: 0.6778  Loss_G: 8.4812  D(x): 0.8195    D(G(z)): 0.2315 / 0.0022
[361/370][0/16] Loss_D: 3.2693  Loss_G: 12.7542 D(x): 0.9747    D(G(z)): 0.7195 / 0.0001
[361/370][12/16]        Loss_D: 0.3831  Loss_G: 5.9367  D(x): 0.7973    D(G(z)): 0.0664 / 0.0114
[362/370][0/16] Loss_D: 0.5647  Loss_G: 10.0312 D(x): 0.9745    D(G(z)): 0.3305 / 0.0002
[362/370][12/16]        Loss_D: 0.3276  Loss_G: 7.1568  D(x): 0.9746    D(G(z)): 0.2275 / 0.0023
[363/370][0/16] Loss_D: 0.2405  Loss_G: 6.0655  D(x): 0.9381    D(G(z)): 0.1303 / 0.0055
[363/370][12/16]        Loss_D: 0.1617  Loss_G: 5.3671  D(x): 0.9200    D(G(z)): 0.0540 / 0.0104
[364/370][0/16] Loss_D: 0.1563  Loss_G: 5.4059  D(x): 0.9934    D(G(z)): 0.1257 / 0.0088
[364/370][12/16]        Loss_D: 0.1329  Loss_G: 5.7556  D(x): 0.9664    D(G(z)): 0.0861 / 0.0076
[365/370][0/16] Loss_D: 0.1250  Loss_G: 5.2169  D(x): 0.9763    D(G(z)): 0.0872 / 0.0103
[365/370][12/16]        Loss_D: 0.0713  Loss_G: 5.1367  D(x): 0.9475    D(G(z)): 0.0139 / 0.0126
[366/370][0/16] Loss_D: 0.0458  Loss_G: 5.1208  D(x): 0.9712    D(G(z)): 0.0158 / 0.0105
[366/370][12/16]        Loss_D: 0.0569  Loss_G: 4.9588  D(x): 0.9817    D(G(z)): 0.0365 / 0.0112
[367/370][0/16] Loss_D: 0.0961  Loss_G: 5.2228  D(x): 0.9641    D(G(z)): 0.0523 / 0.0113
[367/370][12/16]        Loss_D: 0.0822  Loss_G: 4.7507  D(x): 0.9696    D(G(z)): 0.0486 / 0.0124
[368/370][0/16] Loss_D: 0.0608  Loss_G: 4.8219  D(x): 0.9706    D(G(z)): 0.0290 / 0.0132
[368/370][12/16]        Loss_D: 0.0572  Loss_G: 4.9233  D(x): 0.9681    D(G(z)): 0.0237 / 0.0129
[369/370][0/16] Loss_D: 0.0351  Loss_G: 5.3250  D(x): 0.9875    D(G(z)): 0.0219 / 0.0126
[369/370][12/16]        Loss_D: 0.0829  Loss_G: 5.8338  D(x): 0.9801    D(G(z)): 0.0581 / 0.0070
```

Figura 20. Parte final del entrenamietno del DCGAN con imágenes de nódulos malignos.

Fuente: Elaboración propia.

De igual manera al entrenamiento inicial, se realizó una gráfica donde se muestra el comportamiento de la pérdida en ambas redes. Esto se muestra en la Figura 21.

Figura 21. Plot de pérdidas de las red Generadora y Discriminadora de imágenes de nódulos malignos.

Fuente: Elaboración propia.

Esta gráfica muestra el mismo comportamiento que a más iteraciones, mayor divergen-

cia entre las pérdidas.

En la Figura 22 se presentan las imágenes generadas junto a las originales.

Figura 22. Imágenes de ultrasonido de nódulos tiroideos generados y reales (DCGAN de imágenes de nódulos malignos).

Fuente: Elaboración propia.

Los resultados de las imágenes generadas en este modelo muestran mayor ruido comparado las generadas con el entrenamiento inicial, esto se debe principalmente a la menor cantidad de datos usada para el entrenamiento de este nuevo DCGAN. Sin embargo, luego de un rápido análisis visual, se puede observar un comportamiento similar entre todas estas imágenes nuevas, y es que en la mayoría de estas hay lo que parece ser la presencia de un nódulo, aunque difuminado con mucho ruido aleatorio, esto podría indicar un aprendizaje cercano a la distribución de las imágenes de nódulos malignos.

Intentando utilizar el potencial que ofrece el DCGAN, se realizó posteriormente dos tipos de entrenamiento a los modelos de Deep Learning: el primero sin el Aumento de Datos a través de DCGAN, mientras que el segundo combinará los datos generados con la data original de entrenamiento para realizar dicho proceso. Esto permitirá determinar si las imágenes generadas por el DCGAN, aunque con ruido, son capaces de aumentar la capacidad de predicción de los modelos.

La gráfica circula presenta en la Figura 23 muestra el nuevo balance entre ambas clases en la nueva data de entrenamiento.

Figura 23. Gráfica circula de data de entrenamiento balanceada.

Fuente: Elaboración propia.

Ya con la data preprocesada, se inició con la etapa de Aplicación de Deep Learning donde se entrenó, validó y puso a prueba los diversos modelos CNN y ViT. Esta etapa inicia con la siguiente actividad.

Actividad 1: Seleccionar algoritmos a entrenar según usos y resultados de las investigaciones presentadas en los antecedentes

Según los datos analizados en las investigaciones presentadas en el Capítulo 2, el modelo con mayor presencia de los CNN es el ResNet50, mientras que por el lado de los modelos ViT, los más usados son los ViT-Base con tamaño de patch de 16 x 16.

