



UNIVERSIDAD ESAN
FACULTAD DE INGENIERÍA
INGENIERÍA DE TECNOLOGÍAS DE INFORMACIÓN Y SISTEMAS

Diseño de un modelo de Deep Learning para la clasificación de nódulos tiroideos a través de imágenes de ultrasonido

Tesis de investigación para el curso Trabajo de Tesis II

Antonny Fernando Corcino Castillo
Asesor: Marks Calderón

Lima, 21 de abril de 2024

Esta tesis denominada:

**DISEÑO DE UN MODELO DE DEEP LEARNING PARA LA CLASIFICACIÓN DE
NÓDULOS TIROIDEOS A TRAVÉS DE IMÁGENES DE ULTRASONIDO**

ha sido aprobada.

.....

Jurado 1

.....

Jurado 2

.....

Jurado 3

Universidad ESAN

2024

DISEÑO DE UN MODELO DE DEEP LEARNING PARA LA CLASIFICACIÓN DE
NÓDULOS TIROIDEOS A TRAVÉS DE IMÁGENES DE ULTRASONIDO

Dedicatoria

A mi madre.

Gracias por tu apoyo todos estos años.

Índice general

Resumen	1
Introducción	3
Capítulo I: Planteamiento del Problema	5
1.1 Descripción de la Realidad Problemática	5
1.2 Formulación del Problema	9
1.2.1 Problema General	9
1.2.2 Problemas Específicos	9
1.3 Objetivos de la Investigación	9
1.3.1 Objetivo General	10
1.3.2 Objetivos Específicos	10
1.4 Justificación de la Investigación	10
1.4.1 Teórica	10
1.4.2 Práctica	10
1.4.3 Metodológica	11
1.5 Delimitación del Estudio	11
1.5.1 Espacial	11
1.5.2 Temporal	11
1.5.3 Conceptual	11
Capítulo II: Marco Teórico	12
2.1 Antecedentes de la investigación	12
2.2 Bases Teóricas	45
2.2.1 Deep Learning	45
2.2.2 Nódulos Tiroideos	52
2.3 Marco Conceptual	54
2.3.1 Inteligencia Artificial	54
2.3.2 Perceptrón	54
2.3.3 Perceptrón Multicapa	55
2.3.4 Machine Learning	56
2.3.5 Ecografía y las imágenes de ultrasonido	57
2.4 Hipótesis	57
2.4.1 Hipótesis General	57
2.4.2 Hipótesis Específicas	57

Capítulo III: Metodología de la Investigación	59
3.1 Diseño de la investigación	59
3.1.1 Tipo de la investigación	59
3.1.2 Enfoque de la investigación	59
3.1.3 Población	59
3.1.4 Muestra	60
3.2 Metodología de Implementación de la Solución	60
3.3 Metodología para la Medición de Resultados de la Implementación	61
3.4 Cronograma de actividades y presupuesto	64
Capítulo IV: Desarrollo del Experimento	66
4.1 Comprensión del negocio	66
4.2 Comprensión de los datos	67
4.3 Preparación de los datos	84
Capítulo V: Análisis y Discusión de Resultados	90
5.1 Modelamiento	90
5.2 Evaluación	97
5.3 Despliegue	107
Capítulo VI: Conclusiones y Recomendaciones	110
6.1 Conclusiones	110
6.2 Recomendaciones	112
Referencias	114
Anexos	120
A Árbol de Problemas	121
B Árbol de Objetivos	122
C Matriz de Consistencia	123
D Comparación de metodologías de antecedentes	126
E Comparación de objetivos específicos de antecedentes	130
F Parámetros para modelo predictivo de Metainformación	137
G Parámetros para modelo predictivo de Descripción	138
H Parámetros para modelo predictivo de Comentarios	139
I Parámetros para modelo de Aprendizaje Profundo Multimodal	140

Índice de Figuras

Figura 1.	Tasa bruta de incidencia de cáncer de tiroides en mujeres por 100 000 personas	5
Figura 2.	Tasa bruta de mortalidad de cáncer de tiroides en mujeres por 100 000 personas	6
Figura 3.	Tasa bruta de incidencia de cáncer de tiroides en varones por 100 000 personas	6
Figura 4.	Tasa bruta de mortalidad de cáncer de tiroides en varones por 100 000 personas	7
Figura 5.	Tasa bruta de incidencia y mortalidad de cáncer de tiroides por género y región	8
Figura 6.	Representación del método desarrollado	16
Figura 7.	Arquitectura de modelo TC-ViT	17
Figura 8.	Arquitectura de la red desarrollada	19
Figura 9.	Comparación de resultados de los modelos (precision y sensitivity)	21
Figura 10.	Comparación de resultados de los modelos (f1-score y specificity)	22
Figura 11.	Comparación de resultados según accuracy con el conjunto de datos HAM10000	23
Figura 12.	Comparación de resultados según accuracy con el conjunto de datos DER-MOFIT	24
Figura 13.	Balanceo de clases usando aumento de datos	26
Figura 14.	Estructura de ViT implementado	27
Figura 15.	Resultado de los modelos ViT basados en transfer learning en el conjunto de datos DDSM	28
Figura 16.	Resultado de los modelos ViT entrenados desde cero en el conjunto de datos DDSM	29
Figura 17.	Resultado de los modelos CNN basados en transfer learning en el conjunto de datos DDSM	29
Figura 18.	Comparación de resultados de los modelos MedViT con los CNN y AutoML	32
Figura 19.	Resumen de características de los conjuntos de datos	34
Figura 20.	Resultados con el conjunto de datos Chest X-ray	35
Figura 21.	Resultados con el conjunto de datos Kvasir	36
Figura 22.	Resultados con el conjunto de datos Kvasir-Capsule	36
Figura 23.	Resultados de los modelos entrenados con el conjunto de datos 2D (MCC, accuracy y f1-score)	39
Figura 24.	Resultados de los modelos entrenados con el conjunto de datos 2D (AUC, precision y recall)	40

Figura 25.	Resultados de los modelos entrenados con el conjunto de datos 3D (MCC, accuracy y f1-score)	41
Figura 26.	Resultados de los modelos entrenados con el conjunto de datos 3D (AUC, precision y recall)	41
Figura 27.	Imágenes de ultrasonido de la tiroides (originales y generadas)	43
Figura 28.	Comparación de los modelos entrenados	45
Figura 29.	Convolución de imagen de Fashion MNIST	46
Figura 30.	Convolución de imagen con filtro de líneas verticales	47
Figura 31.	Convolución de imagen con filtro de líneas horizontales	47
Figura 32.	Ejemplo de max-pooling con un pool de 2x2	48
Figura 33.	DenseNet con 5 capas	50
Figura 34.	Arquitectura del modelo ViT	52
Figura 35.	Imagen tomográfica de glándula de tiroides	53
Figura 36.	Perceptrón	55
Figura 37.	Perceptrón multicapa	56
Figura 38.	Enfoque del Machine Learning	57
Figura 39.	Metodología de implementación	60
Figura 40.	Matriz de Confusión	62
Figura 41.	Cronograma de actividades	64
Figura 42.	Presupuesto	65
Figura 43.	Vista del website Web Robots (visitado en agosto del 2019)	67
Figura 44.	Tamaño de conjunto de datos al corte de Julio 2019	68
Figura 45.	Visualización del archivo de metainformación subido a Kaggle	69
Figura 46.	Función para extraer textos de modalidad de descripción	70
Figura 47.	Ejecución de la función de extracción de descripciones y almacenamiento	71
Figura 48.	Visualización del archivo de descripción subido a Kaggle	71
Figura 49.	Función para extraer textos de modalidad de comentarios	72
Figura 50.	Ejecución de la función de extracción de comentarios y almacenamiento	73
Figura 51.	Instancias lanzadas en paralelo para la extracción de comentarios	73
Figura 52.	Visualización del archivo de comentarios subido a Kaggle	74
Figura 53.	Distribución de proyectos tecnológicos según su estado	75
Figura 54.	Evolución de cantidad de proyectos tecnológicos por año	75
Figura 55.	Evolución de proyectos tecnológicos, por su estado y año	76
Figura 56.	Distribución de las variables categóricas de Metainformación	76
Figura 57.	Diagrama de caja y bigote de patrocinadores	77
Figura 58.	Diagrama de caja y bigote de meta	78
Figura 59.	Diagrama de caja y bigote de monto patrocinado	78

Figura 60.	Diagrama de caja y bigote de duración	79
Figura 61.	Matriz de correlaciones entre variables independientes	79
Figura 62.	Gráfico de dispersión de correlaciones entre variables independientes	80
Figura 63.	Distribución de proyectos por presencia de descripciones y estado final	81
Figura 64.	Nube de palabras de descripciones	81
Figura 65.	Distribución de proyectos por presencia de comentarios y estado final	82
Figura 66.	Distribución de comentarios en proyectos exitosos y fracasados	83
Figura 67.	Nube de palabras de comentarios más frecuentes	83
Figura 68.	Matriz de correlaciones entre variables independientes considerando adicionales	84
Figura 69.	Función para dividir base de datos en subconjuntos de entrenamiento y prueba	85
Figura 70.	Función para normalizar variables	86
Figura 71.	Nube de palabras de descripciones posterior a la limpieza de texto	87
Figura 72.	Proceso de representación de palabras en vectores codificados	87
Figura 73.	Proceso de creación de matriz de incrustaciones de palabras	88
Figura 74.	Nube de palabras de comentarios posterior a la limpieza de texto	89
Figura 75.	Arquitectura de modelo MLP para la metadata	91
Figura 76.	Arquitectura de modelo CNN para las descripciones	92
Figura 77.	Arquitectura de modelo RNN para los comentarios	94
Figura 78.	Arquitectura del modelo apilado final The Hydra	96
Figura 79.	Exactitud y pérdida respectivamente de los subconjuntos de entrenamiento y validación para el modelo MLP de metadata con 100 épocas	98
Figura 80.	Matriz de confusión para el modelo de metadata	99
Figura 81.	Área bajo la curva ROC de modelo de metadata	100
Figura 82.	Exactitud y pérdida respectivamente de los subconjuntos de entrenamiento y validación para el modelo CNN de descripciones con 100 épocas	100
Figura 83.	Matriz de confusión para el modelo de descripciones	101
Figura 84.	Área bajo la curva ROC de modelo de descripciones	102
Figura 85.	Exactitud y pérdida respectivamente de los subconjuntos de entrenamiento y validación para el modelo RNN de comentarios con 50 épocas	102
Figura 86.	Matriz de confusión para el modelo de comentarios	103
Figura 87.	Área bajo la curva de modelo de comentarios	104
Figura 88.	Exactitud y pérdida respectivamente de los subconjuntos de entrenamiento y validación para el modelo apilado con 200 épocas	104
Figura 89.	Matriz de confusión para el modelo apilado	105
Figura 90.	Área bajo la curva de modelo apilado	106

Figura 91.	Proyecto consultado para la demostración. Captura de pantalla: 15/02/21	108
Figura 92.	Campaña del proyecto consultado. Captura de pantalla: 15/02/21	108
Figura 93.	Variables extraídas por modalidad del proyecto consultado	109
Figura 94.	Resultado de predicción de The Hydra para el proyecto consultado	109

Índice de Tablas

Tabla 1.	Diccionario de datos del dataset final de Metainformación	69
Tabla 2.	Potenciales combinatorias de variables de metainformación	85
Tabla 3.	Diccionario de datos del conjunto final entrenado	95
Tabla 4.	Exactitud de los conjuntos de datos de validación para las 8 combinaciones .	98
Tabla 5.	Informe de clasificación para el modelo de metadata	99
Tabla 6.	Informe de clasificación para el modelo de descripciones	101
Tabla 7.	Informe de clasificación para el modelo de comentarios	103
Tabla 8.	Informe de clasificación para el modelo apilado	105
Tabla 9.	Comparación de resultados de modelos propuestos con antecedentes	106

Índice de Ecuaciones

Ecuación 1	Fórmula para calcular el accuracy	62
Ecuación 2	Fórmula para calcular el recall	63
Ecuación 3	Fórmula para calcular el precision	63
Ecuación 4	Fórmula del escalador Mínimo Máximo	86

Resumen

Con los casos de cáncer de tiroides incrementando cada año en todo el mundo, los especialistas en el área y diversos investigadores se han visto en la necesidad de encontrar un método probado o herramienta eficaz que ayude a los médicos y radiólogos, con distintos niveles de experiencia, a realizar detecciones y diagnósticos de nódulos tiroideos de forma más rápida y con menos errores, incluso con la capacidad de llegar a ser una opción factible en el contexto de las clínicas u hospitales de bajos recursos y poco personal especializado para realizar un diagnóstico temprano de este mal y así evitar futuras complicaciones de cáncer avanzado, ya que existe un porcentaje nada despreciable que un nódulo llegue a convertirse con el paso del tiempo, y sin tratamiento, en un problema mucho más grave difícil de sobrellevar.

Por estos motivos, y tomando en cuenta el grado de desarrollo actual del extenso campo de la inteligencia artificial y junto con el fácil acceso al gran poder computacional, en la presente investigación se plantea desarrollar un modelo de Deep Learning que sirva como herramienta de soporte en el diagnóstico de nódulos tiroideos para mejorar la eficacia y eficiencia del diagnóstico a través de imágenes de ultrasonido.

Palabras claves: Aprendizaje profundo, Redes Neuronales Convolucionales, imágenes, ultrasonido, tiroides, clasificación.

Abstract

With thyroid cancer cases increasing every year around the world, specialists and researchers have found it necessary to find a proven method or effective tool to help physicians and radiologists, with different levels of experience, to detect and diagnose thyroid nodules more quickly and with fewer errors. Even with the ability to become a feasible option in the context of clinics or hospitals with low resources and few specialized personnel to make an early diagnosis of this disease and thus avoid future complications of advanced cancer, since there is a not insignificant percentage that a nodule becomes malignant with the passage of time, and without treatment, a much more serious problem difficult to cope with.

For these reasons, and taking into account the current degree of development of the extensive field of artificial intelligence and the easy access to great computational power, this research proposes to develop a Deep Learning model that serves as a support tool in the diagnosis of thyroid nodules to improve the effectiveness and efficiency of diagnosis through ultrasound images.

Keywords: Deep Learning, Convolutional Neural Networks (CNN), images, ultrasound, thyroid, classification.

Introducción

La presente investigación tiene como principal objetivo el desarrollo y propuesta de un modelo de inteligencia para diagnosticar de manera correcta y rápida la naturaleza de un nódulo en la glándula de la tiroides, esto a través de la clasificación de este en benigno o maligno. Se plantea utilizar recursos y algoritmos que brinda el extenso campo de estudio de la inteligencia artificial. En particular, se usarán los extensamente desarrollados algoritmos de Deep Learning como las redes neuronales convolucionales para lograr encontrar patrones en imágenes de ultrasonido de la glándula de tiroides, y así lograr correctos diagnósticos en nuevos casos que se presenten al modelo. Las imágenes de ultrasonido de la glándula de tiroides que se usará para el entrenamiento y prueba de los distintos modelos a entrenar procederán de repositorio en línea debidamente validado.

El proceso que se seguirá para desarrollar un modelo eficaz, capaz de reducir el tiempo que le toma a un médico o radiólogo en detectar y diagnosticar un nódulo tiroideo consiste en una parte inicial de análisis y filtrado de las imágenes, esto con el objetivo de obtener un buen conjunto de datos para el siguiente paso que es el modelado. Se usarán distintos modelos y técnicas dentro del área de la inteligencia artificial y el Deep Learning. Finalmente, se usarán métricas para evaluar cada uno de estos modelos, con el objetivo final de determinar aquel de mejor desempeño. Esto se mostrará a más detalle en la descripción de la metodología de la implementación.

Esta herramienta de inteligencia artificial no pretende ser un sustituto al diagnóstico de un profesional especializado en este tipo de casos. Como se verá más adelante, en las investigaciones presentadas como antecedentes que desarrollan, de la misma forma que la presente investigación, modelos capaces de clasificar nódulos en la glándula de tiroides a través de imágenes, el objetivo final es brindar una herramienta eficaz y eficiente que permita a los profesionales del sector a realizar diagnósticos con menos probabilidad de errores. Esto es importante debido a la naturaleza de desarrollo de esta enfermedad, pues con una detección temprana y correcta del tipo de nódulo con el que el especialista se enfrenta, se puede acelerar tratamientos eficaces y evitar aquellos que son totalmente innecesarios para el paciente.

El problema a abordar, y principal incentivo de la investigación, radica en la limitada precisión que, de manera general, los médicos o radiólogos han ido desarrollando en el diagnóstico de nódulos toroidales, además de las escasas herramientas basadas en inteligencia artificial en este campo. También observar que los casos de cáncer en esta glándula van en aumento en el Perú y el mundo, genera una preocupación y alta necesidad de desarrollar nuevos y mejores métodos o herramientas junto con los grandes avances tecnológicos como la inteli-

gencia artificial que se ha demostrado, aunque de forma muy escasa, ser capaz de reducir los índices de errores en el diagnóstico o clasificación de un nódulo benigno o maligno.

Capítulo I: Planteamiento del Problema

1.1 Descripción de la Realidad Problemática

A nivel mundial, las tasas de presencia de cáncer de tiroides en personas varían de acuerdo con la edad, al género y al país. En el caso de Perú se tiene que por cada 100 000 mujeres existen 10.9 de casos de incidencia en este tipo de cáncer, y por cada 100 000 varones, se tiene un 3.6 de casos. Además de los casos de incidencia, la mortalidad también está presente y varía en varias partes del mundo. En Perú la mortalidad de pacientes mujeres con cáncer de tiroides es 1.7 por cada 100 000 personas, mientras que en los varones es de 0.77 casos cada 100 000 personas. (Organización Mundial de la Salud, 2022)

A continuación, se presentan dos gráficos 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.

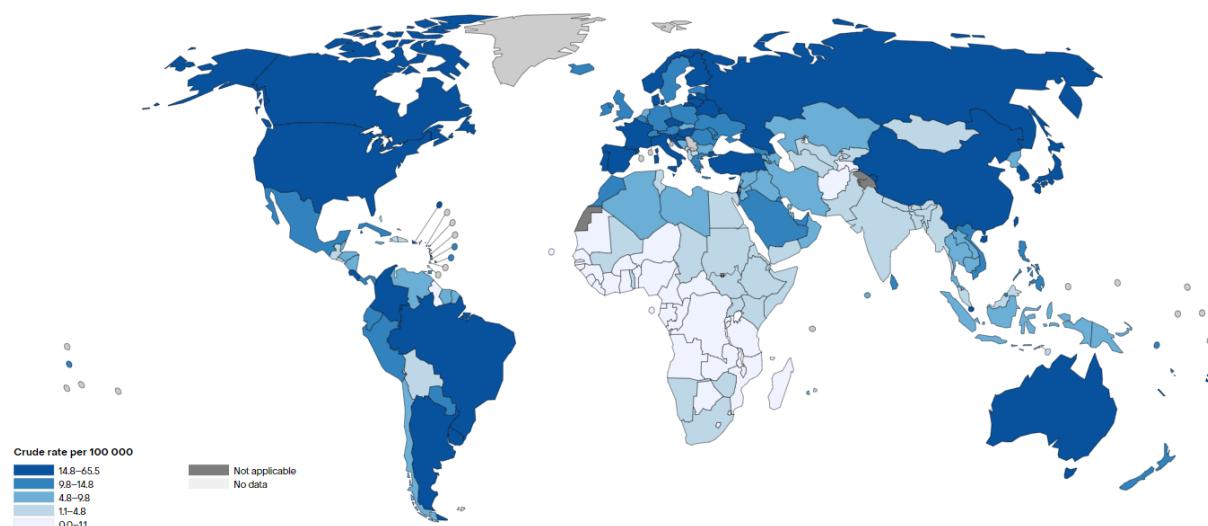


Figura 1. Tasa bruta de incidencia de cáncer de tiroides en mujeres por 100 000 personas.

Fuente: Organización Mundial de la Salud (2022). *Cancer Today*.

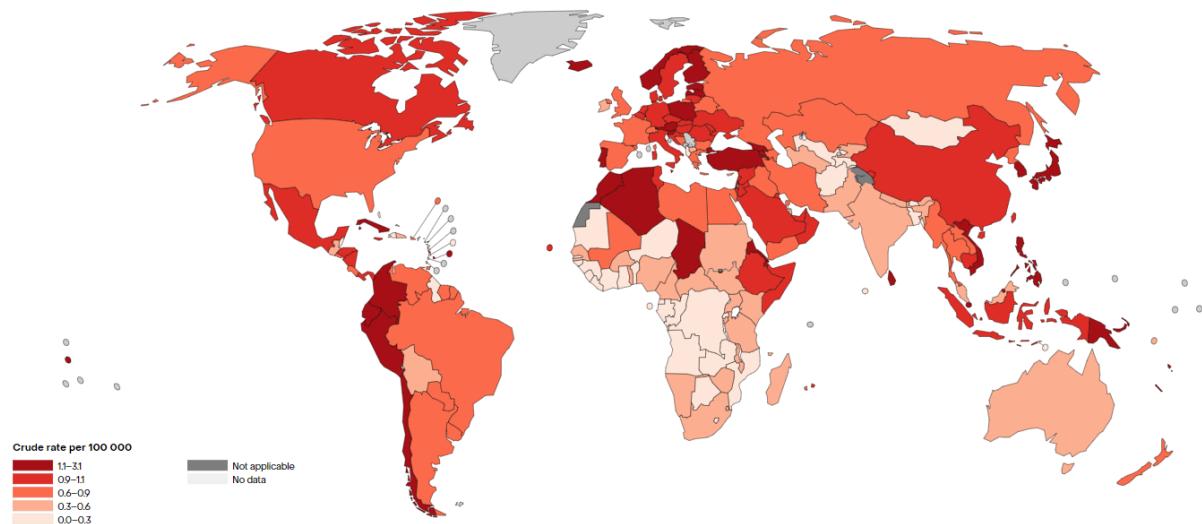


Figura 2. Tasa bruta de mortalidad de cáncer de tiroides en mujeres por 100 000 personas.

Fuente: Organización Mundial de la Salud (2022). *Cancer Today*.

De igual forma, en el caso de los varones, el Perú también se encuentra entre los rangos más altos de incidencia y mortalidad. A continuación, se muestran sus respectivas gráficas.

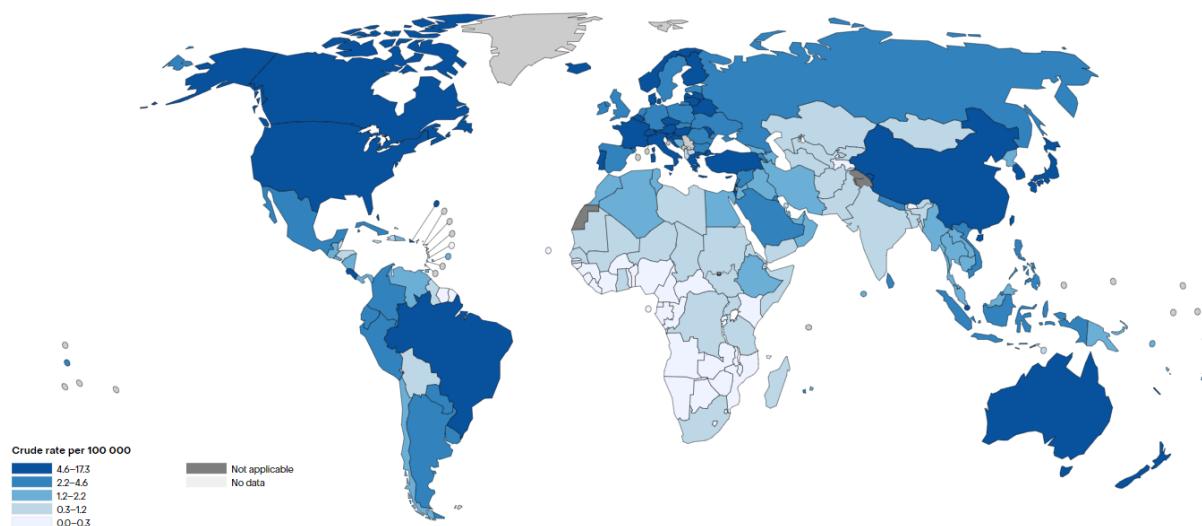


Figura 3. Tasa bruta de incidencia de cáncer de tiroides en varones por 100 000 personas.

Fuente: Organización Mundial de la Salud (2022). *Cancer Today*.