Además, debido al tipo de problema de clasificación presente en esta investigación, y la similitud con el trabajo de JERBI et al. (2023), se decidió por probar con las redes VGG16, ya que ha demostrado ser capaz de obtener un alto desempeño.

En resumen, se ha optado por utilizar las arquitecturas CNN de ResNet50 y VGG16. Además, como parte de las arquitecturas ViT, se optó por el uso de ViT-Base (16).

Actividad 2: Entrenar y probar los modelos de Deep Learning previamente definidos

Para todas las arquitecturas de Deep Learning en esta investigación, se definió como función de activación el sigmoide. De igual manera, para el entrenamiento se usó la capacidad de la

GPU NVIDIA A100-SXM4-40GB obtenido a través de Google Colab para lograr un rápido y eficiente entrenamiento. Además, se determinó una semilla de 2024 para permitir la reproducibilidad de los resultados.

En este primer grupo, no se utilizará el conjunto de datos con imágenes generadas.

Se empezó con los modelos ViT-Base (16).

Para este proceso se crearon los DataLoader correspondientes con las características requeridas por estos modelos ViT: se redimensionaron las imágenes a 224 x 224 y se definió un tamaño de lote de 64 (este último tamaño será constante para todos los modelos mostrados en esta actividad).

En este primer modelo, se usó la técnica de entrenamiento de Fine Tuning junto con el modelo ViT-Base (16) con pesos de IMAGENET1KV1.

La función de pérdida aplicada fue el BCEWithLogitsLoss que evita el uso manual de la sigmoide y lo aplica directamente con la entropía cruzada. Nuevamente, esta función será utilizada en cada uno de los modelos en esta actividad.

Para lograr una clasificación binaria, las últimas capas de estas arquitecturas debió ser reemplazada por un MLP simple que recibe todos los features del modelo y produce una única salida. La configuración del MLP se definió de la forma (256, 1), donde el primer valor es el número de neuronas en la capas oculta que tuvo la función de activación ReLU, y el segundo valor es el número de neuronas en la capa de salida. Esta configuración de MLP se aplicó a todos los modelos ViT y CNN, a excepción de VGG16, donde se usó una configuración por defecto, únicamente modificando a 1 el número de neuronas en la capa de salida. En el caso de los modelos híbridos, solamente se estableció el bloque clasificador a una capa de salida con 1 neurona, debido a las limitaciones de disponibilidad de recursos computacionales.

Finalmente, la función de optimización utilizada fue Adam con una tasa de aprendizaje de 0.0001, y se definió para el entrenamiento 20 épocas. Las demás configuraciones se dejaron por defecto.

Estas configuraciones se aplicarán de igual manera a los demás modelos en esta actividad.

A continuación se presenta en la Figura 24 el proceso de entrenamiento del modelo ViT-Base (16) con los pesos IMAGENET1KV1 y usando la técnica de Fine Tuning.

Figura 24. Proceso de entrenamiento del modelo n°1.

Fuente: Elaboración propia.

Finalmente, para este modelo también se calcularon su Accuracy, Recall y Precision. Estos son presentados en la Tabla 2.

Tabla 2
Accuracy, Recall y Precision del modelo n°1.

Métrica	Accuracy	Recall	Precision
Valor	69.54 %	47.03 %	64.16 %

Fuente: Elaboración propia.

En el siguiente modelo, se usó la técnica de entrenamiento de Transfer Learning junto con el modelo ViT-Base (16) con pesos de IMAGENET1KV1.

El DataLoader constaba de las imágenes redimensionadas nuevamente a 224 x 224, y se aplicó la función de pérdida BCEWithLogitsLoss junto con la función de optimización Adam.

A continuación se presenta en la Figura 25 el proceso de entrenamiento del modelo ViT-Base (16) con los pesos IMAGENET1KV1 y usando la técnica de Transfer Learning.

Figura 25. Proceso de entrenamiento del modelo n°2.

Fuente: Elaboración propia.

También se calcularon su Accuracy, Recall y Precision. Estos son presentados en la Tabla 3.

Tabla 3
Accuracy, Recall y Precision del modelo n°2.

Métrica	Accuracy	Recall	Precision
Valor	68.08 %	53.39 %	59.43 %

Fuente: Elaboración propia.

Este modelo usó la técnica de entrenamiento de Fine Tuning junto con el modelo ViT-Base (16) con pesos de IMAGENET1KSWAGE2EV1.

En este caso, el DataLoader constaba de las imágenes redimensionadas a 384 x 384.

Se presenta en la Figura 26 el proceso de entrenamiento del modelo ViT-Base (16) con los pesos IMAGENET1KSWAGE2EV1 y usando la técnica de Fine Tuning.

Figura 26. Proceso de entrenamiento del modelo n°3.

Fuente: Elaboración propia.

También se calcularon su Accuracy, Recall y Precision. Estos son presentados en la Tabla 4.

Tabla 4
Accuracy, Recall y Precision del modelo n°3.

Métrica	Accuracy	Recall	Precision
Valor	56.84 %	53.81 %	44.88 %

Fuente: Elaboración propia.

El modelo n°4 empleó la técnica de entrenamiento de Transfer Learning junto con el modelo ViT-Base (16) con pesos de IMAGENET1KSWAGE2EV1.

En este caso, nuevamente el DataLoader constaba de las imágenes redimensionadas a 384 x 384.

Se presenta en la Figura 27 el proceso de entrenamiento del modelo ViT-Base (16) con los pesos IMAGENET1KSWAGE2EV1 y usando la técnica de Transfer Learning.

Figura 27. Proceso de entrenamiento del modelo n°4.

Fuente: Elaboración propia.

También se calcularon su Accuracy, Recall y Precision. Estos son presentados en la Tabla 5.