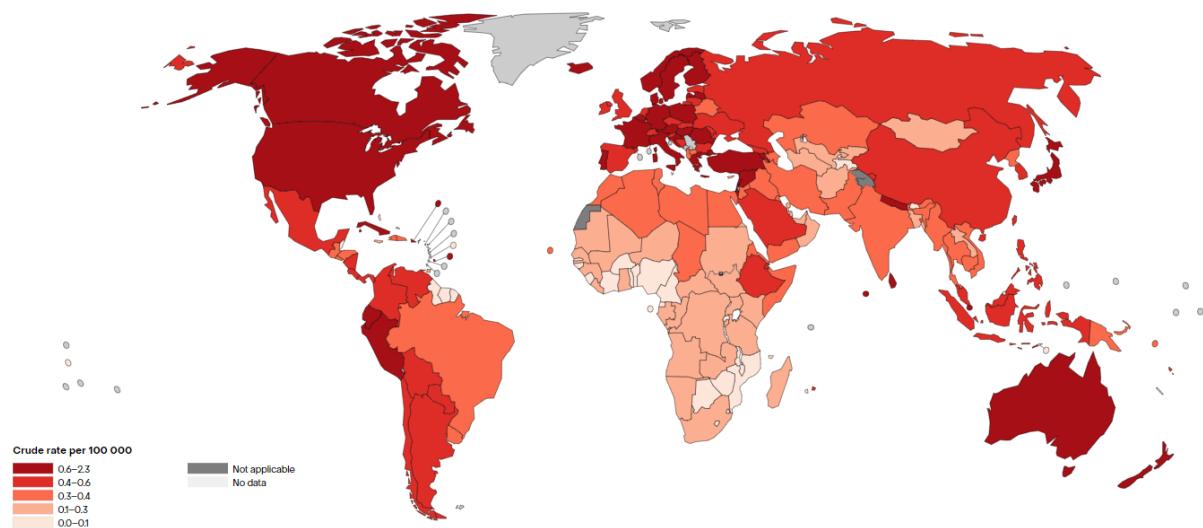


Figura 4. Tasa bruta de mortalidad de cáncer de tiroides en varones por 100 000 personas.

Fuente: Organización Mundial de la Salud (2022). *Cancer Today*.

Con el siguiente gráfico que muestra los mismos índices distribuidos por género y regiones del mundo, es fácil notar la alta incidencia de este tipo de cáncer en las mujeres, siendo la región con mayor incidencia América del Norte, mientras que la de mayor mortalidad es Oceanía. La región de Latino América y el Caribe supera a Norte América en mortalidad, aunque se encuentra por debajo de Oceanía y Europa.

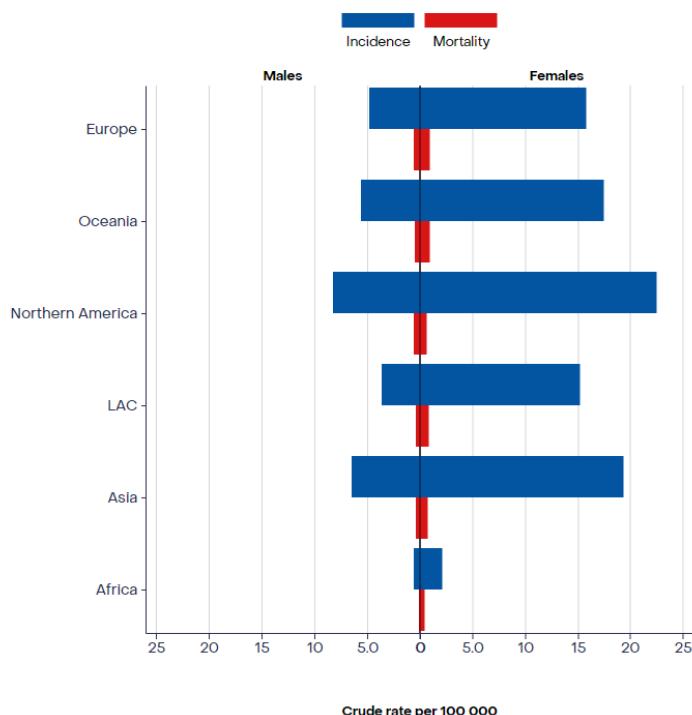


Figura 5. Tasa bruta de incidencia y mortalidad de cáncer de tiroides por género y región.

Fuente: Organización Mundial de la Salud (2022). *Cancer Today*.

Además, es importante mencionar que este aumento de incidencia a nivel global de cáncer de tiroides se debe a diversos factores relacionados a problemas de salud como la obesidad y a factores medioambientales como por ejemplo exposición al yodo (Kim et al., 2020). Sin embargo, otro factor es la poca disposición y capacidad de realizar un diagnóstico a tiempo de los nódulos tiroideos que, de forma general, cerca del 7 % al 15 % de los casos de llegan a ser cancerígenos (Haugen et al., 2016).

Para la detección temprana de este tipo de cáncer o desarrollo de tumores, depende en gran medida de la experiencia y la capacidad cognitiva de un experto en radiología, y muchos de estos se ven en la necesidad de utilizar no muy avanzados sistemas de diagnóstico por computadora, mejor conocido como CAD por sus siglas en inglés (Zhu et al., 2021). Ante grandes limitaciones de sistemas CAD básicos, y aprovechando el extenso uso de la inteligencia artificial, el Deep Learning y sus algoritmos son capaces de incorporar mayor eficacia a dichos sistemas.

El Deep Learning ha sido usado ampliamente como herramienta para el procesamiento de imágenes médicas, no solo para detectar diferentes tipos de cáncer o nódulos como los relacionados a los pulmones, sino también para la retinopatía diabética y localización de feto en ecografías, e inclusive en la detección del COVID-19 (Bhattacharya et al., 2021). Además,

el aumento de la calidad de imágenes de ultrasonido que se fue desarrollando en los últimos 30 años, aumentando cada vez más resolución de las imágenes y el tiempo de adquisición, ha permitido una mejora en términos de detección de enfermedad; sin embargo, aún se requiere de un médico especializado y debidamente entrenado para lograr un realizar un correcto análisis de las imágenes, por ello se vio en la necesidad de encontrar métodos de clasificación automatizada, pero dichos métodos antiguos consumían bastante tiempo, gran poder computacional y poca capacidad de generalización de resultados, es así que en este contexto, apareció una novedosa arquitectura de Deep Learning que actualmente es usado en diferentes área el día de hoy: las Redes Neuronales Convolucionales o CNN, quitando en gran medida los problemas de las antigua técnicas (Singh et al., 2020). Aunque existe varios antecedentes con esta técnica de Deep Learning en detección de nódulos en la tiroides, muy pocos se han centrado en el grado de ayuda que se brinda a un médico especialista, ya que una simple detección no aceleraría debidamente el proceso de descarte de una anomalía en la tiroides.

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: ¿Es posible implementar un modelo de Deep Learning para el diagnóstico de nódulos tiroideos a través de imágenes de ultrasonido?

1.2.2 Problemas Específicos

- PE1: ¿Qué características del conjunto de datos a usar para entrenar y evaluar los modelos son ideales para obtener buenos resultados?
- PE2: ¿Qué técnicas de preprocesamiento se deberían aplicar a las imágenes de ultrasonido del conjunto de datos?
- PE3: ¿De qué forma se puede evaluar el rendimiento de los modelos de Deep Learning en la clasificación de imágenes de ultrasonido?
- PE4: ¿Qué arquitecturas de Deep Learning tienen el más alto desempeño en la clasificación de imágenes de ultrasonido de la tiroides?

1.3 Objetivos de la Investigación

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

1.3.1 Objetivo General

OG: Diseñar un modelo de Deep Learning para el diagnóstico de nódulos tiroideos a través de imágenes de ultrasonido.

1.3.2 Objetivos Específicos

- OE1: Determinar las características ideales del conjunto de datos a usar para entrenar y evaluar los modelos.
- OE2: Determinar las técnicas de preprocessamiento que se deben aplicar a las imágenes de ultrasonido del conjunto de datos.
- OE3: Identificar las métricas de evaluación de rendimiento de los modelos de Deep Learning en la clasificación de imágenes de ultrasonido.
- OE4: Determinar las arquitecturas de Deep Learning que tienen el más alto desempeño en la clasificación de imágenes de ultrasonido de la tiroides.

1.4 Justificación de la Investigación

1.4.1 Teórica

La investigación se desarrolla con el propósito de otorgar mayor conocimiento sobre el uso de la inteligencia artificial en el diagnóstico de nódulos en la glándula de la tiroides, esto a través de imágenes de ultrasonido. Esta investigación aportará conocimiento sobre el diseño y eficacia de los sistemas de diagnóstico asistido por computadora con base en el Deep Learning en el diagnóstico de nódulos tiroideos. Esto servirá para que en un futuro se generen nuevos y más eficaces sistemas de apoyo médico dentro de esta área.

1.4.2 Práctica

Los antecedentes presentados en la investigación califican la capacidad de un modelo de Deep Learning en la detección y diagnóstico de una imagen de ultrasonido de la tiroides, esto usando diferentes algoritmos y técnicas con el objetivo de aumentar la capacidad del modelo. Muchos de estos abarcan la parte de clasificar una imagen, mientras que otros realizan segmentación del área donde el nódulo se está desarrollando, basándose además en estándares médicos de diagnóstico. Muy pocos realizan ambos: clasificación y segmentación. La presente investigación se centrará en desarrollar un modelo de clasificación.

1.4.3 Metodológica

El modelo desarrollado en la investigación no propone un reemplazo de los médicos especialistas en el área, sino una herramienta potente para estos que ayudará a diagnosticar los nódulos que se desarrollan en la glándula de tiroides. La investigación presentará y concluirá con las herramientas tecnológicas y técnicas de Deep Learning ideales para el desarrollo de sistemas de diagnóstico asistido por computadora, esto luego de una extensa recopilación de datos y métodos que permitan la construcción y desarrollo de un modelo eficiente y eficaz.

1.5 Delimitación del Estudio

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

1.5.1 Espacial

Debido a que la problemática y necesidad de diagnosticar a tiempo los nódulos tiroideos no es propio de una región o país, la mayoría de la data para realizar un correcto análisis de inteligencia artificial son basados en países extranjeros y de gran alcance como Estados Unidos, India y China. Los proyectos previos más afines al tema presentados en la investigación son desarrollados en diversos países extranjeros.

1.5.2 Temporal

Los datos presentados en esta investigación sobre casos de incidencia y mortalidad de la tiroides son hasta el 2022. De igual forma, las imágenes de ultrasonido de glándulas de tiroides con presencia de nódulos que se usarán para entrenar y validar el modelo de clasificación son recopiladas del conjunto de datos de acceso libre TNCD recolectada hasta finales del año 2022. Los antecedentes relacionados a esta investigación fueron publicados entre el 2019 y 2023.

1.5.3 Conceptual

La presente investigación se centrará en el desarrollo de un modelo de Deep Learning para realizar la clasificación de nódulos de la glándula de tiroides y así lograr determinar si es de carácter benigno o maligno.

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 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.

Moreira Aresta (2021) presenta una investigación relacionada a la elaboración de un sistema CAD (Computer-aided Detection) para la detección y segmentación de nódulos presentes en los pulmones.

Para realizar esta investigación, se tuvo en consideración la realidad que se atraviesa a nivel mundial sobre el cáncer de pulmón, y como este afecta a las personas sin importar su género, convirtiéndolo uno de los tipos de cáncer más mortales según la investigación mencionada.

La presencia de nuevas formas de detectarlo como el uso de tomografías computarizadas tuvo un gran impacto para la detección temprana de los nódulos en estos órganos, ya que estos pueden significar un futuro desarrollo de cáncer. Sin embargo, realizar un análisis de este tipo de imágenes médicas no era nada trivial debido al complejo proceso de análisis con estas tecnologías. Ante esto, el Deep Learning apareció con una herramienta eficaz para ayudar a los médicos en el análisis de estas imágenes; sin embargo, pese a sus grandes ventajas, su aplicabilidad es muy limitada. Por tal motivo, esta tesis incluye modernos enfoques y técnicas para una detección automática de nódulos pulmonares.

De forma general, el algoritmo presentado en esta investigación consiste en dos bloques. El primero de estos se encarga de la detección de nódulo a través de técnicas de detección de objetos, mientras que el segundo se encarga de la segmentación de este. Los resultados obtenidos fueron del 79 % en recall; sin embargo, esto no fue suficiente para poder considerar al modelo como robusto, por ello, posteriormente, se utilizaron técnicas innovadoras en colaboración expertos en el área. Las anotaciones de los especialistas, junto con los resultados del sistema, mejoraron el desempeño general de la detección.

Para aumentar aún más las capacidades del sistema, se desarrolló en la parte final de la investigación un modelo de Deep Learning para la detección automática y la segmentación de nódulos pulmonares a través de imágenes tomográficas. Este también consistió en dos bloques, uno encargado de la segmentación automática y otro que corrige el anterior en base a dos puntos en los límites del nódulo. El modelo consiguió demostrar su capacidad para corregir la

segmentación de nódulos pequeños, además de segmentar los nódulos no sólidos, los cuales presentaban un reto para el sistema.

Felgueiras Carvalho (2019) reafirman la importancia de construir este tipo de sistemas mencionando al cáncer de pulmón como el principal tipo de cáncer en causar muertes en todo el mundo.

El sistema desarrollado en esta investigación fue basado también en el diagnóstico a través de imágenes tomográficas que comúnmente es realizado por un médico experto; sin embargo, se menciona que este análisis siempre está sujeto a errores ya que existe mucha subjetividad en este tipo de diagnóstico, lo que conlleva muchas veces errores de detección. Además de la existencia de la tendencia al rechazo hacia esto tipo de sistemas CADx (Computer-aided Diagnosis) por parte de los médicos, esto debido a la falta de comprensión del cómo se realiza este tipo de diagnóstico.

Para afrontar esta desconfianza a este tipo de sistemas, se menciona la existencia del Deep hierarchical semantic convolutional neural network (HSCNN) que añadía a la capacidad de predecir si un nódulo era maligno o benigno, la función de otorgar evidencia visual del diagnóstico a través de la predicción de las características del nódulo. Sin embargo, el conjunto de datos usado en esta investigación difería de las evaluaciones de los propios médicos. Por tal motivo, en la presente investigación se puso como un objetivo probar si la disminución de esta varianza en los datos podría mejorar los resultados del modelo HSCNN.

A través de un análisis de los datos, se logró mejorar la descripción de características. Este nuevo conjunto de datos fue revisado por especialista para comprobar su validez antes de ser ingresado al modelo HSCNN para predecir si un nódulo era maligno o benigno. Este proceso se hizo a través de un k-folds igual a 4 junto con cross-validation.

Los resultados fueron comparados con el modelo inicial, y se obtuvo que el nuevo HSCNN era mejor solo cuando se trataba de predecir si un nódulo era maligno. Las métricas del nuevo modelo fueron de 0.78 en accuracy, el AUC de la curva ROC es de 0.74, una sensibilidad 0.83 y una especificidad de 0.89, frente a las métricas de modelo original que obtuvo un accuracy de 0.84, un AUC de la curva ROC de 0.86, sensibilidad de 0.71 y una especificidad de 0.89. Esto significó que el modelo se equivocaba más cuando los nódulos a analizar eran pequeños. En general, los resultados distaron de los propios del modelo inicial, esto es debido a la reducción considerable de imágenes del conjunto de datos original.

Supanta Zapata (2021) menciona la incidencia de cáncer de pulmón en el Perú en el periodo 2010-2012 con un gran número de 3 121 casos diagnosticados, convirtiéndolo en el tercer tipo de cáncer más común en el país, y ocupa el segundo lugar en mortalidad con un

número de 2 591 de muertes en el mismo periodo, esto debido al tardío diagnóstico de este mal. Por este motivo, es obvia la necesidad de realizar un examen de detección temprana donde es posible un tratamiento y, posteriormente, una posible cura a esta enfermedad.

La investigación presenta una forma de realizar detección de nódulos pulmonares a través de imágenes radiográficas digitales que comúnmente presentan problemas como baja claridad y resaltado de las características, por ende, generan dificultades para realizar un correcto análisis de este tipo de imágenes. Las técnicas usadas son corrección gamma, análisis de proyecciones, erosión, filtros geométricos, dilatación, filtro de convergencia y la umbralización por el método Otsu, esto aplicado a un conjunto de datos de 50 radiografías de tórax.

Los resultados muestran una sensibilidad del 0.91, especificidad de 0.96 y precisión de 0.94.

Otra investigación relacionada a detectar alguna enfermedad a través de técnicas de Deep Learning y visión por computadora es la presentada por Monroy Malca (2021) donde toma a la enfermedad de Parkinson como mal a detectar, y esto a través del análisis de la escritura de una persona. Para tal objetivo, en la investigación se desarrolló un modelo de visión computacional para el prediagnóstico de esta enfermedad.

La metodología empieza con la adquisición y preprocesamiento de los datos, para posteriormente usar técnicas de extracción de características como SIFT, SURF, ORB y HOG. Estos serán ingresados a un modelo de Machine Learning (SVM, RF, KNN) que finalmente realizará una clasificación. Además, se usaron diferente arquitectura de redes neuronales convolucionales como Inception, VGG16, VGG19, LeNet y ResNet50.

Los resultados finales fueron un 0.99 en accuracy, 0.99 en precisión, 0.99 en recall, 0.98 en F1-score y AUC.

Kang et al. (2022) muestran la frecuencia de diagnóstico de nódulos tiroideos, esto en el rango de 19 % y 68 % de casos clínicos. El cáncer de tiroides es el número 9 en incidencia, mientras que es el número 6 en mortalidad, esto a nivel mundial según datos del 2018.

Se menciona que es importante tener métodos menos invasivos para determinar el cáncer de tiroides, teniendo en cuenta además que es necesario la detección a tiempo, pues esto aumenta la probabilidad de ser curado. El análisis de imágenes de ultrasonido es una buena opción a el análisis de, por ejemplo, cirugías. Sin embargo, este diagnóstico depende mucho de la experiencia del médico especialista, lo cual vuelve a este tipo de análisis muy subjetivo.

La segmentación y clasificación son herramientas claves dentro de un sistema de diagnóstico asistido, siendo ambos muy relacionados entre sí, pues comparten ciertas características de

las imágenes (bordes de las imágenes pueden ser usados para la clasificación y segmentación). Afrontar ambas tareas al mismo tiempo en un solo modelo podría ser más robusto.

La red MTL propuesta en uno de los antecedentes es similar a una red de segmentación multiclases que genera como salida mapa de segmentación y predicciones categóricas.

La investigación presentada en este antecedente se centra también en una red MTL que también realizará procesos de segmentación y clasificación.

La metodología empieza con la data. Esta fue recolectada del West China Hospital. Se tuvieron 4 493 imágenes de ultrasonido, cada uno de un paciente. En específico, 2 576 imágenes pertenecen a nódulos benignos, mientras que 1 917 son malignos. Las anotaciones fueron verificadas por especialistas.

Para la evaluación de la clasificación se usó el accuracy, f1-score, ROC, área bajo la curva ROC (AUC). En el caso de la segmentación, se usó Dice coefficient y Intersection of Union (IoU).

Además, se definieron 3 medidas para cuantificar la inconsistencia al nivel de las tareas. El primero de estos evalúa a la clasificación, el segundo por la tarea de segmentación de segunda clase, y el tercero es para la segmentación de tercera clase.

El modelo construido en esta investigación se basó en MSL (Multi Stage Learning) y MTL (Multi Task Learning). Se diseño una red MS-MTL. Además, se determinó diferentes tipos de consistencia de tareas: intra e inter task consistency.

Los resultados fueron extraídos de 3 modelos CIsNet, MS-MTL y MS-MTL con intra e inter task consistency. La medida AUC obtenida para cada modelo es 0.9277, 0.9523 y 0.9608, respectivamente. Además, se determinó que el intra task consistency y el MSL aumenta el desempeño de la segmentación. El caso de inter task consistency con MTL mejora el desempeño de las tareas de clasificación y segmentación. Con la combinación de ambas tareas de consistencias también se demuestra su efectividad.

El método usado en esta investigación se muestra a continuación.

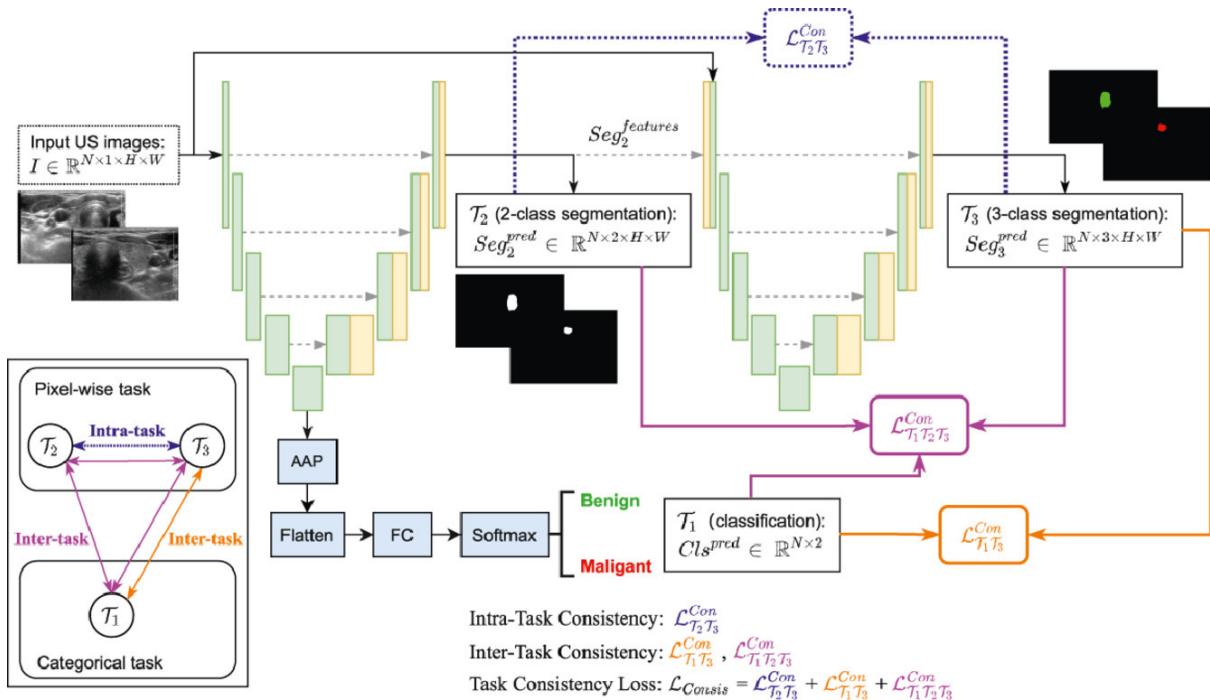


Figura 6. Representación del método desarrollado.

Fuente: Kang et al. (2022). *Thyroid nodule segmentation and classification in ultrasound images through intra- and inter-task consistent learning.*

Sun et al. (2023) presenta el problema de diagnosticar nódulos tiroideos nivel 3 debido a pocas características representativas que diferencien a los nódulos benignos de nivel 3 con los nódulos malignos, esto conlleva a obtener bajas precisiones en el diagnóstico. Ante esto, en el artículo se presenta un modelo clasificador de nódulos tiroideos con ViT (Vision-Transformer-based) y el contrast learning. Estas técnicas ayudaron a minimizar la distancia de características en nódulos de una misma clase, lo cual mejora la capacidad predictiva del modelo. Finalmente, en la fase de testeo del modelo se logró un accuracy de 0.869, mientras que las demás métricas indican la superioridad frente a otros modelos clásicos de Deep Learning que también son usados para clasificar.

En la siguiente figura se presenta la arquitectura de Vison Transformer desarrollada.

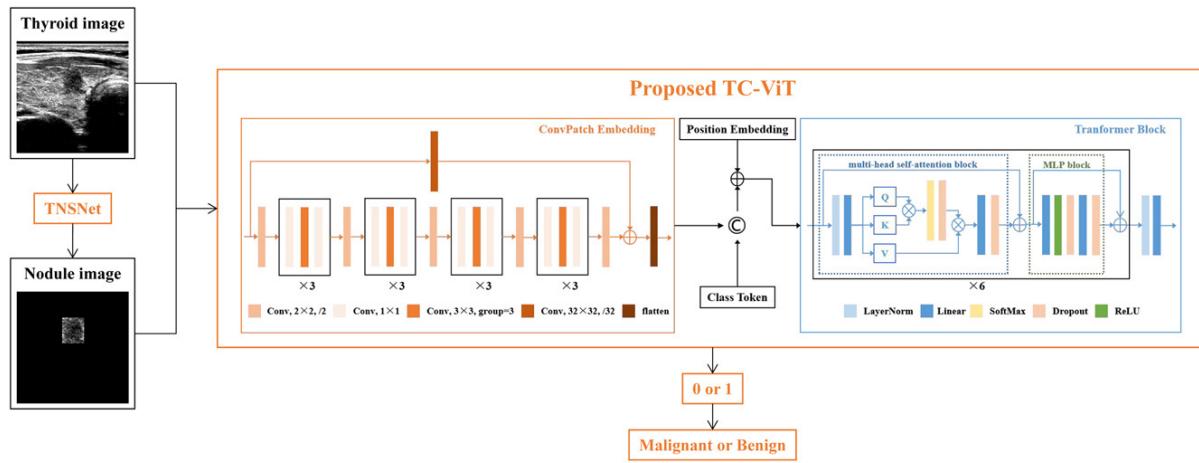


Figura 7. Arquitectura de modelo TC-ViT.