Tabla 5
Accuracy, Recall y Precision del modelo n°4.

Métrica	Accuracy	Recall	Precision
Valor	69.22 %	55.51 %	60.93 %

Fuente: Elaboración propia.

El modelo n°5 empleó la técnica de entrenamiento de Fine Tuning junto con el modelo ViT-Base (16) con pesos de IMAGENET1KSWAGLINEARV1.

En este caso, el DataLoader constaba de las imágenes redimensionadas a 224 x 224.

Se presenta en la Figura 28 el proceso de entrenamiento del modelo ViT-Base (16) con los pesos IMAGENET1KSWAGLINEARV1 y usando la técnica de Fine Tuning.

Figura 28. Proceso de entrenamiento del modelo n°5.

Fuente: Elaboración propia.

También se calcularon su Recall y Precision. Estos son presentados en la Tabla 6.

Tabla 6
Accuracy, Recall y Precision del modelo n°5.

Métrica	Accuracy	Recall	Precision
Valor	58.47 %	34.75 %	44.81 %

Fuente: Elaboración propia.

El modelo n°6 empleó la técnica de entrenamiento de Transfer Learning junto con el modelo ViT-Base (16) con pesos de IMAGENET1KSWAGLINEARV1.

En este caso, el DataLoader constaba nuevamente de las imágenes redimensionadas a 224 x 224.

Se presenta en la Figura 29 el proceso de entrenamiento del modelo ViT-Base (16) con los pesos IMAGENET1KSWAGLINEARV1 y usando la técnica de Transfer Learning.

Figura 29. Proceso de entrenamiento del modelo n°6.

Fuente: Elaboración propia.

También se calcularon su Accuracy, Recall y Precision. Estos son presentados en la Tabla 7.

Tabla 7
Accuracy, Recall y Precision del modelo n°6.

Métrica	Accuracy	Recall	Precision
Valor	65.31 %	41.95 %	56.57 %

Fuente: Elaboración propia.

El modelo n°7 empleó la técnica de entrenamiento de Fine Tuning junto con el modelo VGG16 con pesos de IMAGENET1KV1.

En este caso, el DataLoader fue cargado con imágenes redimensionadas a 128 x 128. Este tamaño será constante para los demás modelos CNN.

Se presenta en la Figura 30 el proceso de entrenamiento del modelo VGG16 con los pesos IMAGENET1KV1 y usando la técnica de Fine Tuning.

Figura 30. Proceso de entrenamiento del modelo n°7.

Fuente: Elaboración propia.

También se calcularon su Accuracy, Recall y Precision. Estos son presentados en la Tabla 8.

Tabla 8
Accuracy, Recall y Precision del modelo n°7.

Métrica	Accuracy	Recall	Precision
Valor	71.99 %	58.90 %	64.95 %

Fuente: Elaboración propia.

El modelo n°8 empleó la técnica de entrenamiento de Transfer Learning junto con el modelo VGG16 con pesos de IMAGENET1KV1.

Se presenta en la Figura 31 el proceso del entrenamiento del modelo VGG16 con los pesos IMAGENET1KV1 y usando la técnica de Transfer Learning.

Figura 31. Proceso de entrenamiento del modelo n°8.

Fuente: Elaboración propia.

También se calcularon su Accuracy, Recall y Precision. Estos son presentados en la Tabla 9.

Tabla 9
Accuracy, Recall y Precision del modelo n°8.

Métrica	Accuracy	Recall	Precision
Valor	65.80 %	67.37 %	54.45 %

Fuente: Elaboración propia.

El modelo n°9 empleó la técnica de entrenamiento de Fine Tuning junto con el modelo RESNET50 con pesos de IMAGENET1KV1.

Se presenta en la Figura 32 el proceso del entrenamiento del modelo RESNET50 con los pesos IMAGENET1KV1 y usando la técnica de Fine Tuning.

Figura 32. Proceso de entrenamiento del modelo n°9.

Fuente: Elaboración propia.

También se calcularon su Accuracy, Recall y Precision. Estos son presentados en la Tabla 10.

Tabla 10
Accuracy, Recall y Precision del modelo n°9.

Métrica	Accuracy	Recall	Precision
Valor	72.48 %	58.90 %	65.88 %

Fuente: Elaboración propia.

El modelo n°10 empleó la técnica de entrenamiento de Transfer Learning junto con el modelo RESNET50 con pesos de IMAGENET1KV1.

Se presenta en la Figura 33 el proceso del entrenamiento del modelo RESNET50 con los pesos IMAGENET1KV1 y usando la técnica de Transfer Learning.

Figura 33. Proceso de entrenamiento del modelo n°10.

Fuente: Elaboración propia.

También se calcularon su Accuracy, Recall y Precision. Estos son presentados en la Tabla 11.

Tabla 11
Accuracy, Recall y Precision del modelo n°10.

Métrica	Accuracy	Recall	Precision
Valor	62.38 %	37.71 %	51.45 %

Fuente: Elaboración propia.

El modelo n°11 empleó la técnica de entrenamiento de Fine Tuning junto con el modelo RESNET50 con pesos de IMAGENET1KV2.

Se presenta en la Figura 34 el proceso del entrenamiento del modelo RESNET50 con los pesos IMAGENET1KV2 y usando la técnica de Fine Tuning.

Figura 34. Proceso de entrenamiento del modelo n°11.

Fuente: Elaboración propia.

También se calcularon su Accuracy, Recall y Precision. Estos son presentados en la Tabla 12.

Tabla 12
Accuracy, Recall y Precision del modelo n°11.

Métrica	Accuracy	Recall	Precision
Valor	67.59 %	51.27 %	59.02 %

Fuente: Elaboración propia.

El modelo n°12 empleó la técnica de entrenamiento de Transfer Learning junto con el modelo RESNET50 con pesos de IMAGENET1KV2.

Se presenta en la Figura 35 el proceso del entrenamiento del modelo RESNET50 con los pesos IMAGENET1KV2 y usando la técnica de Transfer Learning.

Figura 35. Proceso de entrenamiento del modelo n°12.

Fuente: Elaboración propia.

También se calcularon su Accuracy, Recall y Precision. Estos son presentados en la Tabla 13.

Tabla 13
Accuracy, Recall y Precision del modelo n°12.

Métrica	Accuracy	Recall	Precision
Valor	62.05 %	49.15 %	50.66 %

Fuente: Elaboración propia.

Una vez culminado el entrenamiento de los modelos con el conjunto de datos sin imágenes generadas, se inició un nuevo proceso de entrenamiento, esta vez con el conjunto de datos aumentado.

Todos los modelos, tanto ViT como CNN, poseen las mismas configuraciones definidas antes, así que no hay necesidad de especificarlo nuevamente.

A continuación se presentan las siguientes Figuras y Tablas donde se muestran todos los resultados de los modelos de manera similar a como se presentó anteriormente.

Figura 36. Proceso de entrenamiento del modelo n°13.

Fuente: Elaboración propia.

Tabla 14
Accuracy, Recall y Precision del modelo n°13.

Métrica	Accuracy	Recall	Precision
Valor	70.36 %	61.44 %	61.44 %

Fuente: Elaboración propia.

Figura 37. Proceso de entrenamiento del modelo n°14.

Fuente: Elaboración propia.

Tabla 15
Accuracy, Recall y Precision del modelo n°14.

Métrica	Accuracy	Recall	Precision
Valor	69.06 %	50.00 %	62.11 %

Fuente: Elaboración propia.

Figura 38. Proceso de entrenamiento del modelo n°15.

Fuente: Elaboración propia.

Tabla 16
Accuracy, Recall y Precision del modelo n°15.

Métrica	Accuracy	Recall	Precision
Valor	67.43 %	41.10 %	61.39 %

Fuente: Elaboración propia.

Figura 39. Proceso de entrenamiento del modelo n°16.

Fuente: Elaboración propia.

Tabla 17
Accuracy, Recall y Precision del modelo n°16.

Métrica	Accuracy	Recall	Precision
Valor	69.71 %	64.83 %	59.77 %

Fuente: Elaboración propia.

Figura 40. Proceso de entrenamiento del modelo n°17.

Fuente: Elaboración propia.

Tabla 18
Accuracy, Recall y Precision del modelo n°17.

Métrica	Accuracy	Recall	Precision
Valor	57.65 %	41.95 %	44.59 %

Fuente: Elaboración propia.

Figura 41. Proceso de entrenamiento del modelo n°18.

Fuente: Elaboración propia.

Tabla 19
Accuracy, Recall y Precision del modelo n°18.

Métrica	Accuracy	Recall	Precision
Valor	64.82 %	49.58 %	54.67 %

Fuente: Elaboración propia.

Figura 42. Proceso de entrenamiento del modelo n°19.

Fuente: Elaboración propia.

Tabla 20
Accuracy, Recall y Precision del modelo n°19.

Métrica	Accuracy	Recall	Precision
Valor	71.99 %	59.75 %	64.68 %

Fuente: Elaboración propia.

Figura 43. Proceso de entrenamiento del modelo n°20.

Fuente: Elaboración propia.

Tabla 21
Accuracy, Recall y Precision del modelo n°20.

Métrica	Accuracy	Recall	Precision
Valor	66.45 %	52.97 %	56.82 %

Fuente: Elaboración propia.

Figura 44. Proceso de entrenamiento del modelo n°21.

Fuente: Elaboración propia.

Tabla 22
Accuracy, Recall y Precision del modelo n°21.

Métrica	Accuracy	Recall	Precision
Valor	70.85 %	68.22 %	60.75 %

Fuente: Elaboración propia.

Figura 45. Proceso de entrenamiento del modelo n°22.

Fuente: Elaboración propia.

Tabla 23
Accuracy, Recall y Precision del modelo n°22.

Métrica	Accuracy	Recall	Precision
Valor	64.17 %	33.90 %	55.56 %

Fuente: Elaboración propia.

Figura 46. Proceso de entrenamiento del modelo n°23.

Fuente: Elaboración propia.

Tabla 24
Accuracy, Recall y Precision del modelo n°23.

Métrica	Accuracy	Recall	Precision
Valor	67.92 %	55.51 %	58.74 %

Fuente: Elaboración propia.

Figura 47. Proceso de entrenamiento del modelo n°24.

Fuente: Elaboración propia.

Tabla 25
Accuracy, Recall y Precision del modelo n°24.

Métrica	Accuracy	Recall	Precision
Valor	62.87 %	47.03 %	51.87 %

Fuente: Elaboración propia.

Actividad 3: Definir modelos híbridos entre ViT y CNN

Luego del entrenamiento y un análisis rápido de resultados de los modelos CNN y ViT, se determinó que la arquitectura de mejor desempeño fue ResNet50 con los pesos de IMAGENET1KV1 entrenado con la técnica de Fine Tuning y sin la data aumentada, pues su métrica Accuracy superó a los demás modelos. En segundo lugar se ubica el modelo VGG16 con los pesos de IMAGENET1KV1 entrenado con la técnica de Fine Tuning y sin la data aumentada.