Fuente: Sun et al. (2023). *Classification for thyroid nodule using ViT with contrastive learning in ultrasound images.*

Zhang et al. (2023) presenta el nivel de importancia de poseer una buena base de datos etiquetada correctamente para lograr entrenar un buen modelo de Machine Learning. En este artículo se desarrolla y presenta a la herramienta Multistep Automated Data Labelling Procedure (MADLaP) que facilita y automatiza el proceso de etiquetado de datos relacionados a nódulos tiroideos. Este incluye el procesamiento de lenguaje natural basado en reglas, el Deep Learning para segmentación de imágenes y el reconocimiento óptico de caracteres. MADLaP fue desarrollado en la fase de entrenamiento con datos de 378 pacientes, mientras que en la fase de prueba se usaron datos de 93. Este obtuvo finalmente un accuracy de 0.83.

Deng et al. (2022) ponen al cáncer de tiroides como el que más ha prevalecido en las últimas 3 décadas. Existen diversos sistemas que ayudan a la detección de los nódulos en esta glándula; sin embargo, muchos de estos solo se limitan a determinar si un nódulo es maligno o benigno, y no muestran el porqué de la toma de esa decisión por parte del sistema, lo cual genera desconfianza entre los especialistas al momento de usarlos. Para afrontar esto, se desarrolla primeramente una estratificación de riesgo basada en el léxico estandarizado ACR TI-RADS. Posteriormente, se realiza la clasificación entre benigno y maligno. De forma general, el método realizará una caracterización del nódulo basado en ACR TI-RADS para detectar su nivel de riesgo y la clase al que pertenece (benigno o maligno). Los resultados muestran en la evaluación un accuracy de 0.9355, un sensitivity de 0.9386 y una specificity de 0.9314.

Se realizó la notación de las imágenes de nódulos con los indicadores del diccionario de ACR TI-RADS. Para afrontar el desbalanceo de la data, se realizó un proceso de mejora de la data creando imágenes a través de giros, recortes y mezclas. Se extrajeron las áreas de interés

de las imágenes a través de una red en cascada. Se quitaron las anotaciones manuales en las imágenes (limpieza de imagen). Una vez concluido con el procesamiento de las imágenes, se realizó la construcción y ejecución del modelo de Deep Learning Multi-Task Learning (MLT). Finalmente, el modelo obtenido fue comparado con otros a través de los siguientes indicadores: accuracy, sensitivity, specificity y el área bajo la curva.

Wang et al. (2020) mencionan que los nódulos tiroideos son uno de los primeros síntomas que podrían conllevar a un cáncer en la tiroides. Este tipo de cáncer es uno de los que tiene mayor incidencia y que esta tendencia ha ido en crecimiento durando los últimos 30 años.

Además, se menciona que, para ayudar a esta detección, se han propuesto anteriormente varios sistemas de diagnóstico asistido; sin embargo, estos solo realizan dicho proceso a través de solo una imagen de ultrasonido en vez de usar todas aquellas que se obtienen de un examen. Por esto, en este artículo, se desarrolla un modelo de Deep Learning para el diagnóstico de tiroides a través de varias imágenes de ultrasonido. Esto a través de una integración de todas las características de las imágenes realizadas en un examen.

La base de datos usada fue construida, y se obtuvieron resultados perfectamente comparables con los resultados de los antecedentes revisados en este artículo.

La metodología consistía en la construcción de tres redes distintas: feature extraction network, attention-based feature aggregation network, classification network. Estas tres redes juntas forman el modelo objetivo desarrollado en este artículo.

En general, la metodología consistía en, primero, la construcción del conjunto de datos que se conforma de imágenes de ultrasonido procedentes de un hospital. Se recolectaron cerca de 7 800 imágenes de 1 046 exámenes de entre los años 2015 y 2018. En segundo lugar, se realizó el etiquetado de los datos. Estas anotaciones se realizaron de acuerdo con los exámenes en general, y no a cada una de las imágenes que la conforman. Luego, se realizó la separación de la data en entrenamiento, prueba y validación. En la parte final de experimentación, se realizó un preprocesamiento con data augmentation y se definió 100 épocas para la fase de entrenamiento. El modelo ganador de la fase de validación fue probado con la data de prueba. Las métricas usadas para la evaluación fueron el accuracy, sensitivity (true positive rate) y el área bajo la curva (AUC ROC).

La siguiente figura muestra la arquitectura propuesta en esta investigación.

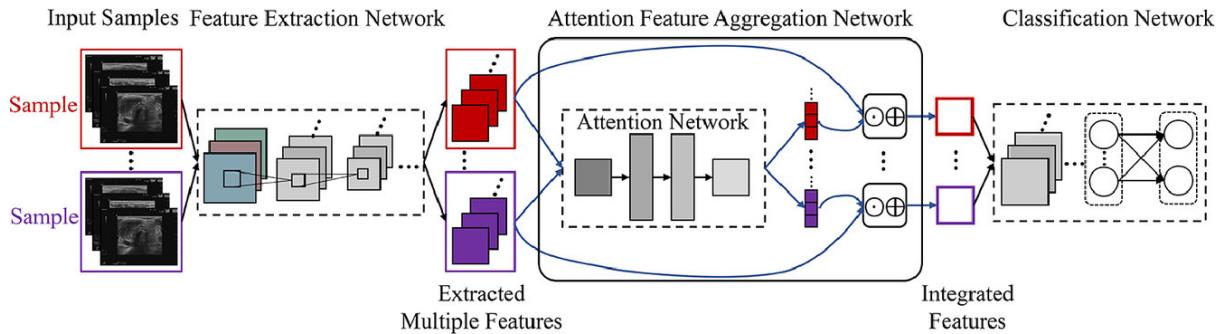


Figura 8. Arquitectura de la red desarrollada.

Fuente: Sun et al. (2023). *Automatic diagnosis for thyroid nodules in ultrasound images by deep neural networks.*

Sun et al. (2022) mencionan que, en el sistema endocrino, un problema muy común son los nódulos tiroideos. Normalmente estos pueden ser sólidos o de otras variadas texturas. La incidencia de esta enfermedad se ha ido incrementando a través de los años. Comúnmente se usan las imágenes de ultrasonido para detectar estos nódulos, esto lo hacen en tiempo real, tiene bajo costo y no es invasivo, por lo cual es una buena opción. Estas imágenes pueden otorgar información valiosa de los nódulos como los márgenes, ecogenicidad, calcificación, composición, etc. Además, existe el Thyroid Image Reporting y el sistema de datos TI-RADS pueden cuantificar las características otorgadas por las imágenes de ultrasonido y evaluar si es benigno o maligno. Sin embargo, estas imágenes pueden traer problemas como bajo contraste y ruido de moteado, lo cual genera retos para extraer sus características. Este problema puede superarse a través de una correcta segmentación de los nódulos tiroideos. Así, la segmentación es un paso importante para realizar diagnóstico correcto. Esto puede ser aplicado a la evaluación automática en TI-RADS para facilitar la clasificación de los nódulos. Se añadió un shape-supervised path para mejorar la identificación de la forma, y así lograr una mejor segmentación.

La data que usaron se obtuvo de diversos escáneres disponibles comercialmente, juntando 3 786 imágenes de ultrasonido de nódulos tiroideos. Posteriormente, se realiza un preprocesamiento para estandarizar la data. Se realizó un proceso de limpieza de las imágenes para quitar los datos privados de los pacientes, y borrar imágenes erróneas. Se propuso TNSNet para asegurar una mejor detección y segmentación. Este modelo es un dual-path network que contiene dos partes region path y shape path. Ambos caminos se enfocan en las características de bordes y texturas (cada uno de estos para cada camino).

Se diseñó un dual-path loss function para el entrenamiento de ambos caminos del modelo. Para el camino de región, se usó el binary cross-entropy loss y el generalized Dice loss, ambos

para entrenar la región de segmentación de los nódulos tiroideos. En el caso del segundo camino (camino de la forma) se combinó el Hausdorff distance loss y un modificado active contour loss. Esto fue construido para el entrenamiento del contorno de segmentación.

Las métricas usadas para evaluar la segmentación del modelo son Dice similarity coefficient (DSC), sensitivity, specificity y el accuracy.

Los resultados muestran que en el caso de los nódulos benignos todos los modelos (el construido en el artículo y los de referencia) logran buenos resultados, esto debido a la buena calidad de las imágenes. En el caso de los nódulos malignos, el modelo TNSNet superó por completo a los otros modelos de referencia. Hubo otros experimentos, en los cuales, el modelo presentado en este artículo, de manera general, obtuvo mejores resultados.

La investigación realizada por Yang et al. (2023) se centra en la problemática del cáncer de piel, específicamente en el grave problema de los melanomas y la alta necesidad de una detección temprana para mejorar las posibilidades de supervivencia de los pacientes con este mal.

Además, se menciona que aunque la técnica más usada para la detección de melanomas es la dermatoscopia, su precisión es muy variada dependiendo el tipo de cáncer de piel con el que se está tratando. En este contexto, las técnicas de Deep Learning han demostrado ser un herramienta capaz de mejorar la precisión en las tareas relacionadas a clasificar el cáncer de piel, incluso superando al desempeño de los propios dermatólogos.

A pesar de estos logros del desarrollo de Deep Learning en esta área, se menciona que aún hay dificultad para la clasificación de algunos tipos de cáncer de piel como el melanoma. Por este motivo se propone una nueva arquitectura de red basada en los transformadores con el objetivo de mejorar la capacidad de los algoritmos de clasificación de cáncer de piel.

La metodología seguida por los autores inicia con la preparación de los datos. En este caso se usó el conjunto de datos HAM10000 que consta de 7 clases distintas de cáncer de piel. Las imágenes tuvieron que ser redimensionadas para los 4 distintos modelos a entrenar: Inception ResNet con Soft Attention (IRv2 + SA), ResNet50 con Soft Attention (ResNet50 + SA), ViTfSCD-Large y ViTfSCD-Base. Los parámetros de estos dos últimos modelos se muestran en la siguiente figura.

Además, se aplicó a las imágenes una limpieza, eliminando los duplicados. También pasó por un proceso de balanceo de las clases donde se intentó igualar la cantidad de imágenes en cada una de estas. Se realizó el proceso de aumento de datos a través de modificaciones en las imágenes como alterar la saturación de forma aleatoria, cambiar el contraste y el brillo. Finalmente, se definió los porcentajes de división para el conjunto de datos, siendo 85 % para

el entrenamiento y 15% para la prueba.

Para la evaluación de los modelos se utilizaron las métricas de accuracy, precision, recall (sensitivity), specificity y F1 score. Los resultados de evaluar su capacidad de clasificación de los 4 modelos se muestran a continuación.

Skin cancer type	Precision				Sensitivity			
	IRv2	Res	ViTf	ViTf	IRv2	Res	ViTf	ViTf
	+	Net	SCD-B	SCD-L	+	Net	SCD-B	SCD-L
	SA	50 +	SA	SA	SA	50 +	SA	SA
AKIEC	0.81	0.67	0.82	0.66	0.57	0.52	0.61	0.83
BCC	0.95	0.90	1.00	0.89	0.73	0.69	0.54	0.65
BKL	0.73	0.72	0.81	0.86	0.79	0.70	0.65	0.76
DF	0.62	0.62	0.67	0.60	0.83	0.83	1.00	1.00
MEL	0.70	0.62	0.60	0.79	0.47	0.59	0.53	0.68
NV	0.95	0.96	0.94	0.97	0.97	0.98	0.98	0.99
VASC	0.91	0.83	0.83	1.00	1.00	1.00	1.00	1.00

Figura 9. Comparación de resultados de los modelos (precision y sensitivity).

Fuente: Yang et al. (2023). *A Novel Vision Transformer Model for Skin Cancer Classification*.

F1 score				Specificity			
IRv2 + SA	Res Net 50 + SA	ViTf SCD-B	ViTf SCD-L	IRv2 + SA	Res Net 50 + SA	ViTf SCD-B	ViTf SCD-L
0.67	0.59	0.70	0.73	1.00	0.99	1.00	0.99
0.83	0.78	0.70	0.76	1.00	1.00	1.00	1.00
0.76	0.71	0.72	0.81	0.98	0.98	0.99	0.99
0.71	0.71	0.80	0.75	1.00	1.00	1.00	1.00
0.56	0.61	0.56	0.73	0.99	0.99	0.99	0.99
0.96	0.97	0.96	0.98	0.80	0.84	0.76	0.88
0.95	0.91	0.91	1.00	1.00	1.00	1.00	1.00

Figura 10. Comparación de resultados de los modelos (f1-score y specificity).

Fuente: Yang et al. (2023). *A Novel Vision Transformer Model for Skin Cancer Classification.*

De los resultados, se observó que el modelo ViTfSCD-Large obtuvo el más alto precision, recall, y F1 scores. Además, el mismo modelo obtuvo un accuracy de 94.1 %, siendo el valor más alto, mientras que el modelo ViTfSCD-Base obtuvo 91.4 %.

Finalmente, se realizó una comparación de los modelos a través de su accuracy. En esta última sección, se comparó con otros modelos del estado de arte desarrollados con el conjunto de datos HAM10000. El primero de estos, implementado en 2020 y denominado EfficientNet logró un accuracy de 92.6 %. También, en el año 2021, se desarrolló un método semi supervisado para la clasificación de imágenes médicas que obtuvo un accuracy de 92.5 %.

Para esta comparación final, también se consideraron los resultados originales de los modelos IRv2, IRv2 + SA, ResNet50 y ResNet50 + SA junto con los resultados obtenidos en esta investigación. Además, también se realizaron experimentos con los modelos ViT originales (ViT-Base 16 y ViT-Large 16). Los resultados finales muestran que los modelos ViTfSCD-Base y ViTfSCD-Large son superiores a sus versiones originales.

Methods	Year	Overall accuracy in their paper	Overall accuracy in our experiments
Loss balancing and ensemble [17]	2020	92.6%	–
Semi-supervised [18]	2021	92.54%	–
ResNet50 [10]	2021	90.5%	–
IRv2 [10]	2021	91.2%	–
ResNet50 with Soft Attention [10]	2021	91.5%	91.5%
Inception ResNet with Soft Attention [10]	2021	93.4%	91.9%
Vision transformer (ViT)-Base16	2021	–	91.1%
ViT-Large16	2021	–	93.7%
ViT for skin cancer detection-Base	2022	–	91.4%
ViT for skin cancer detection-Large	2022	–	94.1%

Figura 11. Comparación de resultados según accuracy con el conjunto de datos HAM10000.

Fuente: Yang et al. (2023). *A Novel Vision Transformer Model for Skin Cancer Classification.*

También se realizaron experimentos con el conjunto de datos DERMOFIT que consta de imágenes de lesiones clínicas de la piel divididas en 10 clases distintas. Los resultados originales con este conjunto de datos muestran que el modelo ResNet50 tuvo mejor desempeño que el modelo de árbol de decisión y KNN.

Los nuevos experimentos con este conjunto de datos se llevaron a cabo con los modelos ResNet50, IRv2 y ViTfSCD-Base. Los siguientes resultados muestran el desempeño superior del modelo ViTfSCD frente a los demás.

Methods	Year	Overall accuracy in their paper	Overall accuracy in our experiments
Decision tree [29]	2020	78.1%	–
Flat ResNet50 [29]	2020	78.7%	–
ResNet50	2022	–	71.2%
Inception ResNet (IRv2)	2022	–	71.9%
IRv2 + Soft Attention	2022	–	75.0%
Vision transformer for skin cancer detection-Base	2022	–	80.5%

Figura 12. Comparación de resultados según accuracy con el conjunto de datos DERMOfIT.

Fuente: Yang et al. (2023). *A Novel Vision Transformer Model for Skin Cancer Classification.*

La investigación presentada por Ayana et al. (2023) se enfoca en el cáncer más común de las mujeres en los Estados Unidos: el cáncer de mama. Se dice que la tasa de incidencia de este mal ha ido en aumento en 0.5 % cada año; sin embargo, la tasa de mortalidad ha caído a un 45 % considerando datos desde el año 1989 hasta el 2020, esto debido a las mejoras constantes de los tratamientos y a las detecciones a tiempo, siendo la mamografía una técnica crucial en este último.

A pesar del importante papel de las mamografías en la detección a tiempo del cáncer de mama, existen casos de diagnósticos erróneos que muchas veces llevan a procedimientos y gastos innecesarios para un paciente.

Ante este problema, se dice que se han desarrollado diversos sistema de detección asistida por computadora o CAD por sus siglas en inglés que tiene el principal objetivo de reducir los errores en la detección de este cáncer.

Existen varios tipos de sistema CAD. Algunos no usan la inteligencia artificial, mientras otros utilizan las técnicas altamente difundidas del Deep Learning. Específicamente los modelos basados en las redes neuronales convolucionales (CNN) han sido consideradas como prometedoras para las tareas detección de tumores de mama, llevándolos incluso a ser capaces de reducir la probabilidad de error humano. Sin embargo, esta clase de modelos también tienen sus dificultades.

Los modelos CNN son computacionalmente caros debido a la complejidad y cantidad

de convoluciones que se realizan. Además, en el contexto del cáncer de mama, estos modelos tienen dificultades en la localización de tumores, y una alta necesidad en el pre procesamiento, pues es necesario mejorar la calidad y reducir el ruido de las imágenes.

Otro problema a considerar es la falta de conjuntos de datos para el análisis de imágenes médicas. Para abordar esto, es necesario aplicar transfer learning y el aumento de datos.

En esta investigación se desarrolló un modelo de Deep Learning basado en los Vision Transformers y el transfer learning para la detección de cáncer de mama a través de mamografías. Para lograr esto, primero se abordó el problema de desbalanceo de las dos clases (benigno y maligno) en el conjunto de datos de mamografías, posteriormente, se desarrolló un método de transfer learning basado en los vision transformer con el objetivo de mejorar las deficiencias de los métodos de transfer learning basados en CNN.

El conjunto de datos usado para entrenar y probar el modelo fue el Digital Database for Screening Mammography (DDSM). Este consiste (hasta el último acceso del 12 de septiembre del 2022) de 13 128 imágenes mamográficas de las cuales 5 970 son benignas y 7 158 son malignas. Esto presentaba un claro desbalance con un ratio de 0.65:0.35. Esta distribución entre las clases podría generar errores en la etapa de aprendizaje del modelo, por este motivo, se desarrolló un nuevo método para el balanceo de datos a través de la técnica de aumento de datos, específicamente, se aplicó variaciones en la fluctuaciones de color de las imágenes, corrección gamma, giro horizontal, sal y pimientas, y afilado o sharpening. El objetivo fue balancear el conjunto de datos para realizar validación cruzada de 5 pliegues, para lograr esto, se dividió al conjunto de datos en 5 partes o pliegues. Para las primeras 4 partes, se colocaron 1 145 imágenes de tumores malignos en cada una, mientras que para el quinto pliegue se tenían 1 146. Además, en los primeros 4 pliegues se tuvieron 955 imágenes de tumores benignos, a la vez que en el quinto pliegue se tuvieron 956. Para el lograr el balanceo de clases, se sometió a imágenes de tumores benignos al proceso de aumento de imágenes 5 veces, mientras que para al grupo de imágenes de tumores malignos solo se aplicó una vez. Así, finalmente se obtuvo 1 146 imágenes para ambas clases presentes en cada pliegue. Este proceso se muestra en la figura a continuación.

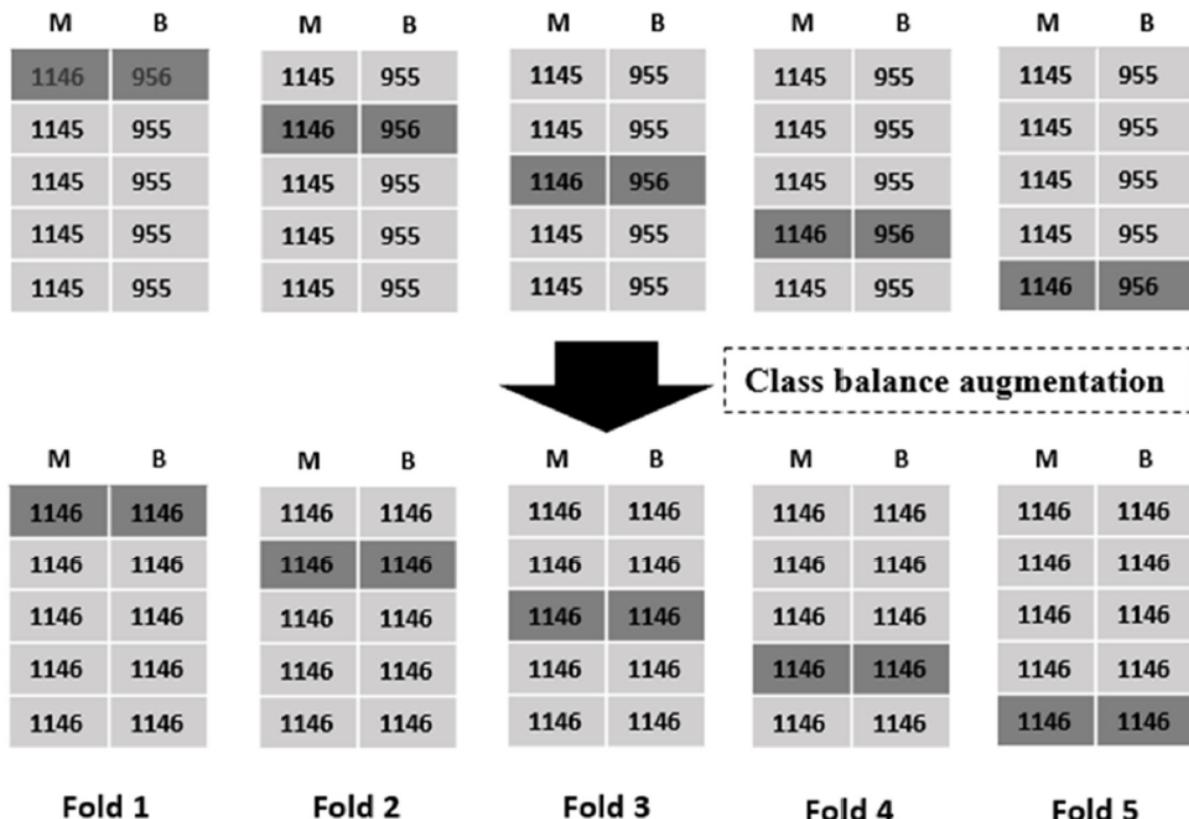


Figura 13. Balanceo de clases usando aumento de datos.

Fuente: Ayana et al. (2023). *Vision-Transformer-Based Transfer Learning for Mammogram Classification.*

Una vez obtenida la data ya balanceada, se realizó el preprocesamiento a las imágenes, específicamente, se redimensionó a las imágenes a 244 x 244 pixeles, esto debido a que el tamaño de las imágenes del conjunto de datos era variable.

El diseño del modelo consiste inicialmente de la arquitectura de vision transformer. Se explica que siendo un modelo derivado de los convencionales transformer usados para el proceso del lenguaje natural o NLP donde las entradas son de una sola dimensión, los vision transformer se adaptaron para recibir entradas de dos dimensiones (imágenes).

Específicamente, el modelo se encarga de dividir las entradas en pequeñas partes, también de dos dimensiones, conocidas como patches que finalmente servirán como entrada y serán pasadas como word tokens, de igual forma que en un modelo transformer NLP. Aunque antes de proveer estos patches, se realizan los procesos de flattening, sequence imbedding, learnable embedding y patch embedding.

Además del bloque transformer, se tiene una parte de MLP encargada de la tarea de

clasificación. Este posee una sola capa oculta y utiliza la función de activación GELU.

En el siguiente gráfico se muestra la estructura del modelo implementado en la investigación.