Se decidió tomar ambos modelos CNN para la construcción de modelos híbridos con ViT. Para lograr esto, se usó la librería “transformers” de Python que se encuentra actualmente mantenida por Hugging Face. Esta librería permitió personalizar un modelo ViT híbrido base, lo cuál facilitó su construcción, a diferencia de las herramientas incluidas en mismo Pytorch, específicamente Torchvision, donde los modelos ViT venían de forma predefinida y dificultaban su personalización de acuerdo a las necesidades de esta investigación.

Es así que, para lograr un correcta integración de los modelos ResNet50 y VGG16 con ViT, fue necesario configurar todos los híbridos con un nuevo tamaño de features que recibirá

la arquitectura ViT dependiendo del tipo de backbone, además, el tamaño de la imagen de entrada fue establecida a 128 x 128. El tamaño de patch se estableció en 1 (valor por defecto en la creación del modelo) de igual forma a cómo se define el paper de JERBI et al. (2023) que fue fuente original y de inspiración para esta investigación. Finalmente, se configuró el bloque clasificador del ViT a un MLP con dos capas de configuración (256, 1) y con una función de activación ReLU en la primera de estas.

De forma adicional, se usó el modelo híbrido desarrollado por Google y preentrando con el conjunto de datos IMAGENET21K: ViT-hybrid-base-bit-384. Este modelo usa como backbone una versión modificada de las arquitecturas ResNet. Para lograr adaptarlo a la tarea de esta investigación, se modificó su bloque clasificador a un MLP con una sola capa de una neurona.

Actividad 4: Entrenar y probar modelos híbridos CNN + ViT

Ya con los modelos híbridos construidos se pasó a la etapa de entrenamiento.

Debido a que los modelos CNN ResNet50 y VGG16 de mayor desempeño se entrenaron con los pesos de IMAGENET1KV1 con Fine Tuning y sin la data aumentada, se usó esta misma configuración para los modelos híbridos. De igual forma, la función de pérdida y de optimización se mantuvieron como los anteriores modelos.

En el caso del modelo híbrido de Google, la única diferencia fue que se estableció el tamaño de lote a 32 debido a las limitaciones de hardware.

Con todas esta especificaciones, se inició el entrenamiento de los modelos híbridos con 20 épocas.

A continuación, en la Figura 48 se presenta el proceso de entrenamiento del modelo híbrido ViT + ResNet50 con los pesos pre-entrenados de IMAGENET1KV1 y el tipo de entrenamiento Fine Tuning.

Figura 48. Proceso de entrenamiento del modelo híbrido ViT + ResNet50.

Fuente: Elaboración propia.

Además, también se presenta las métricas de Accuracy, Recall y Precision en la Tabla 26.

Tabla 26
Accuracy, Recall y Precision del modelo híbrido ViT + ResNet50.

Métrica	Accuracy	Recall	Precision
Valor	57.00 %	48.73 %	44.57 %

Fuente: Elaboración propia.

De igual manera, en la Figura 49 se presenta el proceso de entrenamiento del modelo híbrido ViT + VGG16 con los pesos pre-entrenados de IMAGENET1KV1 y el tipo de entrenamiento Fine Tuning.

Figura 49. Proceso de entrenamiento del modelo híbrido ViT + VGG16.

Fuente: Elaboración propia.

Se presenta las métricas de Accuracy, Recall y Precision en la Tabla 27.

Tabla 27
Accuracy, Recall y Precision del modelo híbrido ViT + VGG16.

Métrica	Accuracy	Recall	Precision
Valor	55.37 %	44.49 %	42.34 %

Fuente: Elaboración propia.

Finalmente, en la Figura 50 se presenta el proceso de entrenamiento del modelo híbrido de Google con los pesos pre-entrenados de IMAGENET21k y el tipo de entrenamiento Fine Tuning.

Figura 50. Proceso de entrenamiento del modelo híbrido de Google.

Fuente: Elaboración propia.

Se presenta las métricas de Accuracy, Recall y Precision en la Tabla 28.

Tabla 28
Accuracy, Recall y Precision del modelo híbrido de Google.

Métrica	Accuracy	Recall	Precision
Valor	77.20 %	77.97 %	67.65 %

Fuente: Elaboración propia.

4.3 Medición de la solución

4.3.1 Análisis de Indicadores cuantitativo

Los indicadores a analizar en esta sección son los obtenidos en el entrenamiento de todos los modelos CNN y ViT, incluyendo además los modelos híbridos. Cada uno de estos dio como resultado principal tres indicadores: Accuracy, Recall y Precision. El primero de estos permitió determinar la capacidad de predicción para ambas clases. El segundo y tercer indicador ayudan a evaluar el modelo en cuestión de capacidad de predicción para la clase positiva. En el caso de esta investigación, la clase positiva es maligno (1).

En la Tabla 29 se presentan los resultados de los modelos.

Tabla 29
Resultados de los modelos entrenados con los dos conjuntos de datos, técnicas de entrenamiento y pesos.