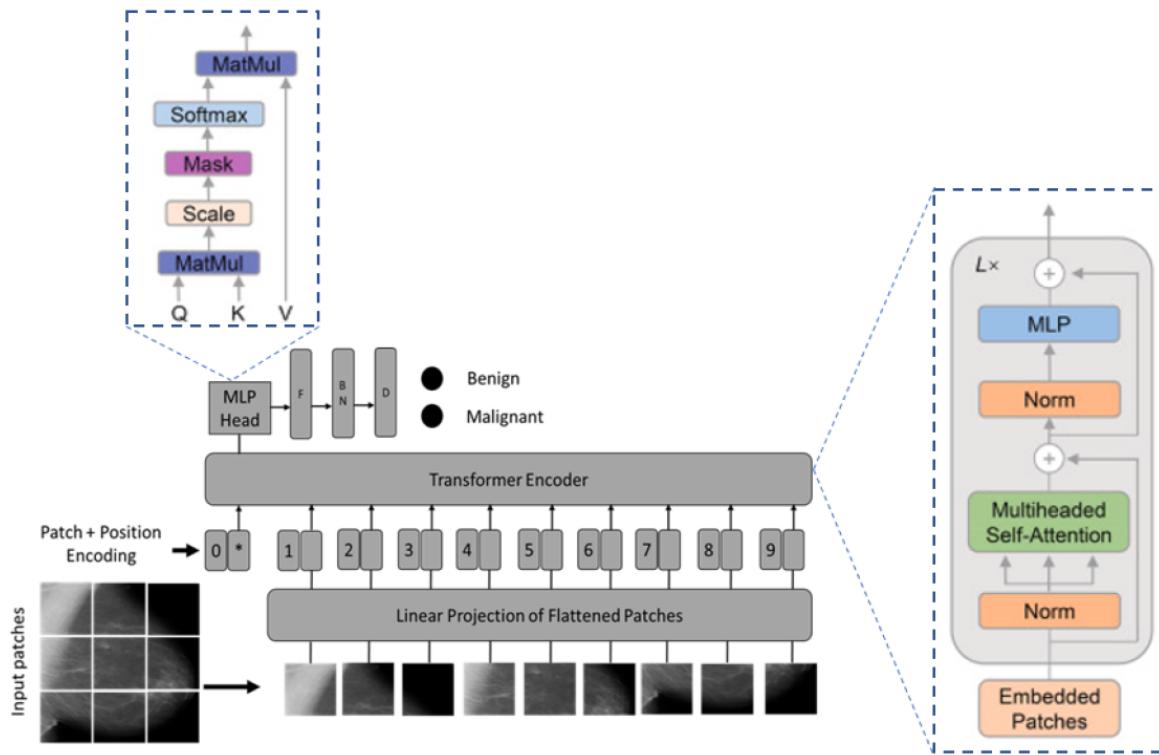


Figura 14. Estructura de ViT implementado.

Fuente: Ayana et al. (2023). *Vision-Transformer-Based Transfer Learning for Mammogram Classification.*

El transfer learning fue aplicado en la investigación a través del uso de modelos de vision transformer pre-entrenados con el extenso conjunto de datos ImageNet, y posteriormente utilizados para entrenar el conjunto de datos de imágenes mamográficas. Es decir que se aprovechó el conocimiento de dichos modelos para la tarea final de clasificación de mamografías en benigno y maligno. Los modelos usados para este proceso fueron el Vision Transformer (ViT), Swin Transformer (Swin-T) y el Pyramid Vision Transformer (PVT).

El modelo Swin-T posee cuatro distintas versiones: Swin-base, Swin-tiny, Swin-small y Swin-large. Las variantes usadas en esta investigación fueron el Swin-small y Swin-base. Mientras que de las 4 versiones de PVT (PVT-tiny, PVT-small, PVT-medium y PVT-large), se usaron solo el PVT-medium y PVT-large.

Los modelos fueron entrenados con 50 épocas, una tasa de aprendizaje de 0.0001, op-

timizador Adam y un tamaño de lote de 64. Además, se dividió el conjunto de datos en 80 % para entrenamiento y 20 % para prueba.

Los vision transformer usaron GELU como función de activación, mientras que los CNN usaron ReLu. Ambos usaron el regularizador L2.

También se aplicó validación cruzada de 5 para lograr una mejor comparación en el rendimiento de los modelos.

Las métricas usadas para el proceso de evaluación de los modelos fueron el accuracy, AUC, F1-score, precision, recall, el coeficiente de correlación de Matthew y el kappa score.

Se evaluó el método propuesto a través de 5 formas distintas: comparación de desempeño de los 3 modelos con transfer learning, comparación de los modelos vision transformer entrenados desde cero con el conjunto de datos de mamografías, comparación de transfer learning de vision transformer con los CNN, comparación de costo computacional de cada modelo vision transformer y, finalmente, se comparó el desempeño de los métodos desarrollados en esta investigación con otros que usaron el mismo conjunto de datos.

Como se muestra en la siguiente imagen, los 6 modelos transfer learning basados en arquitectura de vision transformer y entrenados con el conjunto de datos DDSM tuvieron un desempeño uniforme a través de todas las métricas.

Architecture	Model	Accuracy (95%)	AUC (95%)	F1 Score (95%)	Precision (95%)	Recall (95%)	MCC (95%)	Kappa (95%)
Vision transformer	ViT-base	1 ± 0	1 ± 0	1 ± 0	1 ± 0	1 ± 0	1 ± 0	1 ± 0
	ViT-large	1 ± 0	1 ± 0	1 ± 0	1 ± 0	1 ± 0	1 ± 0	1 ± 0
Swin transformer	Swin-small	1 ± 0	1 ± 0	1 ± 0	1 ± 0	1 ± 0	1 ± 0	1 ± 0
	Swin-base	1 ± 0	1 ± 0	1 ± 0	1 ± 0	1 ± 0	1 ± 0	1 ± 0
Pyramid vision transformer	PVT-medium	1 ± 0	1 ± 0	1 ± 0	1 ± 0	1 ± 0	1 ± 0	1 ± 0
	PVT-large	1 ± 0	1 ± 0	1 ± 0	1 ± 0	1 ± 0	1 ± 0	1 ± 0

Figura 15. Resultado de los modelos ViT basados en transfer learning en el conjunto de datos DDSM.

Fuente: Ayana et al. (2023). *Vision-Transformer-Based Transfer Learning for Mammogram Classification.*

La siguiente imagen muestra los resultados de los modelos vision transformer implementados desde cero y entrenados con el conjunto de datos DDSM.

Architecture	Model	Accuracy (95%)	AUC (95%)	F1 Score (95%)	Precision (95%)	Recall (95%)	MCC (95%)	Kappa (95%)
Vision transformer	ViT-base	0.74 ± 0.02	0.73 ± 0.03	0.74 ± 0.01	0.74 ± 0.01	0.74 ± 0.01	0.73 ± 0.03	0.73 ± 0.02
	ViT-large	0.72 ± 0.04	0.72 ± 0.02	0.72 ± 0.03	0.72 ± 0.04	0.72 ± 0.03	0.71 ± 0.02	0.72 ± 0.01
Swin transformer	Swin-small	0.75 ± 0.02	0.75 ± 0.03	0.75 ± 0.01	0.75 ± 0.02	0.75 ± 0.02	0.74 ± 0.03	0.74 ± 0.02
	Swin-base	0.76 ± 0.01	0.75 ± 0.02	0.75 ± 0.02	0.75 ± 0.01	0.76 ± 0.01	0.75 ± 0.01	0.75 ± 0.02
Pyramid vision transformer	PVT-medium	0.78 ± 0.02	0.77 ± 0.02	0.78 ± 0.01	0.78 ± 0.02	0.78 ± 0.02	0.77 ± 0.01	0.77 ± 0.02
	PVT-large	0.77 ± 0.03	0.77 ± 0.01	0.77 ± 0.02	0.77 ± 0.02	0.77 ± 0.02	0.77 ± 0.01	0.77 ± 0.01

Figura 16. Resultado de los modelo ViT entrenados desde cero en el conjunto de datos DDSM.

Fuente: Ayana et al. (2023). *Vision-Transformer-Based Transfer Learning for Mammogram Classification.*

Se puede observar que el modelo PVT-medium obtuvo un mejor desempeño comparado a los demás modelos. Sin embargo, estos resultados se mantienen muy por debajo de los presentados anteriormente.

Finalmente, también se obtuvieron resultados de los experimentos de transfer learning con modelos CNN y el conjunto de datos DDSM. Estos se muestran a continuación.

Architecture	Model	Accuracy (95%)	AUC (95%)	F1 Score (95%)	Precision (95%)	Recall (95%)	MCC (95%)	Kappa (95%)
ResNet	ResNet50	0.95 ± 0.01	0.96 ± 0.01	0.95 ± 0.01	0.95 ± 0.01	0.95 ± 0.01	0.94 ± 0.01	0.94 ± 0.02
	ResNet101	0.95 ± 0.01	0.95 ± 0.02	0.95 ± 0.01	0.95 ± 0.01	0.95 ± 0.01	0.94 ± 0.02	0.94 ± 0.02
EfficientNet	EfficientNetB0	0.94 ± 0.02	0.94 ± 0.01	0.94 ± 0.01	0.94 ± 0.01	0.94 ± 0.01	0.93 ± 0.03	0.93 ± 0.02
	EfficientNetB2	0.95 ± 0.01	0.95 ± 0.01	0.95 ± 0.01	0.95 ± 0.01	0.95 ± 0.01	0.95 ± 0.01	0.95 ± 0.01
InceptionNet	InceptionNetV2	0.93 ± 0.02	0.93 ± 0.01	0.93 ± 0.02	0.93 ± 0.02	0.93 ± 0.02	0.93 ± 0.03	0.92 ± 0.02
	InceptionNetV3	0.94 ± 0.01	0.94 ± 0.01	0.94 ± 0.01	0.94 ± 0.01	0.94 ± 0.01	0.93 ± 0.02	0.93 ± 0.02

Figura 17. Resultado de los modelos CNN basados en transfer learning en el conjunto de datos DDSM.

Fuente: Ayana et al. (2023). *Vision-Transformer-Based Transfer Learning for Mammogram Classification.*

El desempeño de los modelos CNN también se encuentran por debajo de los vision transformer presentados inicialmente.

Manzari et al. (2023) nos menciona que dentro del análisis de imágenes médicas, la tarea de clasificación es crucial. Por ello, este se ha visto beneficiado constantemente de novedosos sistemas de diagnóstico asistido, siendo los de mayor desempeño aquellos basados en redes neuronales convolucionales (CNN), pues ofrecen predicciones cada vez más precisas incluso comparándose con los médicos. Sin embargo, estos también tienen algunas desventajas

o dificultades, principalmente lo relacionado a su capacidad de aprendizaje de dependencias a largo alcance en conjuntos de datos visuales, esto debido al conocido sesgo local de las CNN.

En este contexto, las arquitecturas basadas en transformers y su mecanismo de auto-atención, han demostrado ser capaces de solucionar estas desventajas de las CNN. Esto debido a su capacidad de modelar las dependencias a largo alcance de los datos, convirtiéndolos así en modelos superiores a las arquitectura de CNN, aunque con mayor necesidad en cantidad de datos de entrenamiento, recursos computacionales y tiempo.

A pesar de los beneficios que pueden traer las arquitecturas transformer frente a los CNN, estos están muy limitadas en su uso en las tareas de análisis de imágenes médicas. Esto es debido principalmente a sus desventajas ya mencionadas (alta necesidad de recursos computacionales y tiempo), pues en el entorno clínico, la eficiencia y la velocidad es de suma importancia para realizar diagnósticos satisfactorios.

Los altos requerimientos de los transformer traen consigo grandes dificultades, impiadiendo que estos puedan ser usados en situaciones clínicas reales. Además, también existe el problema de la suposición de una distribución similar en los datos que se van a ingresar al modelo; sin embargo, se sabe que esto no siempre se cumple en el entorno real, debido a que las imágenes médicas son capturadas a través de distintos dispositivos, diversos protocolos y diferentes lugares. Esta discrepancia en los datos puede generar que el rendimiento del modelo caiga considerablemente.

Ante este problema, la investigación propone un nueva arquitectura basada en transformer con capacidad de generalización de distintos tipos de imágenes médicas como las tomografías computarizadas, rayos X y ultrasonido. Su estructura se caracteriza por ser híbrida jerárquica, poseer patch embedding y poseer convoluciones y bloques de transformer. Además, se incluyó en la arquitectura del transformer un bloque de atención convolucional multi-cabezal con el fin de mejorar la capacidad de aprendizaje de representación local, incluso este permitió reducir la complejidad computacional comparado con un modelo transformer convencional.

Comparándolo con los CNN, el modelo desarrollado también demostró una capacidad superior en la tarea de predicción, al mismo tiempo que poseía menor complejidad.

De forma generar el modelo MedViT, propuesto en esta investigación, tiene como objetivo lograr una arquitectura híbrida capaz de desempeñarse en la tarea de clasificación de imágenes médicas, a través de la combinación de bloques de convolución y transformer, además de estar también compuesto por capas de path embedding, siguiendo así una tradicional arquitectura piramidal jerárquica.

Los bloques principales de modelo son el Efficient Convolution Block (ECB) y el Lo-

cal Transformer Block (LTB). Estos se encargan de capturar las dependencias de los datos de entrada a largo y corto plazo.

También se propuso una nueva técnica de aumento de datos denominada Patch Momentum Charge (PMC) con el objetivo de mejorar el rendimiento del modelo.

Los experimentos se realizaron con 12 distintos conjuntos de datos de la categoría imágenes médicas. Todos estos se encontraron en MedMINIST, y conforman imágenes de tomografías computarizadas, rayos X, ultrasonido y tomografías de coherencia óptica (OCT). Esta variedad de datos permiten el entrenamiento de modelos en tareas de clasificación.

De forma específica, los conjuntos de datos usados fueron el PathMNIST (posee 100 000 image patches categorizadas en 9 clases distintas relacionadas a patologías en el colon), ChestMNIST (posee 112 120 imágenes frontales de rayos X de 32 717 pacientes dividida en 14 clases distintas de enfermedades de tórax), DermaMNIST (posee 10 015 imágenes dermatoscópicas recopiladas de distintas fuentes y divididas en 7 clases relacionadas a enfermedades de la piel), OCTMNIST (consta de 109 309 imágenes de OCT recopiladas de pacientes con enfermedades de la retina y dividida en 4 clases distintas), PneumoniaMNIST (recopila 5 856 imágenes radiográficas pediátricas de tórax categorizadas en 2 clases que indican presencia o no de neumonía), RetinaMNIST (provee 1 600 imágenes de fondo de retina de 628 pacientes y se divide en 5 clases que indican la gravedad de la enfermedad de retinopatía diabética), TissueMNIST (dividida en 8 clases, este conjunto de datos está compuesto de 236 386 imágenes segmentadas de tejidos de corteza renal), BloodMNIST (consta de 17 092 imágenes de células sanguíneas categorizadas en 8 clases), BreastMNIST (categorizada en las clases de benigno, maligno y normal, este conjunto de datos posee 780 ecografías mamarias) y OrganMNIST (consta de imágenes de tomografías computarizadas abdominales divididas en 11 clases en sus 3 versiones: Axial, Coronal y Sagital).

En la etapa de entrenamiento del modelo con los 12 conjuntos de datos descritos anteriormente, se siguieron los mismos ajustes de entrenamiento de MedMNISTv2. Se usaron 100 épocas con un tamaño de lote de 128 y se aplicó el redimensionado a las imágenes para obtener la forma de 224 x 224 pixeles. Además, se usó el optimizar AdamW con una tasa de aprendizaje de 0.0001.

MedViT consiste en 3 distintas arquitecturas definidas por su tamaño: MedViT-T, MedViT-S y MedViT-L. Estas fueron entrenadas de forma separada con cada uno de los conjuntos de datos. También se incluyó el aumento de datos a través de PMC.en la etapa de entrenamiento.

Las métricas usadas para evaluar las 3 versiones del modelo MedViT fueron el accuracy y la curva ROC. Esta elección fue debido a que el experimento se llevó a cabo en varios

conjuntos de datos distintos entre sí.

De los resultados mostrados a continuación, los modelos MedViT superan, de forma general, en gran medida a los modelos basados en CNN en la misma tarea. Esto lleva a concluir que el modelo demuestra una alta capacidad de generalización en imágenes médicas.

Methods	PathMNIST		ChestMNIST		DermaMNIST		OCTMNIST		PneumoniaMNIST		RetinaMNIST	
	AUC	ACC	AUC	ACC	AUC	ACC	AUC	ACC	AUC	ACC	AUC	ACC
ResNet-18 (28) [20]	0.983	0.907	0.768	0.947	0.917	0.735	0.943	0.743	0.944	0.854	0.717	0.524
ResNet-18 (224) [20]	0.989	0.909	0.773	0.947	0.920	0.754	0.958	0.763	0.956	0.864	0.710	0.493
ResNet-50 (28) [20]	0.990	0.911	0.769	0.947	0.913	0.735	0.952	0.762	0.948	0.854	0.726	0.528
ResNet-50 (224) [20]	0.989	0.892	0.773	0.948	0.912	0.731	0.958	0.776	0.962	0.884	0.716	0.511
auto-sklearn [75]	0.934	0.716	0.649	0.779	0.902	0.719	0.887	0.601	0.942	0.855	0.690	0.515
AutoKeras [76]	0.959	0.834	0.742	0.937	0.915	0.749	0.955	0.763	0.947	0.878	0.719	0.503
Google AutoML [77]	0.944	0.728	0.778	0.948	0.914	0.768	0.963	0.771	0.991	0.946	0.750	0.531
MedViT-T (224)	0.994	0.938	0.786	0.956	0.914	0.768	0.961	0.767	0.993	0.949	0.752	0.534
MedViT-S (224)	0.993	0.942	0.791	0.954	0.937	0.780	0.960	0.782	0.995	0.961	0.773	0.561
MedViT-L (224)	0.984	0.933	0.805	0.959	0.920	0.773	0.945	0.761	0.991	0.921	0.754	0.552
Methods	BreastMNIST		BloodMNIST		TissueMNIST		OrganAMNIST		OrganCMNIST		OrganSMNIST	
	AUC	ACC	AUC	ACC	AUC	ACC	AUC	ACC	AUC	ACC	AUC	ACC
ResNet-18 (28) [20]	0.901	0.863	0.998	0.958	0.930	0.676	0.997	0.935	0.992	0.900	0.972	0.782
ResNet-18 (224) [20]	0.891	0.833	0.998	0.963	0.933	0.681	0.998	0.951	0.994	0.920	0.974	0.778
ResNet-50 (28) [20]	0.857	0.812	0.997	0.956	0.931	0.680	0.997	0.935	0.992	0.905	0.972	0.770
ResNet-50 (224) [20]	0.866	0.842	0.997	0.950	0.932	0.680	0.998	0.947	0.993	0.911	0.975	0.785
auto-sklearn [75]	0.836	0.803	0.984	0.878	0.828	0.532	0.963	0.762	0.976	0.829	0.945	0.672
AutoKeras [76]	0.871	0.831	0.998	0.961	0.941	0.703	0.994	0.905	0.990	0.879	0.974	0.813
Google AutoML [77]	0.919	0.861	0.998	0.966	0.924	0.673	0.990	0.886	0.988	0.877	0.964	0.749
MedViT-T (224)	0.934	0.896	0.996	0.950	0.943	0.703	0.995	0.931	0.991	0.901	0.972	0.789
MedViT-S (224)	0.938	0.897	0.997	0.951	0.952	0.731	0.996	0.928	0.993	0.916	0.987	0.805
MedViT-L (224)	0.929	0.883	0.996	0.954	0.935	0.699	0.997	0.943	0.994	0.922	0.973	0.806

Figura 18. Comparación de resultados de los modelos MedViT con los CNN y AutoML.

Fuente: Manzari et al. (2023). *MedViT: A robust vision transformer for generalized medical image classification.*

Regmi et al. (2023) nos menciona la gran importancia que tiene el análisis de imágenes médicas para la detección a tiempo de enfermedades potencialmente mortales. Los radiólogos y personal clínico realizan esta importante tarea en la gran mayoría de los casos; sin embargo, sus interpretaciones de las imágenes no siempre están en lo correcto debido a que su análisis muchas veces está condicionado por el mismo observador, llevando a que los niveles de precisión para interpretar este tipo de datos no sean tan altos.

Esta tarea de interpretación de imágenes médicas es aun más importante en los casos de una alta necesidad de detectar a tiempo alguna enfermedad; por ejemplo, las relacionadas a los pulmones o las enfermedades gastrointestinales que pueden no causar daño como el reflujo ácido u otras de grave impacto como el cáncer de colon. Una detección a tiempo de estos permite aplicar los tratamientos debidos y así, consecuentemente, mejorar la calidad de vida de los pacientes. En este contexto, los sistemas de diagnóstico asistido por computadora (CAD por sus siglas en inglés) han demostrado ser de gran ayuda para los médicos y expertos clínicos en la tarea de realizar diagnósticos tempranos.

La capacidad de las redes neuronales convolucionales (CNN) de poder adaptarse en detectar distintas característica de un conjunto de datos han permitido su desarrollo en el campo médico, específicamente en las tareas de segmentación, clasificación, registro y reconstrucción de imágenes médicas.

Otra técnica popular que va tomando cada vez más mayor importancia son los Vision Transformer (ViT) capaces también de desempeñarse satisfactoriamente en las tareas de clasificación de imágenes, a pesar de ser basados en los originales transformers especializados en tareas relacionadas a texto.

Otro tipo de arquitectura basada en transformer son los Data-Efficient Image Transformer (DeiT), una versión de nuevos de los ViT caracterizados por su baja dependencia a los datos a través de su enfoque maestro-estudiante.

En esta investigación se presentan distintas arquitecturas basadas en CNN y Transformer para la clasificación de radiografías de tórax en distintos tipos de enfermedades relacionadas. Específicamente, se usaron los modelos DeiT, ViT y Ensemble, este último basado en otros modelos CNN. También se aplicó transfer learning junto a modelos CNN previamente entrenados. Finalmente todos los modelos fueron comparados en la misma tarea de clasificación de imágenes.

Se usaron distintas versiones de ViT (ViT-B/16, ViT-L/16, ViT-L/32) pre-entrenados con el conjunto de datos ImageNet-21k.

La diferencia principal entre ViT-L/16 y ViT-L/32 radica en el tamaño de los patch de entrada que permiten, siendo para el primer caso un patch de tamaño 16x16, mientras que el segundo un patch de 32 x 32. Todos estos modelos pasaron por un proceso de fine-tuning con 3 conjuntos de datos.

El primer conjunto de datos usado fue el Chest X-ray dataset que consistía en 7 135 imágenes, divididas en 4 clases (COVID-19, neumonía, tuberculosis y normal).

El conjunto de datos Kvasir consiste de 1 000 imágenes en cada una de las 8 clases existentes.

El último conjunto de datos, Kvasir-Capsule dataset consiste de 44 228 imágenes de endoscopía divididas en 13 clases. La cantidad de imágenes varía en cada clase del conjunto de datos, lo cual lo vuelve altamente imbalanceado.

A continuación se resumen las características de los 3 conjuntos de datos.

Dataset	No. of Images	Input size	Train	Valid	Test	Application
Chest X-ray dataset	7135	variable	5693	671	771	Lung disease
Kvasir dataset	8000	720 × 556	6,400	800	800	Endoscopy
Kvasir-Capsule	44,152	336 × 336	19,280	4,820	23061	Bowel disease

Figura 19. Resumen de características de los conjuntos de datos.

Fuente: Regmi et al. (2023). *Vision transformer for efficient chest X-ray and gastrointestinal image classification.*

El modelo ViT base (ViT-B) está compuesto por 12 bloques transformer con módulos de autoatención de 12 cabezas, permitiendo un total de 86 millones de parámetros entrenables.

El modelo ViT large (ViT-L) consiste de 24 bloques transformer con módulos de autoatención de 16 cabezas, formando así 307 millones de parámetros entrenables.

Las imágenes del conjunto de datos Chest X-ray fueron redimensionadas a 224 x 224 pixeles. Esto también se aplicó al conjunto de datos Kvasir.

En el caso de las imágenes de Kvasir-Capsule, estas tuvieron que ser redimensionadas a 64 x 64 pixeles.

Se usaron dos modelos de DeiT: DeiT-Ti y DeiT-B 384. El primero de estos permitía imágenes de 224 x 224 de resolución; mientras que el segundo, 384 x 384.

Todos los conjuntos de datos fueron divididos en data de entrenamiento, validación y prueba.

Además, se aplicaron distintas técnicas de aumento de datos para cada uno de los conjuntos de datos. En el caso de Chest X-ray se aplicó el reescalado, cambio de brillo, desplazamiento vertical y horizontal, y zoom de forma aleatoria. Para el conjunto de datos Kvasir, se alteró el brillo de las imágenes, se rotaron y se dieron vuelta de manera vertical. Finalmente, para Kvasir-Capsule, se aplicó rotación, se dio vuelta a las imágenes de manera horizontal y vertical.