Aumento de datos con DCGAN	Modelo	Pesos	Tipo de entrena-miento	Accuracy	Recall	Precision
No	ViT B 16	I-NET1K V1	FT	69.54 %	47.03 %	64.16 %
			TL	68.08 %	53.39 %	59.43 %
		I-NET1K SE V1	FT	56.84 %	53.81 %	44.88 %
			TL	69.22 %	55.51 %	60.93 %
		I-NET1K SL V1	FT	58.47 %	34.75 %	44.81 %
			TL	65.31 %	41.95 %	56.57 %
	VGG16	I-NET1K V1	FT	71.99 %	58.90 %	64.95 %
			TL	65.80 %	67.37 %	54.45 %
	RESNET50	I-NET1K V1	FT	72.48 %	58.90 %	65.88 %
			TL	62.38 %	37.71 %	51.45 %
		I-NET1K V2	FT	67.59 %	51.27 %	59.02 %
			TL	62.05 %	49.15 %	50.66 %
Sí	ViT B 16	I-NET1K V1	FT	70.36 %	61.44 %	61.44 %
			TL	69.06 %	50.00 %	62.11 %
		I-NET1K SE V1	FT	67.43 %	41.10 %	61.39 %
			TL	69.71 %	64.83 %	59.77 %
		I-NET1K SL V1	FT	57.65 %	41.95 %	44.59 %
			TL	64.82 %	49.58 %	54.67 %
	VGG16	I-NET1K V1	FT	71.99 %	59.75 %	64.68 %
			TL	66.45 %	52.97 %	56.82 %
	RESNET50	I-NET1K V1	FT	70.85 %	68.22 %	60.75 %
			TL	64.17 %	33.90 %	55.56 %
		I-NET1K V2	FT	67.92 %	55.51 %	58.74 %
			TL	62.87 %	47.03 %	51.87 %
No	ViT + ResNet50	I-NET1K V1	FT	57.00 %	48.73 %	44.57 %
No	ViT + VGG16	I-NET1K V1	FT	55.37 %	44.49 %	42.34 %
No	Híbrido Google	I-NET 21k	FT	**77.20 %**	**77.97 %**	**67.65 %**

Fuente: Elaboración propia.

En relación al indicador Accuracy, el modelo de mejor desempeño fue el modelo híbrido de Google con los pesos IMAGENET21k y la técnica de entrenamiento Fine Tuning, junto con el uso del conjunto de datos original, alcanzado un valor de 77.20 %. En este mismo modelo, las otras dos métricas también fueron los más altos, siendo el caso del Recall (77.97 %) el valor más alto y con un gran margen respecto al Recall de los demás modelos, mientras que el Precision (67.65 %) superó, aunque por menor margen, al segundo mejor modelo en esta métrica.

Algo que mecionar sobre los demás modelos no híbridos es que no hubo una gran diferencia entre los resultados de usar la data aumentada y la origianl, pues el DCGAN no logró capturar al distribución de los imágenes originales, por que su impacto en el entrenamiento no

fue tan alto. Esto es debido principalmente a que esta arquitectura fue entrenada con pocos datos (clase minoritaria), lo cuál desencadenó que finalmente las imágenes generadas no tuvieron gran parecido a las originales.

Debido a la naturaleza del problema de esta investigación, se considera que las métricas de Recall y Precision deben ser relativamente altas para lograr seleccionar el modelo, de igual manera, el Accuracy, que mide la capacidad de predicción para ambar clases, también debe ser alta, ya que es importante determinar no solo si un nódulo es malingo, sino también benigno, ya que esto determinará el tipo de tratamiento posterior que se recomendará a los pacientes. Por este motivo, se decidió elegir el modelo híbrido de Google con los pesos IMAGENET21K y la técnica de entrenamiento Fine Tuning como el modelo para la simulación, ya que posee un Accuracy, Recall y Precision superior a los demás modelos.

4.3.2 Simulación de solución

Luego del análisis de los indicadores de los modelos, y de la elección final, se decidió realizar una simulación de uso del modelo con imágenes del conjunto de datos de prueba.

Para lograr esto, se usó la librería ipywidgets de python que permitió, a través de un mismo jupyter notebook, subir una imagen e ingresarlo al modelo ya cargado y obtener finalmente el resultado de su predicción. En la Figura 51 se muestra el cargado de las imágenes, mientras que en la Figura 52 y Figura 53 se muestra la prueba con dos imágenes de ultrasonido de distintas clases.

Figura 51. Cargado de imágenes de ultrasonido para prueba del modelo.

Fuente: Elaboración propia.

Figura 52. Prueba de modelo con imágen de ultrasonido de nódulo maligno.

Fuente: Elaboración propia.

Figura 53. Prueba de modelo con imágen de ultrasonido de nódulo benigno.

Fuente: Elaboración propia.

Capítulo V: Conclusiones y Recomendaciones

A través de toda esta investigación, se ha ido explorando las necesidades y posibles métodos de solución en la problemática del prediagnóstico de nódulos tiroideos a través de imágenes de ultrasonido. Lo más resaltante y constantemente repetido por otros investigadores es la alta urgencia que hay por conocer si un paciente posee un nódulo de carácter benigno o maligno, ya que a través de esta información, los especialistas pueden tomar mejores y más rápida decisiones que pueden mejorar la calidad de vida de las personas. Antes esto, se ha visto el alto desarrollo de Sistema de Diagnóstico Asistido capaces de ayudar y agilizar el proceso de diagnóstico realizado por los médicos. De entre los distintos tipos de este sistema, los actualmente de mayor desempeño son aquellos basados en algoritmos de Inteligencia Artificial, específicamente aquellos que trabajan con imágenes como lo son los CNN. Es por este motivo que en la presente investigación se ha desarrollado, a través de distintos algoritmos de Deep Learning, una herramientas capaz de acelerar el proceso de diagnóstico.

Para ello, fue necesario determinar en primer lugar aquellas características que el conjunto de datos a usar para entrenar los algoritmos debió poseer para lograr un modelo capaz de generalizar en todas de imágenes de ultrasonido de nódulos en la tiroides. Es así que se encontró la necesidad de que el conjunto de datos tengan una cantidad de datos elevada para ambas clases de nódulos, algo que fue difícil encontrar debido a que la mayoría de conjuntos de datos de este tipo no son de libre acceso. Otra característica importante fue la validez de las etiquetas de las imágenes, ya que muchas veces se puede encontrar inconsistencias entre datos de distintos tipos que pueden definir si un nódulo es benigno o maligno. Finalmente, fue necesario encontrar datos que vengan de distintas fuentes; es decir, las imágenes de ultrasonido deben ser obtenidas a través de distintos dispositivos, esto para lograr una mayor capacidad de generalización de los modelos.