Los modelos CNN pre-entrenados que se usaron fueron el DenseNet201, DenseNet121, InceptionRestNetV2, Xception, MovileNetV2 y el Ensemble que combinaba DenseNet201 y DenseNet121.

Se probaron varios hiper-parámetros a través de prueba y error. Además, en el caso del conjunto de datos de Chest X-ray y Kvasir, se usó la función de pérdida categorícal cross-

entropy. Con Kvasir-Capsule, se usó el Focal Loss debido al desbalance de los datos en cada clase.

Las métricas usadas para evaluar la capacidad de clasificación de los modelos fueron el Matthews Correlation Coefficient (MCC), Frames Per Second (FPS), precision, F1-score, recall y accuracy.

El precision utilizado fue la media ponderada, esto pues es un método que asigna pesos de acuerdo a la cantidad de observaciones en cada clase, volviéndolo ideal para conjuntos de datos desbalanceados. Además, también se obtiene la desviación estándar del Recall, F1-score y Precision con el fin de verificar la variabilidad de los resultados.

Se realizó la prueba t por pares para comparar los resultados con la métrica MCC del modelo con mejor rendimiento.

Los resultados de los experimentos se muestran a continuación.

Method	Precision	Recall	F1-score	Accuracy	MCC	P-values	FPS
DenseNet201 + DenseNet121	0.9345 ± 0.0324	0.9326 ± 0.0584	0.9319 ± 0.0236	0.9326	0.8931	1.24e-03	16.08
DenseNet201	0.9169 ± 0.0358	0.9157 ± 0.0538	0.9150 ± 0.0195	0.9157	0.8658	8.41e-05	22.47
DenseNet121	0.9314 ± 0.0358	0.9287 ± 0.0678	0.9277 ± 0.0370	0.9287	0.8878	3.37e-04	26.15
InceptionResNetV2	0.9415 ± 0.0358	0.9403 ± 0.0678	0.9400 ± 0.0370	0.9287	0.9052	1.18e-04	20.48
Xception	0.9380 ± 0.0324	0.9377 ± 0.0461	0.9375 ± 0.0358	0.9377	0.9010	4.80e-05	21.34
MobileNetV2	0.9133 ± 0.0625	0.9092 ± 0.0508	0.9099 ± 0.0401	0.9092	0.8567	8.96e-04	21.85
DeiT-Ti	0.9266 ± 0.0453	0.9261 ± 0.0326	0.9262 ± 0.0368	0.9261	0.8821	3.37e-04	25.48
DeiT-B 384	0.9332 ± 0.0592	0.9300 ± 0.0728	0.9291 ± 0.0272	0.9300	0.8899	1.10e-03	16.75
ViT-L/32	0.9527 ± 0.0362	0.9520 ± 0.0301	0.9521 ± 0.0277	0.9520	0.9236	3.83e-02	11.49
ViT-L/16	0.9533 ± 0.0202	0.9533 ± 0.0301	0.9532 ± 0.0225	0.9533	0.9259	-	11.20
ViT-B/16	0.9291 ± 0.0592	0.9248 ± 0.0728	0.9248 ± 0.0272	0.9248	0.8813	3.77e-04	20.74

Figura 20. Resultados con el conjunto de datos Chest X-ray.

Fuente: Regmi et al. (2023). *Vision transformer for efficient chest X-ray and gastrointestinal image classification.*

Method	Precision	Recall	F1-score	Accuracy	MCC	P-values	FPS
DenseNet201 + DenseNet121	0.9284 ± 0.0559	0.9263 ± 0.0711	0.9258 ± 0.0538	0.9263	0.9161	2.60e-01	12.32
DenseNet201	0.9262 ± 0.0467	0.9250 ± 0.0628	0.9246 ± 0.0472	0.9250	0.9146	1.52e-01	21.59
DenseNet121	0.9136 ± 0.0523	0.9112 ± 0.0749	0.9105 ± 0.0515	0.9113	0.8991	6.63e-02	24.35
InceptionResNetV2	0.8877 ± 0.0619	0.8875 ± 0.0753	0.8870 ± 0.0648	0.8875	0.8716	1.90e-02	21.18
Xception	0.9032 ± 0.0611	0.9025 ± 0.0761	0.9020 ± 0.0639	0.9025	0.8888	4.64e-02	43.12
MobileNetV2	0.8789 ± 0.0689	0.8775 ± 0.0826	0.8769 ± 0.0691	0.8775	0.8603	1.42e-02	43.97
DeiT-Ti	0.9353 ± 0.0337	0.9350 ± 0.0394	0.9349 ± 0.0338	0.9350	0.9258	5.82e-01	25.80
DeiT-B/384	0.9408 ± 0.0401	0.9401 ± 0.0466	0.9399 ± 0.0370	0.9401	0.9319	3.73e-01	15.81
ViT-L/32	0.9375 ± 0.0532	0.9337 ± 0.0698	0.9333 ± 0.0458	0.9337	0.9249	4.55e-01	13.35
ViT-L/16	0.9454 ± 0.0400	0.9437 ± 0.0487	0.9436 ± 0.0345	0.9437	0.9360	-	11.34
ViT-B/16	0.9433 ± 0.0427	0.9400 ± 0.0532	0.9398 ± 0.0260	0.9400	0.9316	5.53e-01	21.50

Figura 21. Resultados con el conjunto de datos Kvasir.

Fuente: Regmi et al. (2023). *Vision transformer for efficient chest X-ray and gastrointestinal image classification.*

Method	Precision	Recall	F1-score	Accuracy	MCC	P-values	FPS
DenseNet201 + DenseNet121	0.6633 ± 0.2485	0.7230 ± 0.2859	0.6737 ± 0.2594	0.7230	0.3560	9.63e-03	578.15
DenseNet201	0.6606 ± 0.2522	0.7198 ± 0.2820	0.6737 ± 0.2583	0.7198	0.3585	8.88e-03	1142.90
DenseNet121	0.6564 ± 0.2660	0.7187 ± 0.2848	0.6720 ± 0.2621	0.7187	0.3500	4.28e-03	1143.35
InceptionResNetV2	0.6059 ± 0.2647	0.6900 ± 0.2742	0.6548 ± 0.0336	0.6203	0.2277	3.06e-04	544.81
Xception	0.6159 ± 0.2484	0.6895 ± 0.2710	0.6310 ± 0.2431	0.6895	0.2544	3.34e-04	1223.44
MobileNetV2	0.5903 ± 0.2645	0.6800 ± 0.2732	0.5925 ± 0.2342	0.6800	0.1637	7.041e-04	3555.12
DeiT-Ti	0.6100 ± 0.2603	0.6839 ± 0.2669	0.6203 ± 0.2431	0.6839	0.2212	5.06e-05	281.24
DeiT-B/384	0.6496 ± 0.3020	0.7180 ± 0.2770	0.6657 ± 0.3010	0.7185	0.3422	6.10e-04	520.21
ViT-L/32	0.6483 ± 0.2922	0.7182 ± 0.2985	0.6631 ± 0.2760	0.7182	0.3377	3.11e-02	343.60
ViT-L/16	0.6425 ± 0.2806	0.6751 ± 0.2725	0.6405 ± 0.2581	0.6751	0.2637	1.69e-03	262.09
ViT-B/16	0.6841 ± 0.2985	0.7156 ± 0.2899	0.7156 ± 0.2779	0.7156	0.3705	-	570.53

Figura 22. Resultados con el conjunto de datos Kvasir-Capsule.

Fuente: Regmi et al. (2023). *Vision transformer for efficient chest X-ray and gastrointestinal image classification.*

Tampu et al. (2023) nos menciona que las tomografías de coherencia óptica (OCT) son un tipo de técnica de imágenes médicas que permiten la visualización de la microestructura de tejidos. Este ha sido altamente utilizado en distintas situaciones de análisis de imágenes médicas; por ejemplo, para las imágenes de arterias coronarias, el diagnóstico de cáncer, la gastroenterología y la odontología.

Debido a la capacidad de proporcionar imágenes de alta calidad, esta técnica es usada para la toma de decisiones en situaciones quirúrgicas a través de su implementación con dispositivos. Por ejemplo, esto es de gran ayuda en los casos de cirugías en la tiroides donde se necesita distinguir entre distintos tipos de tejidos y no dañar partes críticas del organismo, reduciendo así la morbilidad de los pacientes.

A pesar de estas grandes ventajas y aplicaciones de las imágenes OCT, su uso e inter-

pretabilidad está limitado a un pequeño número de profesionales clínicos. Por este motivo, se han ido desarrollando distintos métodos de análisis de imágenes médicas con el fin de facilitar su interpretación; sin embargo, estos están comúnmente basados en las técnicas tradicionales de procesamiento de imágenes que poseen dos grandes limitaciones: procesamiento lento y necesidad de intervención manual.

En este contexto, se han desarrollado nuevos métodos para sobrelevar las desventajas de los OCT. Los más conocidos son los basados en las redes neuronales convolucionales (CNN) que han ido demostrando los últimos años su capacidad de obtener un alto rendimiento en tareas de clasificación, segmentación y detección de objetivos en imágenes.

Existe una gran variedad de arquitecturas de Deep Learning basadas en CNN desarrolladas para la clasificación de imágenes. Las más conocidas son ResNet, AlexNet e Inception. Todas entrenadas con el conjunto de datos ImageNet que consta de imágenes naturales. Sin embargo, las imágenes médicas tienen un comportamiento distinto, y es que estas tienen toda la información dispersa en la imagen y no solo enfocada en una región. Además, se ha ido experimentando con redes poco profundas con este tipo de imágenes y los resultados muestran un desempeño similar a las grandes pre-entrenadas arquitecturas.

Otro tipo de redes, aparte de los CNN, desarrolladas en el campo del análisis de imágenes médicas son los Capsule Networks y Transformer.

Las redes de Deep Learning se han desarrollado en distintos campos de la medicina para la clasificación de tejidos enfermos a través de imágenes dadas por la técnica OCT. Esto ha llevado a que se tengan buenos desempeños en esta tarea, superando en la mayoría de los casos el 85 % en precisión.

El objetivo de esta investigación fue el desarrollo, evaluación y explicación de distintos modelos de Deep Learning en la tarea de clasificación de tejidos en las categorías de normal y enfermo en dos tipos de conjuntos de datos (2D y 3D OCT). Para lograr esto, se evaluaron los modelos actualmente disponibles para tareas con OCT, se implementó un Vision Transformer y se desarrollaron 2 modelos basados en CNN.

Para esta investigación se tuvieron dos tipos de conjuntos de datos.

El primer tipo, OCT 2D, consistía inicialmente de 66 988 2D b-scan (imágenes obtenidas a través de la técnica OCT) de la tiroides. Sin embargo, para una mejor investigación del impacto de resolución anisotrópica de los pixeles, se crearon dos conjuntos de datos. El primero estaba compuesto de imágenes anisotrópicas, mientras que el segundo se construyó a base de aplicar un remuestreo de las imágenes a una resolución isotrópica. Ambos conjuntos de datos tuvieron que ser recortadas a 200 x 200 píxeles. Finalmente, se seleccionó el modelo

LightOCT y la tarea de clasificación de densidad folicular (alta o baja) para la evaluación del impacto de este cambio en las imágenes.

Para la creación de conjunto de datos 3D, se tuvo el reto de superar las limitaciones de la capacidad de memoria del hardware gráfico, y es que el volumen original de los datos era demasiado grande para que los modelos pueden entrenar. De este modo, se realizó un proceso de extracción de 15 b-scan no consecutivos. Además, se pasó por un proceso de remuestreo para que tuvieran dimensiones isotrópicas y se aplicó un recorte para obtener un tamaño de 200 x 200 píxeles.

En la etapa de división de datos en entrenamiento y prueba, se empleó la técnica de división por volume/subject con el fin de evitar la evaluación errónea de los modelos debido al tipo de imágenes de OCT que comúnmente generan filtración de datos entre los conjuntos de entrenamiento y prueba.

Todas las tareas de clasificación tuvieron un conjunto de datos de prueba compuesta por 1000 imágenes 2D en cada una de las clases.

El conjunto de datos 3D fue dividido en 250 muestras para cada clase en la tarea de clasificación por enfermedad.

Los experimentos se desarrollaron con 5 distintos tipos de clasificación. El primer tipo de clasificación fue la distinción entre tejidos normales y anormales. El segundo tuvo el objetivo de categorizar según la estructura folicular (normal, agrandada o encogida). El tercero consistía en clasificar según la densidad folicular (alta o baja densidad). El cuarto tipo de clasificación determinaba si una estructura folicular era normal o agrandada. El último tuvo el objetivo de clasificar las muestras por enfermedad (6 clases).

Además de los mencionados anteriormente, también se evaluó el rendimiento de los modelos con dos conjuntos de datos de OCT de acceso abierto: Kermany versión 2 de oftalmología y AIIMS de tejidos mamarios cancerígenos. Esto con el objetivo de demostrar la capacidad de generalización de los modelos desarrollados.

Los modelos entrenados desde cero con los conjuntos de datos 2D fueron RestNet50, LightOCT, Custom Mk-CNN (multikernel CNN), Custom sResNet4 (shallow-ResNet) y Vision Transformer. En el caso del conjunto de datos 3D, se usaron dos modelos: 3D-LightOCT y 3D-ViT.

Para el entrenamiento de los modelos se tuvieron distintas configuraciones. El modelo LightOCT se optimizó a través del algoritmo de descenso por gradiente estocástico con el objetivo de reducir la pérdida definida por el categorical cross-entropy. Los demás modelos

usaron la función de pérdida weighted categorical cross-entropy con el optimizador Lookahead y el optimizador interno ADAM. Además, también se calcularon los pesos para el balanceo de clases del conjunto de datos de entrenamiento.

El entrenamiento de los modelos 2D fue con la técnica de validación cruzada 5-folds con 250 épocas. Además, para el conjunto de datos de estos modelos, se aplicó aumento de datos sobre la marcha (giro horizontal y vertical aleatorios, rotación aleatoria de 30 grados y zoom aleatorio de 10 %).

Para evaluar el rendimiento de los modelos se usó la matriz de confusión multiclas. También se calculó el Matthew's Correlation Coefficient (MCC), accuracy, precision, recall, F1-score y AUC para cada modelo.

A continuación se muestran los resultados finales.

2D data	Model	MCC [-1,1] m ± SD (ensemble)	Accuracy [0,1] m ± SD (ensemble)	F1-score [0,1] m ± SD (ensemble)
Task 1 <i>normal vs. abnormal</i> (2 classes)	LightOCT	0.23 ± 0.07 (0.21)	0.58 ± 0.03 (0.55)	0.49 ± 0.05 (0.44)
	ResNet50	0.31 ± 0.09 (0.27)	0.61 ± 0.06 (0.58)	0.54 ± 0.09 (0.49)
	Mk-CNN	0.04 ± 0.09 (-0.01)	0.51 ± 0.02 (0.50)	0.40 ± 0.04 (0.36)
	sResNet4	0.45 ± 0.07 (0.47)	0.68 ± 0.03 (0.69)	0.66 ± 0.04 (0.65)
	ViT	0.37 ± 0.06 (0.41)	0.68 ± 0.03 (0.70)	0.68 ± 0.03 (0.70)
Task 2 <i>Structure based</i> <i>(normal vs. enlarged vs. shrunk + depleted;</i> <i>3 classes)</i>	LightOCT	0.36 ± 0.10 (0.40)	0.57 ± 0.07 (0.59)	0.55 ± 0.07 (0.58)
	ResNet50	0.48 ± 0.11 (0.54)	0.64 ± 0.08 (0.67)	0.64 ± 0.08 (0.67)
	Mk-CNN	0.47 ± 0.13 (0.46)	0.62 ± 0.10 (0.60)	0.55 ± 0.14 (0.52)
	sResNet4	0.47 ± 0.06 (0.52)	0.63 ± 0.04 (0.66)	0.62 ± 0.04 (0.65)
	ViT	0.41 ± 0.05 (0.42)	0.58 ± 0.03 (0.59)	0.54 ± 0.04 (0.54)
Task 5 <i>per-disease</i> <i>(6 classes)</i>	LightOCT	0.33 ± 0.08 (0.36)	0.43 ± 0.07 (0.46)	0.42 ± 0.06 (0.45)
	ResNet50	0.31 ± 0.67 (0.32)	0.41 ± 0.05 (0.42)	0.39 ± 0.07 (0.40)
	Mk-CNN	0.40 ± 0.03 (0.51)	0.47 ± 0.02 (0.58)	0.45 ± 0.04 (0.56)
	sResNet4	0.34 ± 0.08 (0.36)	0.43 ± 0.07 (0.44)	0.37 ± 0.08 (0.36)
	ViT	0.38 ± 0.08 (0.46)	0.47 ± 0.07 (0.54)	0.45 ± 0.06 (0.53)

Figura 23. Resultados de los modelos entrenados con el conjunto de datos 2D (MCC, accuracy y f1-score).

Fuente: Tampu et al. (2023). *Diseased thyroid tissue classification in OCT images using deep learning: Towards surgical decision support.*

AUC [0,1] m ± SD (ensemble)	Precision [0,1] m ± SD (ensemble)	Recall [0,1] m ± SD (ensemble)
0.74 ± 0.09 (0.84)	0.72 ± 0.09 (0.84)	0.58 ± 0.03 (0.55)
0.75 ± 0.10 (0.88)	0.74 ± 0.10 (0.88)	0.61 ± 0.06 (0.58)
0.70 ± 0.15 (0.79)	0.70 ± 0.12 (0.76)	0.51 ± 0.02 (0.50)
0.80 ± 0.04 (0.85)	0.77 ± 0.04 (0.83)	0.68 ± 0.03 (0.69)
0.74 ± 0.02 (0.75)	0.73 ± 0.02 (0.74)	0.68 ± 0.03 (0.70)
0.76 ± 0.05 (0.81)	0.61 ± 0.08 (0.68)	0.57 ± 0.07 (0.59)
0.84 ± 0.06 (0.88)	0.74 ± 0.08 (0.79)	0.64 ± 0.08 (0.67)
0.83 ± 0.04 (0.87)	0.68 ± 0.06 (0.73)	0.62 ± 0.10 (0.60)
0.82 ± 0.04 (0.86)	0.70 ± 0.06 (0.77)	0.63 ± 0.04 (0.66)
0.78 ± 0.02 (0.80)	0.59 ± 0.03 (0.60)	0.58 ± 0.03 (0.59)
0.81 ± 0.05 (0.85)	0.50 ± 0.08 (0.53)	0.43 ± 0.07 (0.46)
0.79 ± 0.06 (0.84)	0.46 ± 0.09 (0.54)	0.41 ± 0.05 (0.42)
0.86 ± 0.03 (0.91)	0.59 ± 0.05 (0.68)	0.47 ± 0.02 (0.58)
0.83 ± 0.07 (0.87)	0.51 ± 0.10 (0.58)	0.43 ± 0.07 (0.44)
0.82 ± 0.06 (0.86)	0.53 ± 0.08 (0.59)	0.47 ± 0.07 (0.54)

Figura 24. Resultados de los modelos entrenados con el conjunto de datos 2D (AUC, precision y recall).

Fuente: Tampu et al. (2023). *Diseased thyroid tissue classification in OCT images using deep learning: Towards surgical decision support.*

3D data	Model	MCC [-1,1]	Accuracy [0,1]	F1-score [0,1]
		2D m ± SD (2D ensemble) 3D ensemble	2D m ± SD (2D ensemble) 3D ensemble	2D m ± SD (2D ensemble) 3D ensemble
Normal vs. abnormal (2 classes)	LightOCT	0.26 ± 0.07 (0.21) 0.63	0.58 ± 0.03 (0.55) 0.81	0.49 ± 0.05 (0.44) 0.81
	ViT	0.37 ± 0.06 (0.41) 0.79	0.68 ± 0.03 (0.70) 0.90	0.68 ± 0.03 (0.70) 0.90
Structure-based (3 classes)	LightOCT	0.36 ± 0.10 (0.40) 0.56	0.57 ± 0.07 (0.59) 0.69	0.55 ± 0.07 (0.58) 0.65
	ViT	0.41 ± 0.05 (0.42) 0.49	0.58 ± 0.03 (0.59) 0.65	0.54 ± 0.04 (0.54) 0.66
Per-disease (6 classes)	LightOCT	0.33 ± 0.08 (0.36) 0.35	0.43 ± 0.07 (0.46) 0.44	0.42 ± 0.06 (0.45) 0.40
	ViT	0.38 ± 0.08 (0.46) 0.58	0.47 ± 0.07 (0.54) 0.62	0.45 ± 0.06 (0.53) 0.61

Figura 25. Resultados de los modelos entrenados con el conjunto de datos 3D (MCC, accuracy y f1-score).

Fuente: Tampu et al. (2023). *Diseased thyroid tissue classification in OCT images using deep learning: Towards surgical decision support.*

AUC [0,1]	Precision [0,1]	Recall [0,1]
2D m ± SD (2D ensemble)	2D m ± SD (2D ensemble)	2D m ± SD (2D ensemble)
3D ensemble	3D ensemble	3D ensemble
0.74 ± 0.09 (0.84) 0.93	0.72 ± 0.09 (0.84) 0.94	0.58 ± 0.03 (0.55) 0.81
0.74 ± 0.02 (0.75) 0.97	0.73 ± 0.02 (0.74) 0.97	0.68 ± 0.03 (0.70) 0.90
0.76 ± 0.05 (0.81) 0.84	0.61 ± 0.08 (0.68) 0.72	0.57 ± 0.07 (0.59) 0.69
0.78 ± 0.02 (0.80) 0.77	0.59 ± 0.03 (0.60) 0.63	0.58 ± 0.03 (0.59) 0.65
0.81 ± 0.05 (0.85) 0.80	0.50 ± 0.08 (0.53) 0.48	0.43 ± 0.07 (0.46) 0.44
0.82 ± 0.06 (0.86) 0.94	0.53 ± 0.08 (0.59) 0.80	0.47 ± 0.07 (0.54) 0.62

Figura 26. Resultados de los modelos entrenados con el conjunto de datos 3D (AUC, precision y recall).

Fuente: Tampu et al. (2023). *Diseased thyroid tissue classification in OCT images using deep learning: Towards surgical decision support.*

Según JERBI et al. (2023) el análisis de imágenes médicas ha tomado gran importancia las últimas décadas en los procesos de diagnóstico de enfermedades. Sin embargo, la complejidad y variedad de este tipo de imágenes conlleva a buscar nuevos métodos de análisis.

Específicamente las tareas de diagnóstico y clasificación de imágenes médicas de tiro-

des conlleva grandes retos que requiere nuevas formas de análisis.

Existen distintos tipos de imágenes médicas para la tiroides. Se tiene a las imágenes de resonancia magnética (MRI) que usa las ondas de radio para crear las imágenes, las tomografías computarizadas (CT) que usan rayos X, tomografías por emisión de positrones (PET) que emplea una serie de fármacos radioactivo o trazadores además de una máquina de escaneo, finalmente, se tiene a las imágenes de ultrasonido, los cuales usan ondas sonoras de alta frecuencia. Este último es considerado como el proveedor de mayor información (estructura, forma y cantidad) de los nódulos en la glándula de la tiroides.

La decisión final en los diagnósticos de este tipo de casos es subjetiva. Esto debido a que se depende de la propia experiencia de los médicos y su mismo entorno.

Es por esto que se halla la necesidad de usar tecnologías, como las basadas en inteligencia artificial, que ayuden a los médicos a realizar diagnósticos más acertados. Por esto, se menciona que existen varios tipos de software aprobados y usados en varias situaciones médicas. Uno de los de mayor desempeño son los sistema de diagnósticos asistido por computadora (CAD), unos de los software con inteligencia artificial más usados. Este tipo de programas de computadora son capaces de analizar imágenes médicas y otorgar un diagnóstico capaz de ayudar en la toma de decisiones.

A pesar de la capacidad de los actuales sistemas CAD capaces de realizar tareas de clasificación de imágenes médicas, hay una necesidad de mejores métodos y técnicas capaces de procesar grandes cantidades de datos.

En ese contexto, los últimos años se han ido desarrollando modelos basados en redes neuronales convolucionales, como VGG16, EfficientNet y ResNet50, capaces de manejar grandes cantidades de datos.

El rendimiento de este tipo de modelos también se ve beneficiado por técnicas de aumento de datos donde se generan nuevas imágenes a través de; por ejemplo, su rotación, cambio de saturación o realce. Sin embargo, también existen técnicas más avanzadas de aumento de datos como el uso de Deep Convolutional Generative Adversarial Networks (DCGAN) que tiene la capacidad de generar imágenes a través de tomar como entrada algunas imágenes reales. Este tipo de procesos tienen el objetivo de solucionar el desbalance o baja cantidad de datos.

Se menciona que otra técnica que ha ido tomando gran importancia estos años son los Vision Transformers (ViT), basadas en los modelos transformer originales diseñadas para tareas de Procesamiento de Lenguaje Natural (NLP).

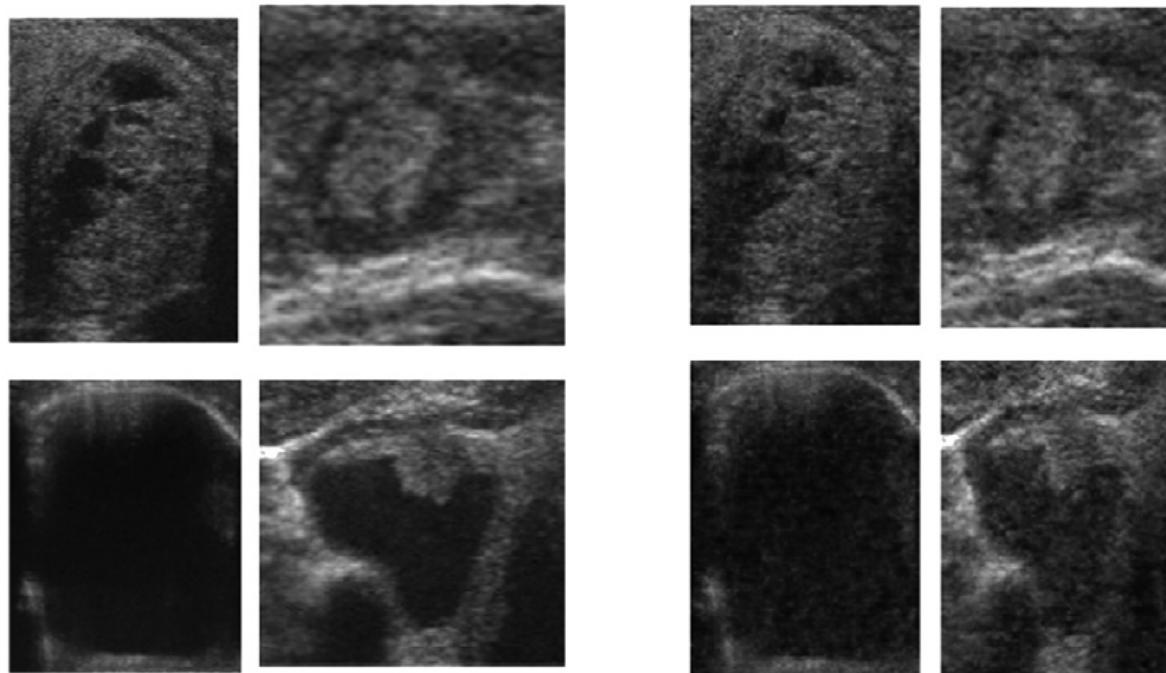
En esta investigación se desarrolló un modelo híbrido entre ViT y CNN con el objetivo

de obtener una arquitectura capaz de otorgar predicciones más acertadas que los convencionales modelo de Deep Learning en la tareas de clasificación de nódulos tiroideos en la glándula de tiroides a través de imágenes de ultrasonido.

El conjunto de datos usado fue el de CIM@LAB de la Universidad Nacional de Colombia y el Instituto de Diagnóstico Médico. Este consiste en 472 imágenes de ultrasonido de la glándula tiroides donde 397 eran de carácter maligno, y 75 de carácter benigno. Las imágenes poseen distintos tamaños, por ello se procedió con su redimensionamiento a 128 x 128 píxeles.

Debido a la baja cantidad de muestras del conjunto de datos, se realizó un aumento de datos a través del modelo DCGAN, el cual generó 10 000 imágenes, repartidas homogéneamente en cada clase.

Algunas imágenes generadas y originales se muestran a continuación.



Source Images:
CIM@LAB

Generated Images:
DCGAN

Figura 27. Imágenes de ultrasonido de la tiroides (originales y generadas).

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

La arquitectura del modelo generador consta de una entrada que es ruido generado aleatoriamente a través de una distribución normal estándar, luego se pasa a través de seis convoluciones 2D transpuestas (fractionally-strided convolution) con tamaño de kernel 4 x 4 con el fin de hacer up-sampling hasta llegar al tamaño de 128 x 128. Todas las capas convolucionales excepto la primera tienen un stride de 2. La primera tiene el stride definido en 1 pixel. Finalmente, se usa ReLU en todas las capas convolucionales a excepción del a última.

La parte de la arquitectura encargada de discriminar tiene una entrada de tamaño 3 x 128 x 128. Este parte del modelo también incluye 6 capas de convolución 2D con un kernel de 4 x 4 y un stride de 2 para todas la capas a excepción de la capa final. Se usó Leaky ReLU con pendiente negativa de 0.2 como función de activación en todas la capas sin contar la capa final que se definió la función sigmoide.

Para la división del conjunto de datos, se usaron dos métodos distintos. El primero consistía en dividir el toda la data en 80 % para el entrenamiento (4 000 imágenes de nódulos benignos y 4 000 imágenes de nódulos malignos), 10% (500 benignos y 500 malignos) para la validación y 10% (500 benignos y 500 malignos) para la prueba. El segundo método fue el cross-validation con k-fold de tamaño de 10; es decir, se tomaron 9 000 imágenes para el entrenamiento y 1 000 para la prueba.

Las redes neuronales convolucionales VGG16, EfficientNetB0 y ResNet50 se probaron con el conjunto de datos (dividido por los dos métodos mencionados) utilizando transfer learning. En la parte del clasificador se usó Softmax y SVM con un kernel igual a l2. Esta parte sirvió para definir cuál modelo CNN será usado para la implementación de la arquitectura híbrida.

La arquitectura de ViT usada fue el ViTB16. Este modelo recibe patches de 16 x 16. Es decir, las imágenes de ultrasonido se dividirán, de acuerdo a su tamaño, en patches de tamaño 16 x 16. Luego, cada uno de estos es aplanado y mapeado a través de una proyección lineal.

A la secuencia de embedded patches se le añade un nuevo tipo de embedding a la posición 0. Además, se añade otros embeddings 1D a cada embedded patch que indiquen sus respectivas posiciones. Esto finalmente será usado en el encoder del transformer.

El Transformer Encoder consta de un Multi-Head Self Attention (MSA) y un Perceptrón Multicapa (MLP) (dos capas con función GELU), donde la salida del MSA es alimentada al MLP.

La investigación propone que en lugar de alimentar con patches al modelo transformer, se use las salidas de un modelo CNN (ResNet50) como entrada. Esto se logra a través de pasar por un proceso de patching a los tensores resultantes del modelo CNN.

Los modelos CNN y ViT fueron entrenados con los mismos hiper-parámetros (optimizador SGD, tasa de aprendizaje de 0.0001, tamaño de lote de 32, 20 épocas e imágenes de 128 x 128).

En el caso del modelo híbrido de mayor desempeño, se usó la función de activación Softmax y la función de pérdida binary-cross entropy. El clasificador fue un modelo SVM con kernel l2, función de activación lineal y función de pérdida hinge.

Las métricas usadas para evaluar los modelos entrenados fueron el accuracy, precision, recall y f1-score.

Los resultados se muestran a continuación.

	Model	Data Split	F1_score(%)	Recall(%)	Precision(%)	Accuracy(%)
Softmax Classifier	VGG16	Classic Split	93.06	93.06	93.06	92.90
	EffecientNetB0	Classic Split	86.42	86.42	86.42	86.40
	ResNet50	Classic Split	96.67	96.67	96.67	96.60
	ViT_B16	Classic Split	92.28	92.28	92.28	92.10
	Hybrid ViT	Classic Split	97.87	97.87	97.87	97.50
	VGG16	10-Folds	93.24	93.24	93.24	93.42
	EffecientNetB0	10-Folds	87.09	87.09	87.09	87.16
	ResNet50	10-Folds	96.66	96.66	96.66	96.71
	ViT_B16	10-Folds	96.10	96.10	96.10	95.98
	Hybrid ViT	10-Folds	96.96	96.96	96.96	97.10
SVM Classifier	VGG16	Classic Split	92.78	90.82	94.96	93.69
	EffecientNetB0	Classic Split	85.38	76.85	96.42	90.79
	ResNet50	Classic Split	95.98	94.14	98.03	97.20
	ViT_B16	Classic Split	94.55	92.08	97.23	95.09
	Hybrid ViT	Classic Split	96.42	94.82	98.24	96.80
	VGG16	10-Folds	92.29	90.86	93.98	93.44
	EffecientNetB0	10-Folds	83.95	75.38	95.46	90.51
	ResNet50	10-Folds	96.46	94.96	98.16	97.32
	ViT_B16	10-Folds	96.12	95.14	97.18	96.17
	Hybrid ViT	10-Folds	96.67	95.01	98.51	97.63

Figura 28. Comparación de los modelos entrenados.

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

En relación a los modelos CNN, se pudo observar que el de mayor desempeño fue el ResNet50 con el clasificador SVM, alcanzando un accuracy de 97.63 %. Este hallazgo permitió determinar el CNN a usar en el modelo híbrido propuesto (ResNet50-ViT-B16) que obtuvo finalmente el mejor desempeño general con un accuracy de 97.63 %.

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)

En el caso de esta investigación, la principal técnica a usar dentro del Deep Learning son las Convoluciones. Esta técnica permite la extracción de características de imágenes para posteriormente realizar su clasificación con respecto a estos. En el proceso de convolución es importante mencionar a los filtros. Estos contienen pesos que son multiplicados por los valores de los pixeles de una imagen, para que finalmente se obtenga un nuevo valor de pixel (Moroney, 2020). A continuación, se presenta un ejemplo gráfico de convolución 2D en una imagen del conjunto de datos Fashion MNIST.

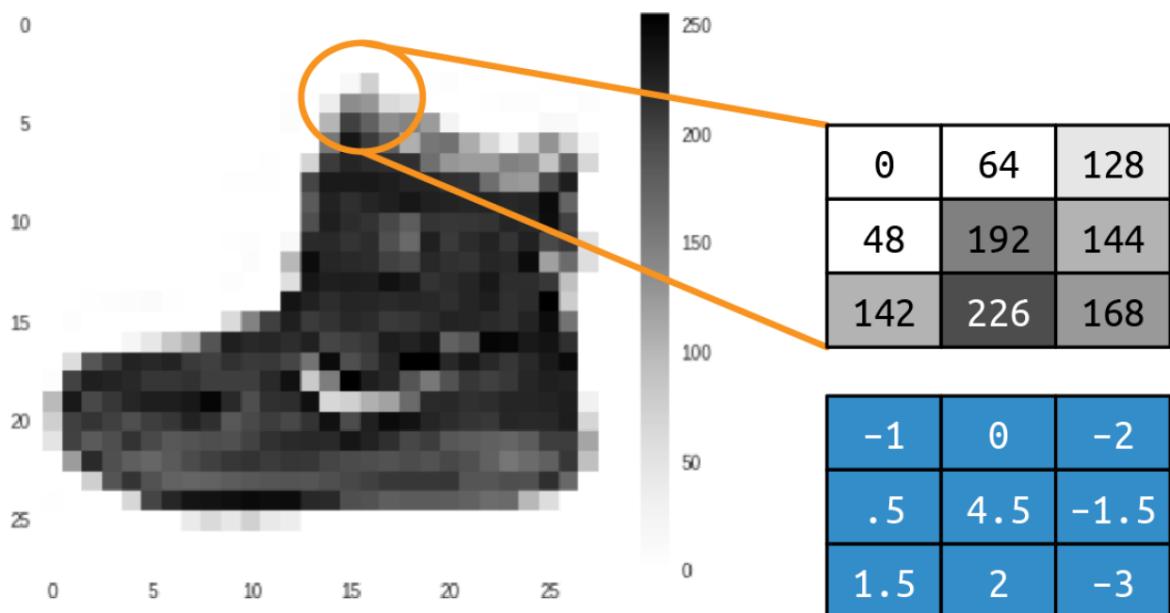


Figura 29. Convolución de imagen de Fashion MNIST.

Fuente: Moroney (2020). *AI and Machine Learning for Coders*.

Con un filtro de 3x3 como se muestra en la imagen, se puede modificar el valor del pixel. Este proceso se repite con cada pixel en imagen. Finalmente, se obtiene una imagen nueva conocida como una imagen filtrada. (Moroney, 2020) En el ejemplo anterior, el nuevo valor del pixel original 192 será 577, pues este es el resultado de multiplicar, y posteriormente sumar, los pesos del filtro con el valor del pixel y sus vecinos.

Existen distintos tipos de filtros que modifican y otorgan distintos tipos de resultados. Como ejemplo se presentan a continuación las siguientes imágenes.

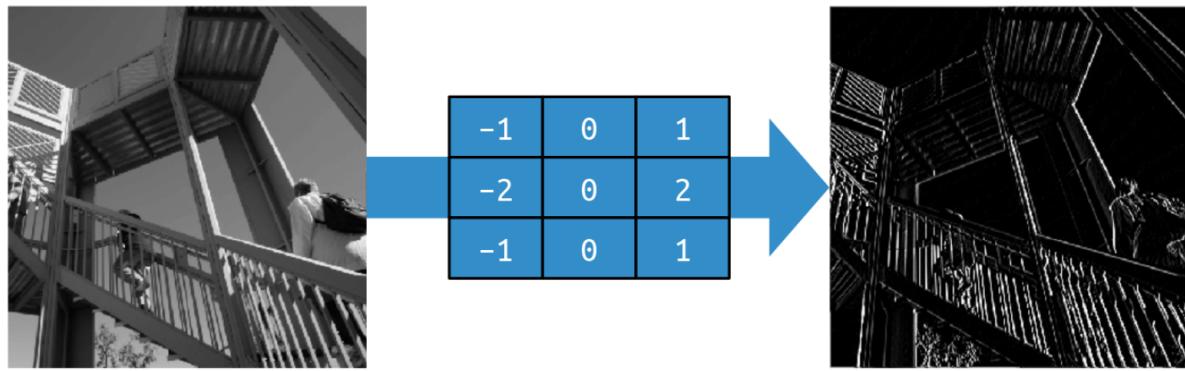


Figura 30. Convolución de imagen con filtro de líneas verticales.

Fuente: Moroney (2020). *AI and Machine Learning for Coders*.

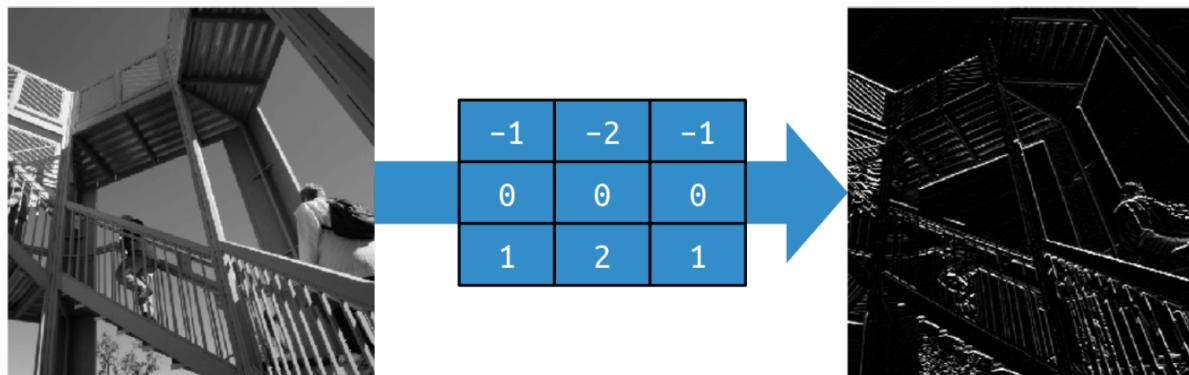


Figura 31. Convolución de imagen con filtro de líneas horizontales.

Fuente: Moroney (2020). *AI and Machine Learning for Coders*.

El primer filtro realiza modificaciones a una imagen original para finalmente obtener otra imagen distinta en donde se resaltan las líneas verticales. Caso contrario pasa con el segundo filtro, donde la imagen original es modificada para resaltar sus líneas horizontales. Así, existen distintos tipos de filtros con distintos pesos. Cada uno de estos permite resaltar en las imágenes, a través de modificaciones en sus pixeles, sus características más importantes que serán de utilidad para diferenciar entre una clase u otra de un conjunto de imágenes. También se podría decir que esta técnica de aplicar convoluciones permite reducir la cantidad de información presente en las imágenes, entonces se podría aprender o encontrar un conjunto de filtros específicos capaces de reducir la gran cantidad de información de las imágenes en

características que estén relacionadas a sus respectivas etiquetas (Moroney, 2020).

Para reducir la cantidad de información presente en las imágenes, manteniendo al mismo tiempo las características obtenidas por las convoluciones, es necesario aplicar otra técnica dentro del mundo del Deep Learning.

El proceso de pooling consiste en eliminar cierta cantidad de pixeles en una imagen, mientras se mantienen las partes resaltantes de la misma. Este proceso normalmente se realiza con agrupaciones de 2x2 en los pixeles de una imagen. Estas agrupaciones también son conocidas como pool. Dentro de cada uno de estos se selecciona el máximo valor de pixel. El proceso se repite con nuevos grupos de pixeles para finalmente obtener una nueva imagen de tamaño considerablemente reducido. (Moroney, 2020) A continuación se presenta un ejemplo gráfico.

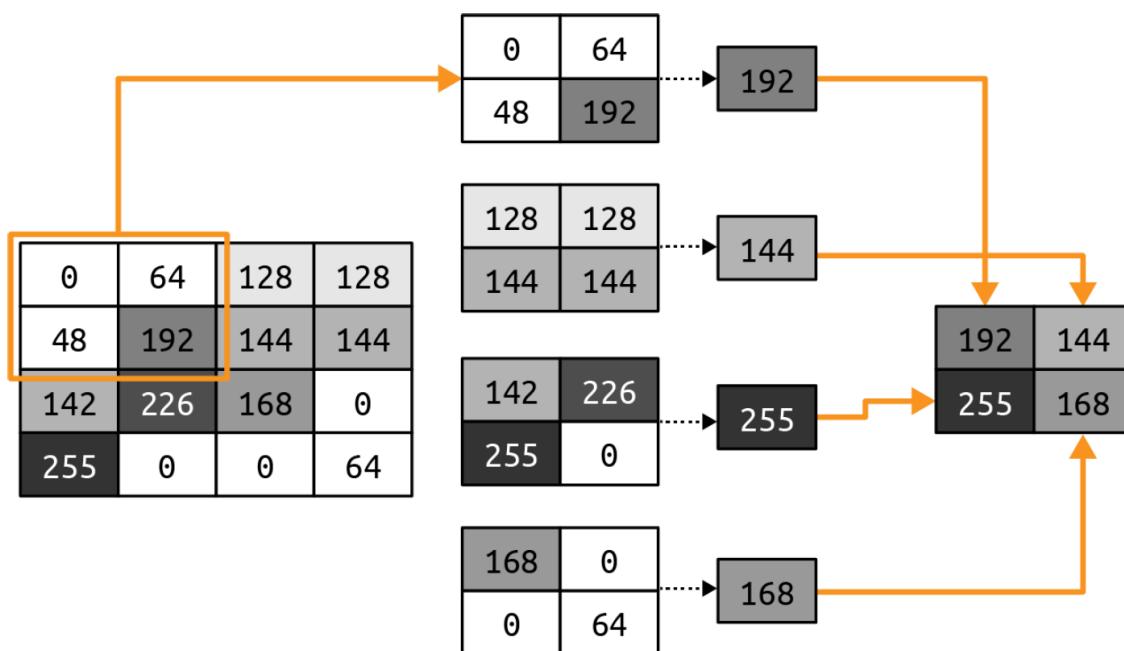


Figura 32. Ejemplo de max-pooling con un pool de 2x2.

Fuente: Moroney (2020). *AI and Machine Learning for Coders*.

La imagen original de 4x4 pixeles, luego de ser aplicado el proceso de pooling, se obtiene una imagen reducida de 2x2 pixeles. De manera general, la imagen redujo a la cuarta parte de la cantidad original de pixeles.

Con estas dos técnicas del Deep Learning se han ido desarrollando distintas arquitecturas de redes neurales en la última década. Algunas más complejas y con más capas que otras. Cada una de estas han logrado desempeñarse de forma satisfactoria con el conjunto de datos con el que se han entrenado, demostrando el gran potencial de las redes neuronales convolucionales.

o CNN.

Algunas de las arquitecturas más conocidas y que han sido usadas en gran variedad de investigaciones son los VGG, ResNet, Inception y DenseNet.

La arquitectura VGG o VGGNet es una red neuronal artificial con una profundidad de 16 o 19 capas (dependiendo de la versión que se analice) y usa pequeños filtros de convolución de tamaño 3x3 a través de estas. Este modelo participó en el ImageNet Challenge 2014, donde logró el primer puesto en las tareas de localización y segundo puesto en clasificación. Además, tiene la alta capacidad de generalización en otros conjuntos de datos, es decir, es capaz de obtener buenos resultados con imágenes distintas a las usadas para su entrenamiento. (Simonyan & Zisserman, 2015)

Otra arquitectura bastante difundida en el mundo del Deep Learning y las CNN es ResNet. Esta arquitectura nace de la premisa del aumento de la dificultad de entrenar las redes neuronales profundas. Con el objetivo de facilitar este proceso se añade a las ya conocidas CNN el concepto de residual, para obtener redes residuales que sean mucho más fáciles de optimizar y capaces de obtener mejor desempeño a más grande sea la profundidad de la red. El modelo tuvo 152 capas, ocho veces más grande que las arquitecturas VGG, y fue evaluada con el conjunto de datos ImageNet. (He et al., 2016)

Inception es una arquitectura que obtuvo un alto desempeño en las tareas de clasificación y detección en el ImageNet Large-Scale Visual Recognition Challenge 2014 (ILSVRC14). La principal distinción de este modelo es el uso de los recursos informáticos a través de la red. Esto significa que mientras más aumente la complejidad del modelo a través de su profundidad, los recursos computacionales no variarán a gran escala como se esperaría. Esto se logra a través del uso de módulos de Inception en algunas capas de la arquitectura en general. Los distintos módulos se encuentran apilados unos sobre otros y con algunas capas max-pooling. Dentro de esta arquitectura, se tiene a GoogLeNet, que no es más que una versión de Inception de 22 capas (este modelo fue el presentado a ILSVRC14). (Szegedy et al., 2015)

DenseNet nace con la premisa de que las CNN pueden obtener mejores desempeños si se tienen conexiones cortas entre las capas cerca de la entrada y salida de la red. La propuesta de esta red consiste en establecer una conexión de cada capa con todas las demás capas de la misma red. Esto trae grandes ventajas como la reducción del problema de gradiente de fuga, fomenta la reutilización de características extraídas en previas capas, además de disminuir la cantidad de parámetros. El modelo fue evaluado con los conjuntos de datos CIFAR-10, CIFAR-100, SVHN e ImageNet para la tarea de reconocimiento de objetos. (Huang et al., 2017) A continuación, se muestra una representación de DenseNet con 5 capas.

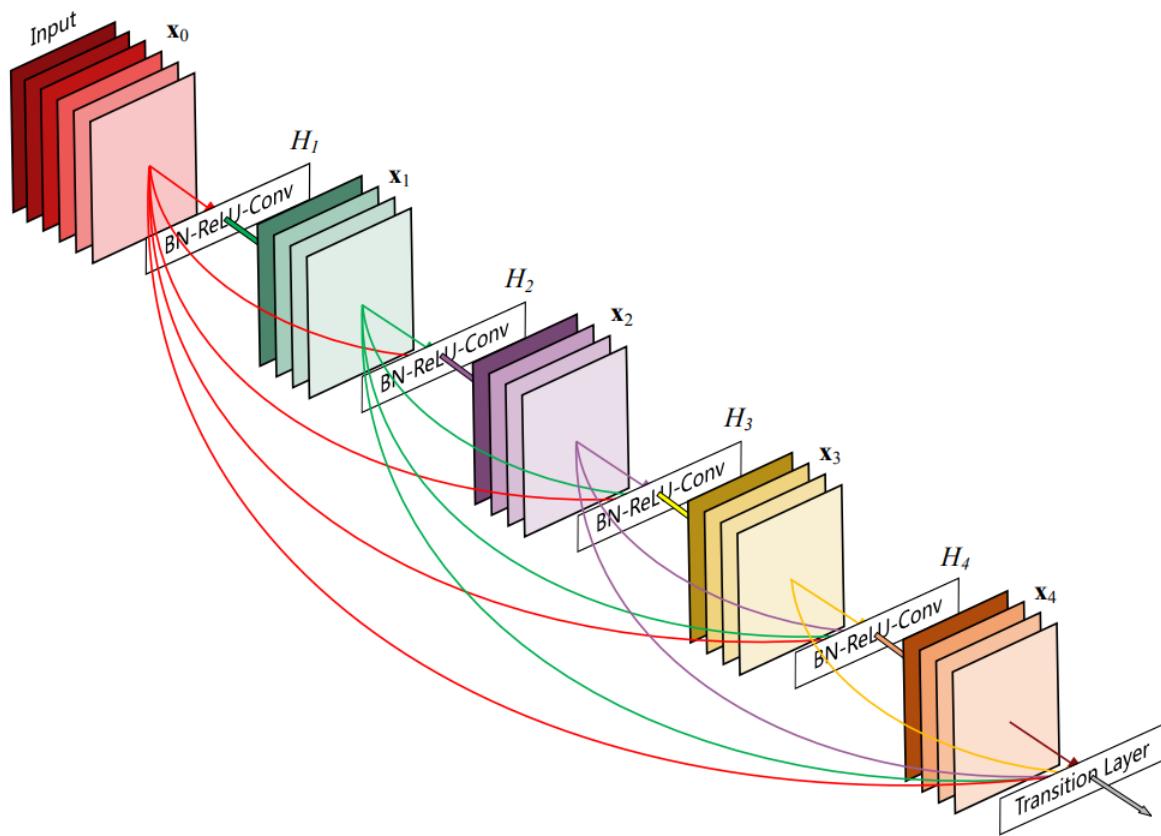


Figura 33. DenseNet con 5 capas.