De igual manera, determinar aquellas técnicas de preprocesamiento de imágenes que deben ser aplicadas fue de gran utilidad ya que debido a la naturaleza y características que permiten el diagnóstico de un nódulo a través de imágenes de ultrasonido, no permitieron utilizar las técnicas convencionales de Aumento de Datos. En primer lugar se encontró que el clásico preprocesamiento de imágenes de reducción de dimensiones sí fue necesario debido a que esto reduce la necesidad de mayor capacidad computacional, de igual forma, por el mismo motivo, se aplicó la Normalización de las imágenes. En segundo lugar, el Aumento de Datos clásico aplicando transformaciones específicas de forma aleatoria para generar nuevos datos y lograr un balance de clases en los datos no se pudo aplicar en las imágenes de ultrasonido, esto debido a que las características que permiten determinar si un nódulo es benigno o maligno se verían totalmente alteradas si algunos de estos cambios fuese a aplicarse. Es por este motivo que se

optó por una nueva forma de Aumento de Datos a través del modelo DCGAN capaz de generar imágenes falsas a través de un proceso de aprendizaje tomando como base ruido aleatorio.

Para evaluar de forma correcta la capacidad de los modelos a ser entrenados, se observó las investigaciones previas y el uso de aquellas métricas de evaluación de rendimiento usadas. Se determinó que el Accuracy es la métrica principal que describe si el modelo evaluado tiene la capacidad de clasificar correctamente si un nódulo es benigno y maligno, por lo que se decidió por su uso en esta investigación. Además, para lograr evaluar la capacidad de un modelo en clasificar correctamente si un nódulo es maligno específicamente, se optó por las métricas de Recall y Precision, usadas en gran medida también por otros investigadores, esto debido a la mayor importancia que se da por determinar si un nódulo pertenece a esta clase.

Determinar las arquitecturas de Deep Learning mayormente usadas y de alto desempeño también fue de gran importancia para reducir la cantidad de pruebas y entrenamiento de distintos algoritmos. Para lograr esto, se extrajo aquellos modelos con altas métricas de otras investigaciones desarrolladas en el mismo tipo de problemas. Así se tuvo, en primera instancia, a la arquitectura de VGG16, ResNet50 y Vision Transformer Base 16. En segundo lugar, se seleccionó también nuevas arquitecturas híbridas (ViT + CNN) que demostraban poseer un alto rendimiento comparado con los demás modelos de Deep Learning. Con todas las arquitecturas ya seleccionadas, el entrenamiento de todos estos fue más enfocado y pudo lograrse los mejores resultados posibles.

A través de todo este proceso de entrenamiento y modelos de Deep Learning con distintas técnicas, y luego de un minucioso análisis de las métricas de rendimiento, se obtuvo finalmente un modelo capaz de predecir con una precisión de 77.20 % si un nódulo es benigno o maligno a través de las imágenes de ultrasonido de la glándula tiroidal.

En esta investigación se ha encontrado con el problema de la baja cantidad de conjuntos de datos de acceso libre y con características ideales que permitan un correcto desarrollo de modelos de Deep Learning capaz de ayudar al pre diagnóstico de nódulos en la tiroides. Es por ello que, para futuras investigaciones, se recomienda una propia recolección de datos en las pertinentes instituciones, de igual forma a cómo se realiza en la mayoría de los antecedentes presentados en esta investigación. Si se consigue obtener un nuevo conjunto de datos, se podría lograr entrenar el mejor modelo DCGAN que sea capaz de generar mejores imágenes falsas, aumentando así la cantidad final de datos que se pueden tener y mejorando así las capacidades de los modelos.

El proceso de clasificación de imágenes es solo una pequeña parte de lo que puede permitir hacer un Sistema de Diagnóstico Asistido. En futuras investigaciones, se podría incluir la clasificación junto con otras tareas como la segmentación de nódulos en imágenes y el análisis

del nivel de hormonas en sangre para mejorar la capacidad del sistema en determinar si un nódulo tiroideo es benigno o maligno, obteniendo así una potente herramienta capaz de ayudar en la toma de decisiones final de los médicos especialistas y así lograr una mejora en la calidad de vida de sus pacientes.

Referencias

Bicbic, J., Macatangay, T. E. G., Miranda, M., Ocina, M., & Santos, A. (2023). Automated Pavement Distress Detection and Classification Using Convolutional Neural Network with Mapping. *TENCON 2023 - 2023 IEEE Region 10 Conference*, 512-517. doi: 10.1109/tencon.2023.512.

Géron, A. (2022). *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow* (2ª ed.). O'Reilly Media.

Ha, J., Kim, D., & Kim, M. (2022). Assessing severity of road cracks using deep learning-based segmentation and detection. *The Journal of Supercomputing*, *78*(16), 17721-17735. doi: 10.1007/s11227-022-04560-x.

Haugen, B. R., Alexander, E. K., Bible, K. C., Doherty, G. M., Mandel, S. J., Nikiforov, Y. E., Pacini, F., Randolph, G. W., Sawka, A. M., Schlumberger, M., Schuff, K. G., Sherman, S. I., Sosa, J. A., Steward, D. L., Tuttle, R. M., & Wartofsky, L. (2016). 2015 American Thyroid Association Management Guidelines for Adult Patients with Thyroid Nodules and Differentiated Thyroid Cancer: The American Thyroid Association Guidelines Task Force on Thyroid Nodules and Differentiated Thyroid Cancer. *Thyroid*, *26*(1), 1-133. doi: 10.1089/thy.2015.0020.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. doi: 10.1109/cvpr.2016.90.