Fuente: Huang et al. (2017). *Densely Connected Convolutional Networks*.

Las tareas relacionadas al procesamiento de imágenes como la segmentación, detección de objetos y clasificación es actualmente dominada por las redes neuronales convolucionales explicadas anteriormente; sin embargo, gracias al gran avance y divulgación de los modelos basados en Transformers, se ha visto el surgimiento de un nueva clase de arquitecturas que prometen mejorar el desempeño de los actuales modelos dominantes: los Vision Transformer (ViT).

Según la investigación original presentada por Dosovitskiy et al. (2021), los modelos Transformer, dominantes actuales en tareas de procesamiento de lenguaje natural (NLP), se han convertido en el modelo predilecto cuando se trata de manejo de texto, esto debido a su capacidad de utilizar los recursos computacionales de forma eficiente y su escalabilidad, permitiendo así obtener modelos de; por ejemplo, 100 mil millones de parámetros.

A diferencia de las tareas de NLP, la visión por computadora tuvo mayores problemas para adaptar este nuevo tipo de arquitecturas a sus tareas específicas.

Los primeros intentos se basaban en usar la capacidad de self-attention de los Transformer y combinarlo con los dominantes modelos CNN. Estos intentos de obtener los excelentes resultados de los Transformer en las tareas de visión por computadora no obtenían los resultados de los modelos de mayor desempeño como son los basados en la arquitectura de ResNet.

Debido a esto y al gran potencial que tenían las arquitecturas basadas en Transformers, propusieron el uso de estas arquitecturas aplicadas de forma directa a imágenes; es decir, aplicando la menor cantidad de cambios antes de ingresar al mismo Transformer. El proceso exacto de cómo funciona su arquitectura se explicará más adelante.

La gran ventaja de este nuevo tipo de modelos se basa en la cantidad de datos con la que es entrenado.

En el caso de usar una pequeña cantidad de imágenes para el entrenamiento se obtienen resultados no favorables, llegando a tener un bajo rendimiento comparado a los modelos basados en CNN. Sin embargo, al realizar su entrenamiento con una mayor cantidad de datos, los Vision Transformer (como se le nombró a este nuevo tipo de arquitectura) obtuvieron mejores resultados que los modelos dominantes en las mismas tareas de visión por computadora.

A diferencia de los Transformer dedica a las tareas de NLP que reciben como entrada token embeddings de una sola dimensión, las imágenes son de dos dimensiones (H , W) y una cantidad de canales (C), lo que dificulta su ingreso directo a este tipo de arquitecturas. Es por esto que es necesario aplicar una división de las imágenes en patches 2D que posteriormente serán aplanados e ingresados al Transformer como haría normalmente los token embeddings.

Estos patches deben tener un tamaño constante P , es decir que se deben obtener divisiones de las imágenes de tamaño $P \times P$. Esto conlleva a que el número de patches final sea igual a $N = HW/P^2$.

Una vez se tienen los N patches, se realiza su aplanamiento y, posteriormente, se mapan en D dimensiones (tamaño del vector latente usado en el Transformer) a través de una proyección lineal capaz de ser entrenada.

Ya obtenidos estos patch embedding, se añade uno nuevo pero con la capacidad de ser aprendible. Estos se agrupan en una secuencia que ingresarán posteriormente al Transformer encoder. Cabe resaltar que, luego de pasar por el encoder, este último embedding añadido representará a la imagen.

A los ya mencionados patch embedding también se le agrega otro tipo de embedding capaz de mantener información relacionada a la posición de cada patch. Estos embedding son 1D y aprendibles. Finalmente, estos serán ingresados juntos con los originales embeddings a

Transformer encoder.

En la parte final de la arquitectura se tiene un MLP con la principal tarea de realizar la clasificación de las imágenes. Este MLP consta de una sola capa oculta en la etapa de entrenamiento inicial del modelo, y con una sola capa lineal en la etapa de fine-tuning.

El bloque de Transformer encoder está compuesto por distintos bloques como los multi headed self attention (MSA), MLP y los LayerNorm (LN). Estos distintos bloques se van alternando de forma específica, introduciendo además conexiones residuales luego del MSA y MLP.

Toda la arquitectura de los ViT se puede resumir con la siguiente imagen.

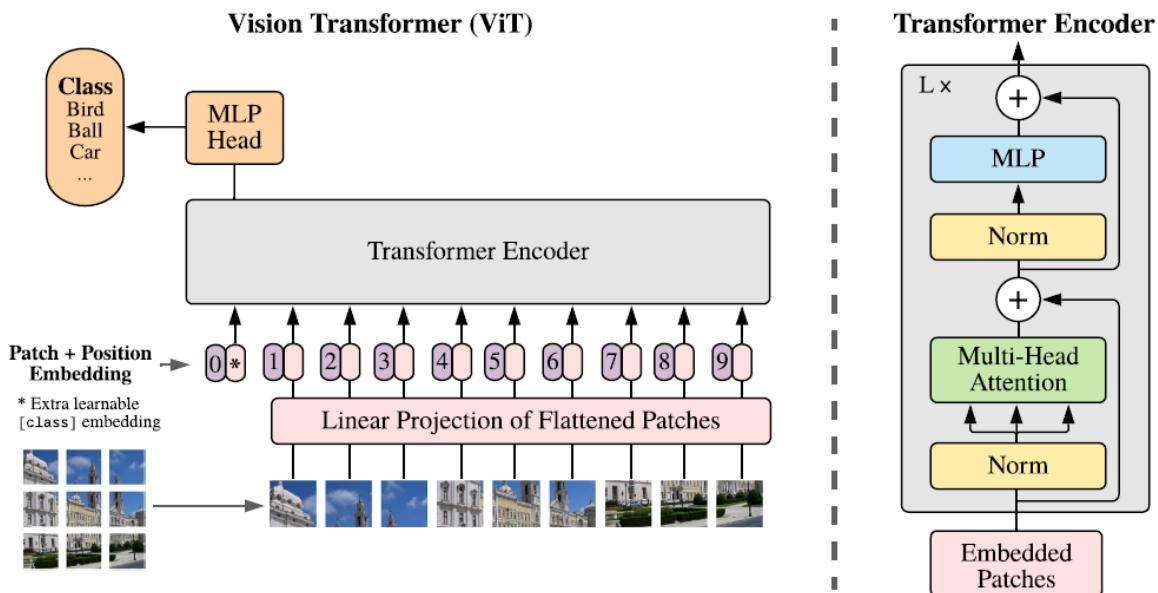


Figura 34. Arquitectura del modelo ViT.

Fuente: Dosovitskiy et al. (2021). *AN IMAGE IS WORTH 16X16 WORDS: TRANSFORMERS FOR IMAGE RECOGNITION AT SCALE.*

2.2.2 Nódulos Tiroideos

Para entender qué es un nódulo tiroideo, primero se debe saber qué es la tiroides. Esta es una glándula localizada en el cuello del cuerpo humano que se encarga de producir hormonas que regulan el metabolismo. Los nódulos que aparecen en esta glándula son los más comunes, y se generan debido a un excesivo crecimiento de las células en esa área, estos pueden ser de textura dura o quísticas. (Deng et al., 2022)

La siguiente figura muestra a la glándula en una imagen tomográfica.

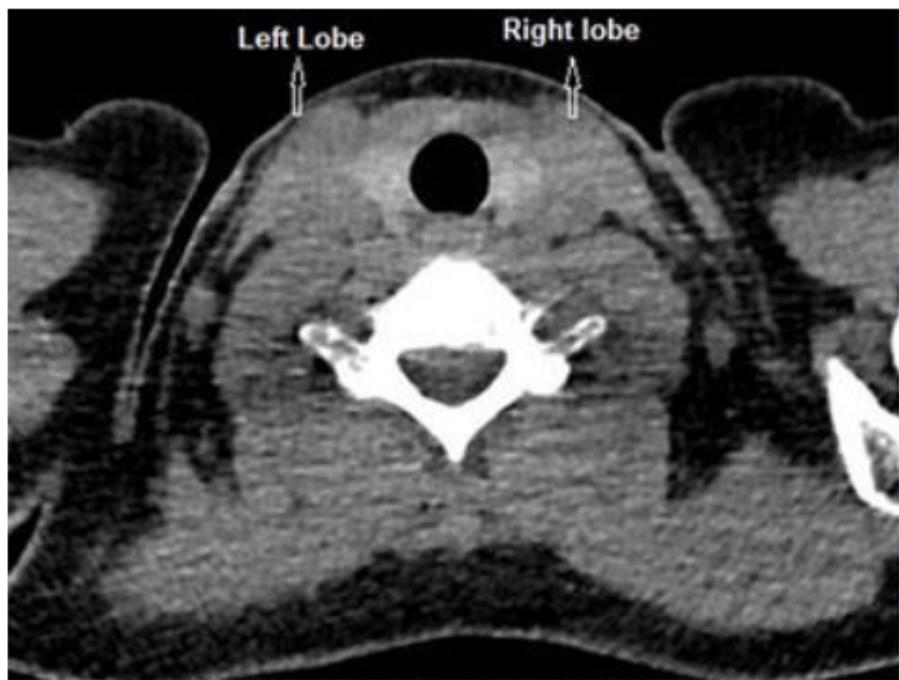


Figura 35. Imagen tomográfica de glándula de tiroides.

Fuente: Binboga et al. (2019). *Thyroid Anatomy*.

Shin et al. (2016) mencionan que existen algunas características de los nódulos en la tiroides que pueden ser analizados para realizar una clasificación si es de carácter benigno o maligno. A continuación, se presentan aquellos más resaltantes.

El tamaño de nódulos en ciertos casos puede determinar el tipo con el que se está tratando; sin embargo, esto no está debidamente comprobado, a pesar de que normalmente el riesgo de ser maligno de un nódulo es más alto en aquellos de mayor tamaño. Esto principalmente se debe a que los nódulos benignos también pueden llegar a un tamaño considerable, pero con una mayor cantidad de tiempo. Lo que sí puede significar un probable nódulo maligno es un rápido crecimiento de un nódulo sólido.

El contenido interno de un nódulo está determinado por el ratio entre su parte quística y su parte sólida. Los nódulos malignos con contenido sólido poseen mayor riesgo de ser malignos que aquellos con contenido parcialmente quístico.

La ecogenicidad de los nódulos se mide de acuerdo al nivel de brillo en las imágenes de ultrasonido con respecto a otras partes de la imagen. Esto quiere decir que un nódulo es hipoecoico si su nivel de brillo en la imagen es menor con respecto a otra parte de la imagen, específicamente se compara con el músculo anterior del cuello o el parénquima tiroideo. La

mayoría de los tumores tiroideos son hipoeicos, y existe mayor riesgo de que los nódulos con la misma característica sean malignos.

La forma y orientación de los nódulos también pueden determinar si son o no malignos. La manera en que los nódulos malignos crecen es de manera centrífuga y a través del plano del tejido, al igual que los benignos; sin embargo, estos crecen de manera paralela. La forma redonda u ovoide de los nódulos se encuentra en aquellos de carácter benigno; sin embargo, no son propias de este. Una característica mucho más específica de los nódulos de carácter malignos es su orientación no paralela, es decir, son más altos que anchos.

Finalmente, el margen de los nódulos también puede dar información del tipo con el que se trata. Una característica que sugiere que un nódulo es maligno es un margen espiculado o micro tubulado.

2.3 Marco Conceptual

2.3.1 Inteligencia Artificial

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)

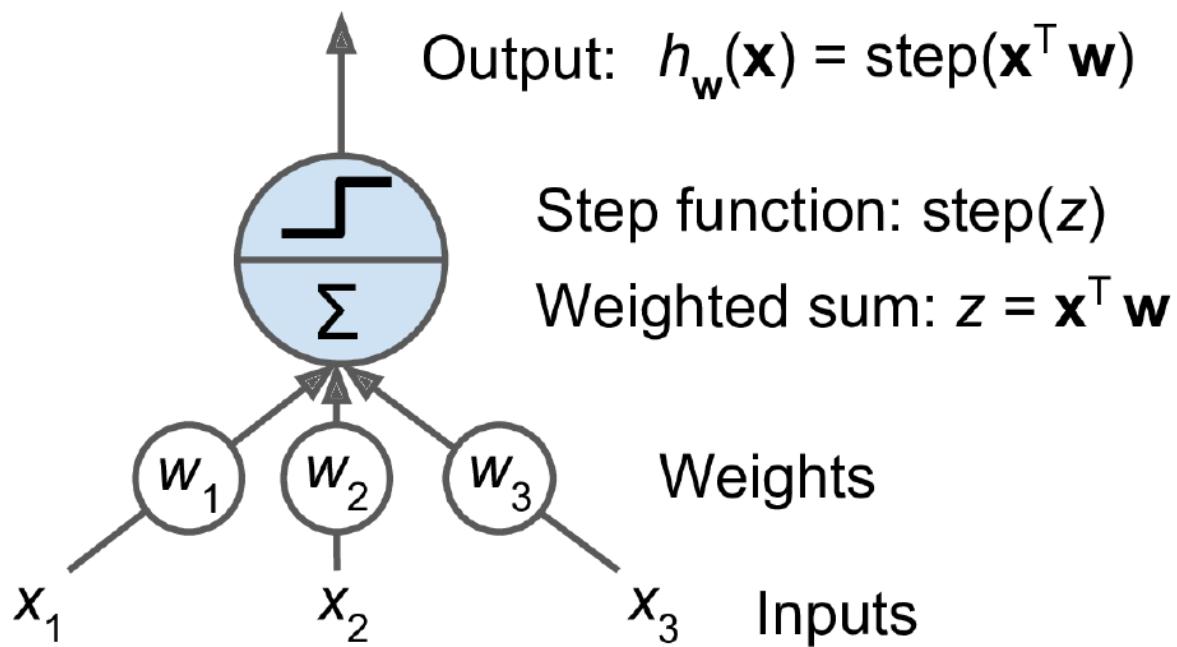


Figura 36. Perceptrón.

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)

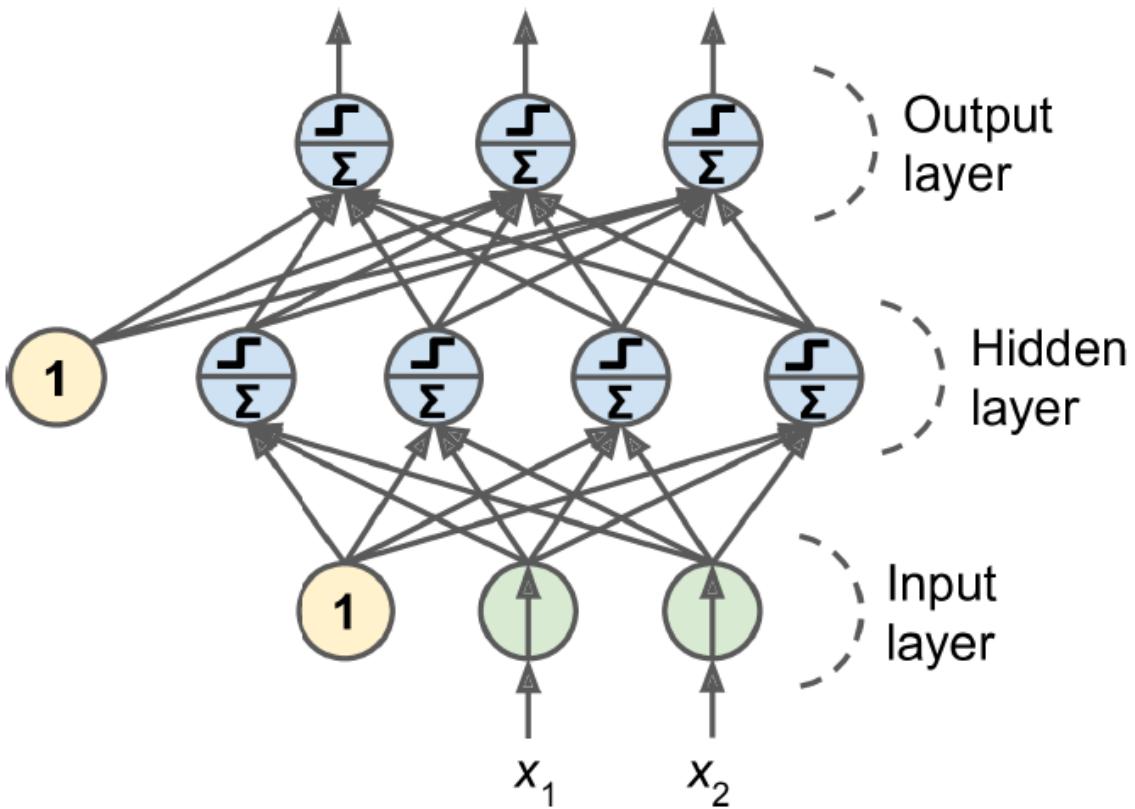


Figura 37. Perceptrón multicapa.

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)

A continuación, se presenta el enfoque tradicional del Machine Learning.

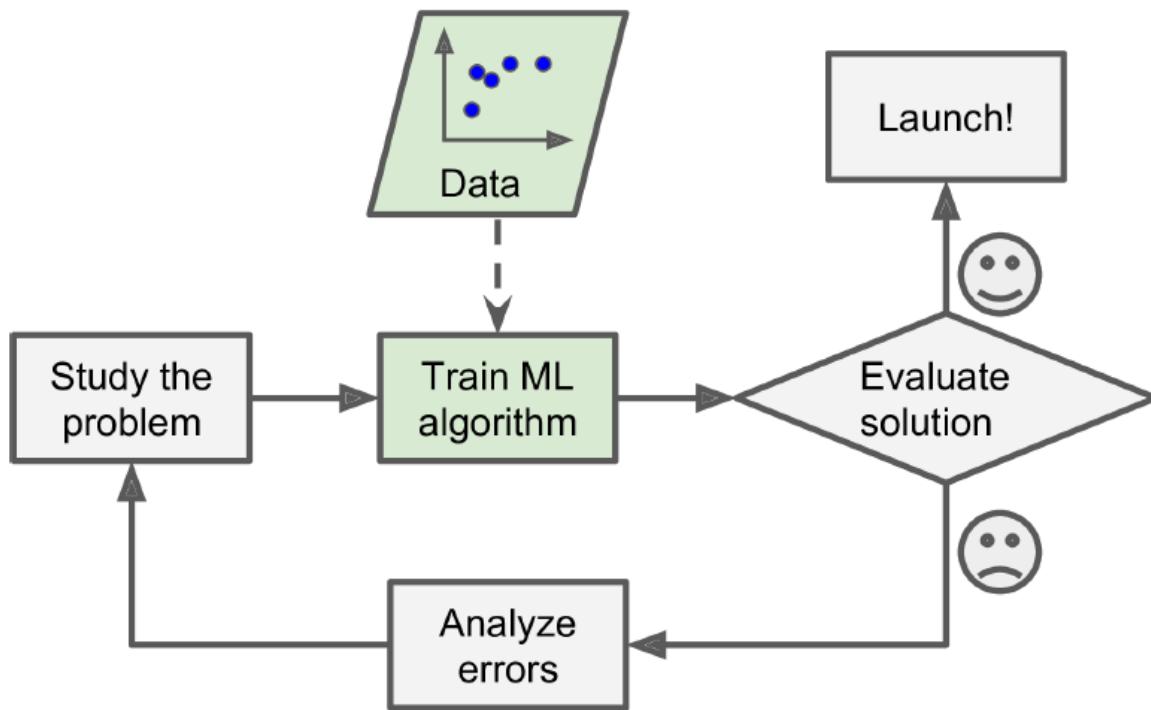


Figura 38. Enfoque del Machine Learning.

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

2.3.5 Ecografía y las imágenes de ultrasonido

Según Herrera Gajardo (2017), la ecografía, que es una técnica de diagnóstico en donde se usan imágenes generadas por ultrasonido, es comúnmente desarrollado en las áreas de cardiología, ginecología, y otras más relacionadas. La popularidad de esta técnica se basa en la capacidad de las imágenes de alta calidad que se obtienen de este proceso, además de no ser un método invasivo o de radiación como muchos otros de su tipo.

2.4 Hipótesis

2.4.1 Hipótesis General

HG: A través del diseño de un modelo de Deep Learning se logra diagnosticar los nódulos tiroideos a través de imágenes de ultrasonido.

2.4.2 Hipótesis Específicas

- HE1: Las mejores características del conjunto de datos son usadas para entrenar y evaluar modelos 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 tiene un diseño experimental porque se pretende establecer la relación entre modelos y técnicas de Deep Learning con el 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 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 del sistema 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. A continuación, se presenta de forma gráfica la metodología a seguir.

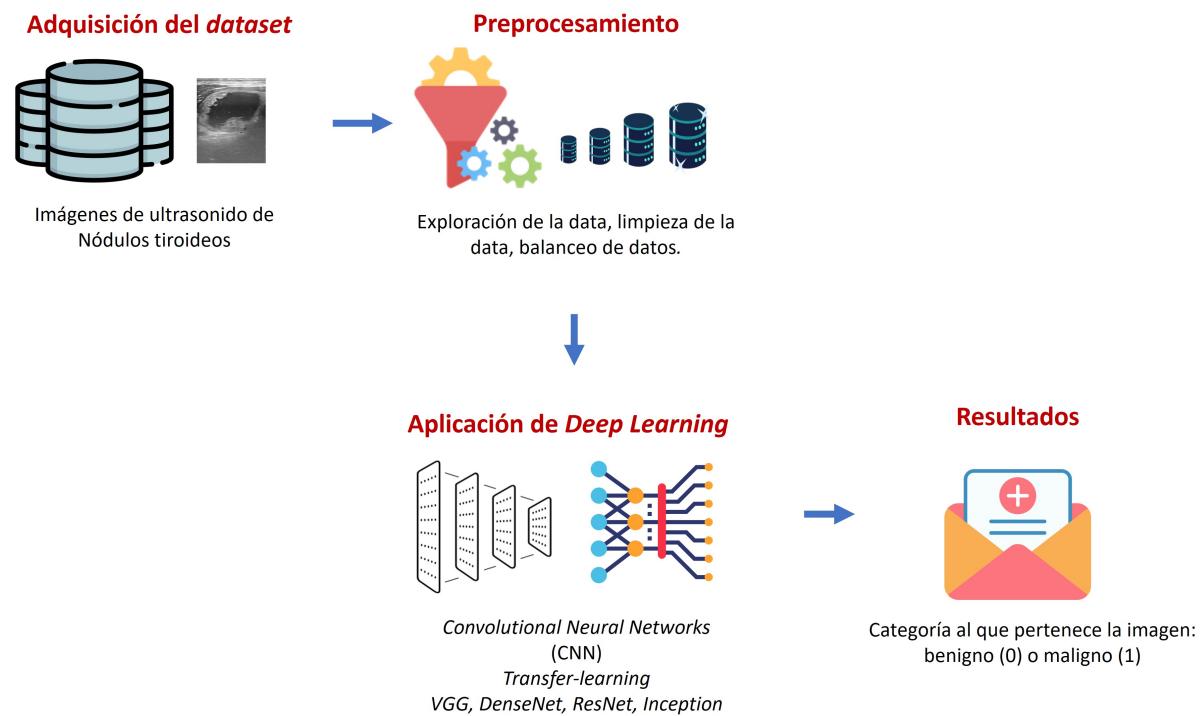


Figura 39. Metodología de implementación.