Hurbans, R. (2020). *Grokking Artificial Intelligence Algorithms*. Manning Publications.

Inkawhich, N. (2024). DCGAN Tutorial. Recuperado de https://pytorch.org/tutorials/beginner/dcgan_faces_tutorial.html.

Izco, F. (2018). Base de Datos Corporativa de Personas. Recuperado de https://bookdown.org/f_izco/BDC-POC/metricas.html.

JERBI, F., ABOUDI, N., & KHLIFA, N. (2023). Automatic classification of ultrasound thyroids images using vision transformers and generative adversarial networks. *Scientific African*, *20*. doi: 10.1016/j.sciaf.2023.e01679.

Monroy Malca, V. d. P. (2021). *Diseño de un sistema de visión computacional para el pre-diagnóstico de la enfermedad de Parkinson a partir de la escritura de una persona* [Tesis de grado]. Universidad ESAN. Recuperado de https://hdl.handle.net/20.500.12640/2262.

Organización Mundial de la Salud. (2022). Cancer Today. Recuperado de https://gco.iarc.who.int/today/en/dataviz/maps-heatmap?mode=population&key=crude_rate&types=0&age_start=0&cancers=32&sexes=2.

Ren, M., Zhang, X., Zhi, X., Wei, Y., & Feng, Z. (2024). An annotated street view image dataset for automated road damage detection. *Scientific Data*, *11*(1), 407. doi: 10.1038/s41597-024-03263-7.

Roman-Garay, M. A., Morales-Rosales, L. A., Rodrıguez-Rangel, H., & Fernández-Gregorio, S. I. (2023). Detección de deterioros superficiales en pavimentos flexibles basada en segmentación semántica y redes transformer [Received 2023-04-17; accepted 2023-05-01. Page 276]. *Research in Computing Science*, *152*(6), 273-286.

Sampieri, R. H., Collado, C. F., & Lucio, P. B. (2014). *Metodología de la investigación* (6ª ed.). McGraw-Hill Education. Recuperado de https://books.google.com.pe/books?id=oLbjoQEACAAJ.

Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*. Recuperado de https://www.scopus.com/inward/record.uri?eid=2-s2.0-85083953063&partnerID=40&md5=a5b2d6b3fc9f0a6f92864467079a1280.

Anexos

A Árbol de Problemas

B Árbol de Objetivos

C Matriz de Consistencia

Título de la tesis	Diseño de un modelo de Deep Learning para el pre-diagnóstico de nódulos tiroideos a través de imágenes de ultrasonido			
Problema General	Objetivo General	Hipótesis General	Variables	Método
¿Cómo se podría detectar y gestionar de manera automatizada y eficiente el desgaste y deterioro de los pavimentos en Lima Metropolitana para reducir los costos de mantenimiento y mejorar la seguridad vial?	Desarrollar un modelo de clasificación automatizada utilizando redes neuronales convolucionales (CNN) para detectar y clasificar el deterioro de los pavimentos en Lima Metropolitana, optimizando el mantenimiento y mejorando la seguridad vial.	A través del diseño de un modelo de Deep Learning se pre-diagnóstica los nódulos tiroideos a través de imágenes de ultrasonido.	Dependiente: Pre-diagnóstico de nódulos tiroideos. Independiente: Modelo de Deep Learning.	
Problemas Específicos	Objetivos Específicos	Hipótesis Específicas	Variables	

¿Cómo podría implementarse un sistema automatizado para monitorear y clasificar en tiempo real el deterioro de los pavimentos?	Implementar un sistema de monitoreo y clasificación basado en IA y visión por computadora para la identificación de grietas y baches en tiempo real.	Las mejores características del conjunto de datos usadas para el proceso de entrenamiento y prueba permiten un mejor desarrollo del modelo de Deep Learning.	Dependiente: Desarrollo del modelo de Deep Learning. Independiente: Las características del conjunto de datos.	Tipo de Investigación: Experimental. Alcance de la investigación: Explicativo.
¿De qué manera podría reducirse el costo económico derivado de la rehabilitación tardía de carreteras en mal estado?	Analizar el impacto económico de la detección temprana de daños en la infraestructura vial de Lima.	Las técnicas de preprocesamiento a aplicar a las imágenes de ultrasonido del conjunto de datos mejora el desempeño del modelo de Deep Learning.	Dependiente: Desempeño del modelo de Deep Learning. Independiente: Técnicas de preprocesamiento.	

¿Cómo se podría minimizar el riesgo de accidentes de tránsito ocasionados por el mal estado de las vías urbanas?	Priorizar áreas críticas de deterioro para optimizar el uso de los recursos en las intervenciones de mantenimiento.	Las métricas de evaluación de rendimiento de los modelos de Deep Learning logra una correcta comparación de modelos en la tarea de clasificación de imágenes de ultrasonido.	Dependiente: Comparación de modelos en la tarea de clasificación de imágenes de ultrasonido. Independiente: Las métricas de evaluación de rendimiento.
¿Qué estrategias permitirían priorizar la reparación de las áreas más deterioradas para optimizar la distribución de los recursos?	Evaluar la precisión y eficiencia del modelo propuesto frente a métodos tradicionales de inspección vial.	Las arquitecturas de Deep Learning con más alto desempeño en la clasificación de imágenes de ultrasonido demuestran superioridad frente a los demás modelos.	Dependiente: Desempeño en la clasificación de imágenes de ultrasonido. Independiente: Las arquitecturas de Deep Learning.