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 preprocessamiento, 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 o weighting en la situación de desbalanceo de datos, con el fin de evitar una baja generalización y sobreajuste en la clase mayoritaria. Finalmente, se aplicará escalado 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 preprocessada, este se usará como entrada para los modelos de Deep Learning. Principalmente, 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, RestNet, Inception y DenseNet, pues son ideales para la extracción de características de las imágenes, facilitando así el proceso final de clasificación.

Cada uno de los modelos serán probados en parte 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.

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

Figura 40. 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.

$$\text{accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

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

$$\text{recall} = \frac{TP}{TP + FN} \quad (2)$$

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

$$precision = \frac{TP}{TP + FP} \quad (3)$$

3.4 Cronograma de actividades y presupuesto

Se propuso un cronograma para la investigación. Este conforma desde el inicio hasta ser terminada con la sustentación final planeada para mediados del año 2024.

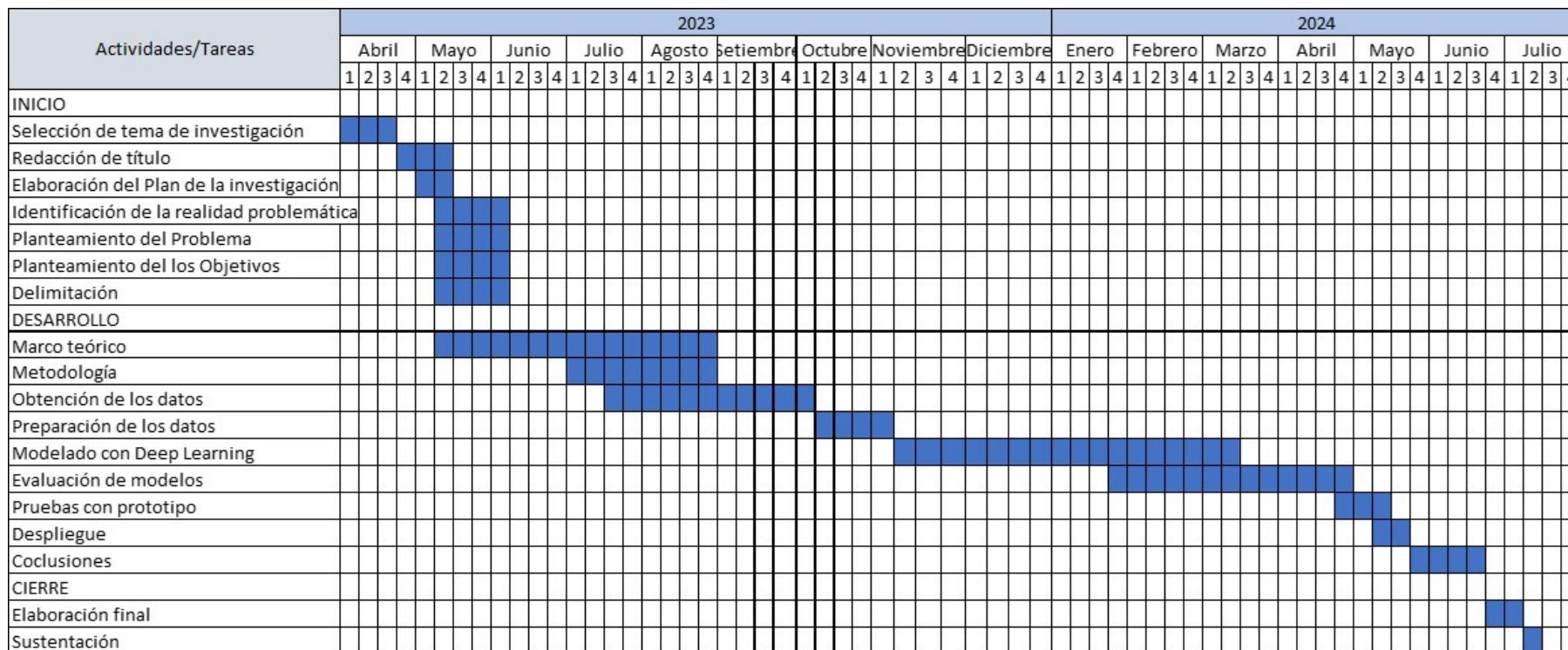


Figura 41. Cronograma de actividades.

Fuente: Elaboración propia.

Además, se determinó el presupuesto necesario para la elaboración completa de la investigación.

Item	Costo (soles)	Subtotal
Recursos materiales		
Laptop	S/ 6.500,00	S/ 6.500,00
Materiales de escritorio		
Software y trámites		
Reserva de tema	S/ 2.700,00	
Derecho de inscripción	S/ 800,00	
Derecho de sustentación	S/ 1.500,00	
Software	S/ 0,00	S/ 5.000,00
Extras		
Consultorías	S/ 100,00	
Movilidad	S/ 200,00	S/ 300,00
Total		S/ 11.800,00

Figura 42. Presupuesto.

Fuente: Elaboración propia.

Capítulo IV: Desarrollo del Experimento

En este capítulo se detalla el proceso explicado en el capítulo anterior para cada actividad de la metodología aplicada, así como los entregables comprometidas.

4.1 Comprensión del negocio

Actividad 1: Definir problemas, objetivos e hipótesis

El inicio de la implementación de la primera fase de la metodología CRISP-DM fue la identificación del problema general a partir del estudio de la realidad problemática abordada en los primeros capítulos. El objetivo, por lo tanto, busca resolver el problema y se encuentra alineado con el título de la investigación. Asimismo, la hipótesis resulta la proposición planteada para explicar el logro del objetivo. Los objetivos específicos del trabajo, que responde a cada problema específico, se definieron a partir de una lluvia de ideas en base a la asociación de objetivos específicos de los antecedentes con la actual investigación. Estos se explican en la siguiente actividad.

Actividad 2: Desarrollar la literatura de la investigación

Se buscaron desde libros y artículos online para la comprensión teórica del financiamiento colectivo e Inteligencia Artificial, hasta papers publicados en conferencias, revistas internacionales y científicas, reportes técnicos y tesis de grado acerca de propuestas para resolver el problema estudiado en la investigación.

Para ello, como se describió en la sección 3.2, a través de búsqueda de palabras clave como *crowdfunding*, *Machine Learning*, *Deep Learning*, *prediction*, *Kickstarter*, *accuracy* y *projects*, y el uso del buscador Google Académico, se encontraron estos papers publicados entre el 2013 y 2020.

A continuación, se realizó un resumen de cada antecedente en una hoja de cálculo de Excel con el fin de comparar sus objetivos y metodologías implementadas, como se puede observar el detalle en el Anexo E.

Actividad 3: Definir metodología de la investigación

Luego de elaborar el detalle del anterior anexo, a excepción de la investigación del autor Fernandez-Blanco et al. (2020) que utilizó la metodología CRISP-DM, cada antecedente fue agrupado con otro similar de acuerdo a los pasos seguidos en su propia metodología. El resultado fue la Tabla ?? explicada en la sección 3.3.1.

4.2 Comprensión de los datos

Actividad 1: Construir base de datos de Metainformación

El punto de partida para la construcción de los conjuntos de datos que se usaron más adelante en cada modelo de acuerdo a su modalidad fue la adquisición de bases de datos capturadas mensualmente desde finales del 2015 hasta 2019 por la página Web Robots (<https://webrOBots.io/kickstarter-datasets/>), fundada por los ex corporativos de TI Tomás Vitulskis y Paulius Jonaitis, como se aprecia en la Figura 43 Web Robots (2019). Para el presente trabajo, se optó por descargar en archivos de valores separados por comas (.csv). De acuerdo sus creadores, se ejecutan robots en dos servidores en la nube encargados de recolectar en un determinado punto del día y una vez al mes información de las campañas que aparecen en Kickstarter.

Kickstarter Datasets

We have a scraper robot which crawls all [Kickstarter](#) projects and collects data in CSV and JSON formats. From March 2016 we run this data crawl once a month. Datasets are available from the following scrape dates:

2019

- 2019-08-15 [[JSON](#)] – [[CSV](#)]
- 2019-07-18 [[JSON](#)] – [[CSV](#)]
- 2019-06-13 [[JSON](#)] – [[CSV](#)]
- 2019-05-16 [[JSON](#)] – [[CSV](#)]
- 2019-04-18 [[JSON](#)] – [[CSV](#)]
- 2019-03-14 [[JSON](#)] – [[CSV](#)]
- 2019-02-14 [[JSON](#)] – [[CSV](#)]
- 2019-01-17 [[JSON](#)] – [[CSV](#)]

2018

- 2018-12-13 [[JSON](#)] – [[CSV](#)]
- 2018-11-15 [[JSON](#)] – [[CSV](#)]
- 2018-10-18 [[JSON](#)] – [[CSV](#)]
- 2018-09-13 [[JSON](#)] – [[CSV](#)]
- 2018-08-16 [[JSON](#)] – [[CSV](#)]
- 2018-07-12 [[JSON](#)] – [[CSV](#)]
- 2018-06-14 [[JSON](#)] – [[CSV](#)]
- 2018-05-17 [[JSON](#)] – [[CSV](#)]
- 2018-04-12 [[JSON](#)] – [[CSV](#)]
- 2018-03-15 [[JSON](#)] – [[CSV](#)]
- 2018-02-15 [[JSON](#)] – [[CSV](#)]
- 2018-01-12 [[JSON](#)] – [[CSV](#)]

Figura 43. Vista del website Web Robots (visitado en agosto del 2019).

Fuente: Elaboración propia.

A continuación, los archivos descargados que se encontraban fraccionados en varios archivos .csv, donde luego de ser descomprimidos, fueron unidos por mes de captura y almacenados en carpetas independientes por mes, cuyo peso individual osciló entre 1 y 5 gigabytes (GB). Con el fin de ahorrar espacio en la computadora, las partes originales fueron eliminadas.

En la Figura 44 se detalla el tamaño del conjunto de datos total al corte del periodo de captura de información Julio 2019, aproximadamente más de 212 mil proyectos de todas las categorías y 37 columnas de variables.

```
In [7]: data_combinada_201907.shape    ##Originalmente habían 212,378 registros de todos los proyectos en Kickstarter
Out[7]: (212378, 37)

In [8]: data_combinada_201907.columns
Out[8]: Index(['backers_count', 'blurb', 'category', 'converted_pledged_amount',
       'country', 'created_at', 'creator', 'currency', 'currency_symbol',
       'currency_trailing_code', 'current_currency', 'deadline',
       'disable_communication', 'friends', 'fx_rate', 'goal', 'id',
       'is_backing', 'is_starrable', 'is_starred', 'launched_at', 'location',
       'name', 'permissions', 'photo', 'pledged', 'profile', 'slug',
       'source_url', 'spotlight', 'staff_pick', 'state', 'state_changed_at',
       'static_usd_rate', 'urls', 'usd_pledged', 'usd_type'],
      dtype='object')
```

Figura 44. Tamaño de conjunto de datos al corte de Julio 2019.

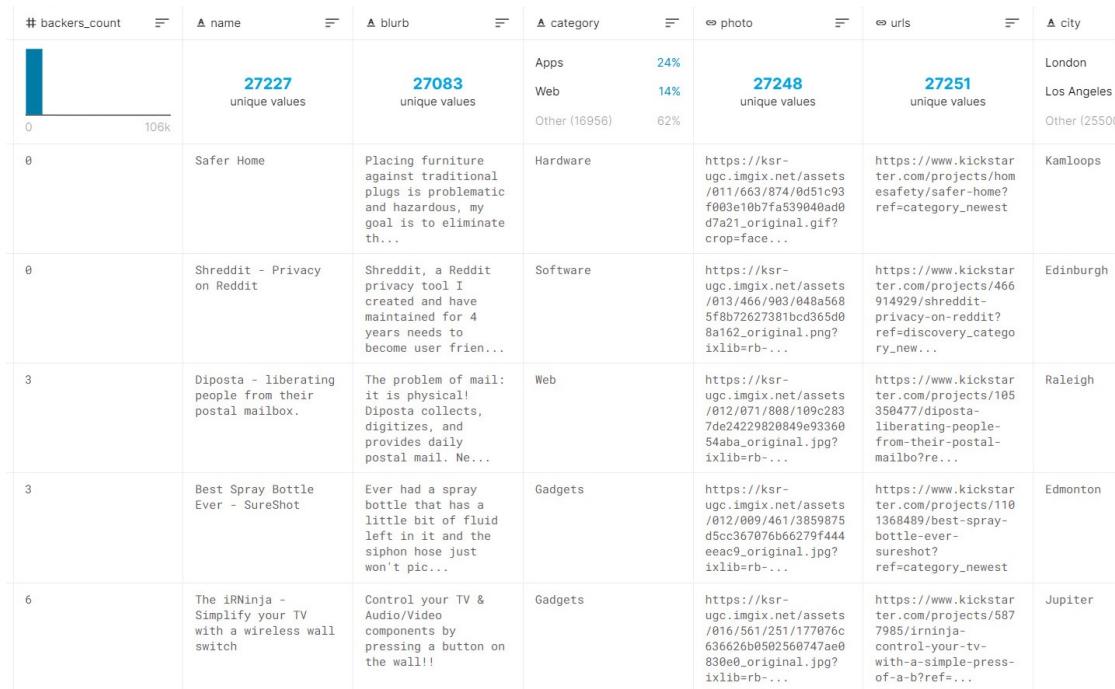
Fuente: Elaboración propia.

A cada conjunto de datos generado se filtraron los que pertenecen a la categoría **Technology**. Al no contar con la información de la columna *main_category*, este proceso se logró utilizando la variable *source_url* seleccionando aquellos registros que contengan la cadena de caracteres “<https://www.kickstarter.com/discover/categories/technology>”.

Cuando se repitió este procedimiento con cada conjunto generado, se observó que la proporción de proyectos tecnológicos en Kickstarter representa el 10% de la totalidad, aproximadamente más de 21 mil registros por mes. Esto se calculó al comparar el tamaño de cada conjunto generado con el total.

A continuación, se unieron los 45 archivos separados por coma (.csv) capturados mensualmente desde noviembre del 2015 hasta agosto del 2019. Se realizaron 2 uniones internamente ya que, a partir de marzo del 2018, algunas de las variables y valores presentan diferente estructura a la de sus predecesoras.

Luego, se realizó limpieza de datos para las variables *category*, *location*, *photo* y *urls*, y se transformaron las variables numéricas en milisegundos *created_at*, *launched_at* y *deadline* a variables de fecha. Esto último permitió calcular la variable *duration* para determinar la duración de la campaña (en días) de un proyecto calculando la diferencia entre la fecha de culminación (*deadline*) y la fecha de lanzamiento (*launched_at*). Luego se excluyeron los proyectos en proceso de la variable *state* para conservar los culminados, es decir, aquellos cuyo valor sea “successful” o “failed” ya que se analizarán solamente los proyectos que han sido exitosos o fracasados. Los proyectos cancelados o suspendidos no aparecieron. El último paso del flujo consistió en generar y exportar el archivo final en formato .csv. En la Figura 45 se visualiza el conjunto final de Metainformación subido públicamente a la plataforma Kaggle. Cada variable se detalla en la Tabla 1.

**Figura 45.** Visualización del archivo de metainformación subido a Kaggle.

Fuente: Elaboración propia.

Tabla 1
Diccionario de datos del dataset final de Metainformación.

Variable	Detalle	Tipo de dato
id	Identificador del proyecto.	number
backers_count	Número de patrocinadores de la campaña del proyecto.	number
name	Nombre del proyecto.	string
blurb	Propaganda del proyecto.	string
category	Categoría (dentro de categoría principal) del proyecto.	string
photo	Dirección de enlace de la foto del proyecto.	string
urls	Dirección de la página de la campaña del proyecto.	string
city	Ciudad del creador del proyecto.	string
country	Código de país del creador del proyecto.	string
goal	Monto de la meta de financiamiento del proyecto.	float
pledge_amounts	Montos disponibles para patrocinar la campaña.	string
pledged	Monto final patrocinado de la campaña.	float
currency	Divisa del monto final patrocinado.	string
usd_pledged	Monto final patrocinado de la campaña (en USD).	float
created_at	Fecha de creación de la campaña.	date
launched_at	Fecha de lanzamiento de la campaña.	date
deadline	Fecha de culminación de la campaña.	date
duration	Duración de la campaña (en días).	number
state	Estado de financiamiento del proyecto.	string

Fuente: Elaboración propia.

Actividad 2: Construir base de datos de Descripción

La variable *description* se obtuvo utilizando web scraping en cada proyecto gracias a la variable *urls*. Para ello, se elaboró un algoritmo usando la librería BeautifulSoup que, mediante acceso y navegación al contenido de estas páginas a través de un agente falso, se dirigió a las descripciones de los proyectos identificando las etiquetas con clase llamada “**rte_content js-full-description responsive-media**” y las almacenó en un vector vacío, uniendo previamente todos los párrafos y eliminando caracteres especiales, para posteriormente asignarle la id de su proyecto y guardarla en un archivo de extensión .csv y exportarlo. En caso el algoritmo no encuentre esta clase dentro de las páginas (*IndexError*), el vector almacena con el valor “null”.

```
def getPageText(url):
    # Crear agente falso para scrapear
    ua = UserAgent()
    user_agent = ua.chrome
    # Solicitar uso de librería urllib con agente falso
    request = urllib.request.Request(url, headers = {"User-Agent":user_agent})
    # Obtener contenido de urls mediante librería urllib
    data = urllib.request.urlopen(request)
    # parse as html structured document
    bs = BeautifulSoup(data, "html.parser")
    # Buscar todos los tags div con una determinada clase
    # A partir de acá, se usa un "try" y un "except" porque hay páginas que no muestran su contenido
    # Para evitar que salga el mensaje de error, se llenarán como null los contenidos que no se puedan descargar
    # El try contiene el algoritmo para descargar la descripción de un proyecto
    try:
        description = bs.find_all("div", {"class":"rte_content js-full-description responsive-media"})
        # Encontrar todos los párrafos
        description = description[0].find_all("p")
        # Crear array vacío donde se almacenará el contenido descargado
        project_description = []
        # Iteración para agregando en lista cada descripción descargada
        for link in description:
            project_description.append(link.text)
        # Juntar párrafos separados en un solo vector, separándolos con un espacio
        project_description = ' '.join(project_description)
        # Eliminar caracteres especiales
        project_description = project_description.replace(u'\xa0', u' ')
        project_description = project_description.replace(u'\n', u' ')
        # Y el except sirve para llenar valores nulos en el array en caso de error
    except (IndexError, ValueError):
        project_description = 'null'
    # Eliminar saltos de línea
    return Newlines.sub('\n', project_description)
```

Figura 46. Función para extraer textos de modalidad de descripción.

Fuente: Elaboración propia.

Debido a la gran cantidad de memoria y tiempo que iba a presentar este proceso, se determinó fraccionar los 27,251 proyectos en tres partes y repetir el mismo en cada uno de ellos. El tiempo aproximado de descarga de cada fracción fue de 6 horas.

Finalmente, las tres partes fueron unidas, se reemplazaron los valores nulos por espacios en blanco y se guardó como un nuevo archivo de valores separados por coma (.csv) en código Unicode UTF-8 para la lectura de caracteres no alfabéticos, como se observa en la Figura 47.

```

# Asignar urls de búsqueda (origen)
urls_primer_parte = urls_list[0:9083]
urls_segunda_parte = urls_list[9084:18168]
urls_tercera_parte = urls_list[18169:27251]
#ids
ids_primer_parte = data_final["id"][0:9083]
ids_segunda_parte = data_final["id"][9084:18168]
ids_tercera_parte = data_final["id"][18169:27251]
description_txt = [getPageText(url) for url in urls_primer_parte]
df = {"id":ids_primer_parte, "description":description_txt}
export_csv = data_final.to_csv (r'D:\TTT\DB\dataset_descripciones\data_final_parte1.csv', index = None, header=True)
# Hacemos lo mismo con la segunda parte
description_txt = [getPageText(url) for url in urls_segunda_parte]
df = {"id":ids_segunda_parte, "description":description_txt}
data_final = pd.DataFrame(df)
export_csv = data_final.to_csv (r'D:\TTT\DB\dataset_descripciones\data_final_parte2.csv', index = None, header=True)
# Hacemos lo mismo con la tercera parte
description_txt = [getPageText(url) for url in urls_tercera_parte]
df = {"id":ids_tercera_parte, "description":description_txt}
data_final = pd.DataFrame(df)
export_csv = data_final.to_csv (r'D:\TTT\DB\dataset_descripciones\data_final_parte3.csv', index = None, header=True)
# Borramos variables para liberar memoria
del description_txt, df, data_final, export_csv
# Me redirijo a la nueva ruta de los archivos generados
os.chdir('D:/TTT/DB/dataset_descripciones')
#Las siguientes líneas son para combinar todos los archivos .csv dentro de la carpeta
extension = '.csv'
all_filenames = [i for i in glob.glob('*.format(extension)))]
#combinar todos los archivos
data_description = pd.concat([pd.read_csv(f) for f in all_filenames ])
#exportar a csv
data_description.to_csv( "data_description.csv", index=False, encoding='utf-8-sig')

```

Figura 47. Ejecución de la función de extracción de descripciones y almacenamiento.

Fuente: Elaboración propia.

El archivo generado fue subido a la plataforma Kaggle de manera pública para que pueda ser descargada a través del API de la web, como se aprecia en la Figura 48.

id	description
48.1k	[null] 2% Galen Framework is... 0% Other (2660) 98%
1000245024	Purpose Safer Home ensures that electrical plugs will not become a fire hazard. The invention comes ...
1000256230	I once had a Reddit account that was four years old with thousands of comments forever stored in Red...
1000261018	Every day you go home to a mail box filled with junk and even worse, if you are a traveler it is ove...

Figura 48. Visualización del archivo de descripción subido a Kaggle.

Fuente: Elaboración propia.

Actividad 3: Construir base de datos de Comentarios

Al igual que la descripción, los comentarios se obtuvieron utilizando la variable *urls* para web scraping, pero reemplazando caracteres que contengan desde “?ref=” en adelante, por “/comments” para redireccionarse a la sección de comentarios de cada proyecto. Para lograrlo, se codificó la función de la Figura 49.

```
def getPageText(url):
    driver = webdriver.Chrome(options=options)
    driver.set_page_load_timeout(30)
    driver.get(url)
    time.sleep(10)
    count = 0

    while (count<13):
        try:
            loadMoreButton = driver.find_element_by_xpath('//*[@id="react-project-comments"]/div/button')
            time.sleep(2)
            loadMoreButton.click()
            time.sleep(5)
            count += 1
        except (NoSuchElementException, StaleElementReferenceException, TimeoutException):
            break

        time.sleep(5)
        comments = driver.find_elements_by_css_selector('span.bg-ksr-green-700.white.px1.type-14.mr1, div.w100p')
        project_paragraphs = []
        for paragraph in comments:
            project_paragraphs.append(paragraph.text)
        idx_comment_creador = [i for i in range(len(project_paragraphs)) if project_paragraphs[i] == "Creator"]
        idx_comment_creador = [i+1 for i in idx_comment_creador]
        for index in sorted(idx_comment_creador, reverse=True):
            del project_paragraphs[index]
        idx_comment_creador = [i for i in range(len(project_paragraphs)) if project_paragraphs[i] == "Creator"]
        for index in sorted(idx_comment_creador, reverse=True):
            del project_paragraphs[index]
        project_paragraphs = [x for x in project_paragraphs if ("This comment") not in x]
        project_paragraphs = [x for x in project_paragraphs if ("0:00") not in x]
        project_paragraphs = [x for x in project_paragraphs if ("Showing ") not in x]
        project_paragraphs = [x for x in project_paragraphs if x]
        project_comments = []
        for project_text in project_paragraphs:
            project_text = project_text.replace(u'\xa0', u' ')
            project_text = project_text.replace(u'\n', u' ')
            project_text = re.sub(useless_characters, ' ', project_text)
            project_comments.append(project_text)
        del(project_paragraphs)
        driver.quit()
        time.sleep(randint(1,10))
    return project_comments
print("Scraping...")
```

Figura 49. Función para extraer textos de modalidad de comentarios.

Fuente: Elaboración propia.

Los comentarios, al ser dinámicos, no podían ser extraídos mediante la librería BeautifulSoup como en el caso de las descripciones, por lo que se utilizó la librería Selenium para extraerlos al iniciar una sesión desde Google Chrome. Una vez permitido el acceso al navegador, el algoritmo se redirecciona a la sección de comentarios del proyecto, espera un máximo de 30 segundos de carga de la página, busca el elemento Xpath “//*[@id=react-project-comments]/div/button” para hacer clic en el botón “Load More” (Cargar más) hasta un máxi-

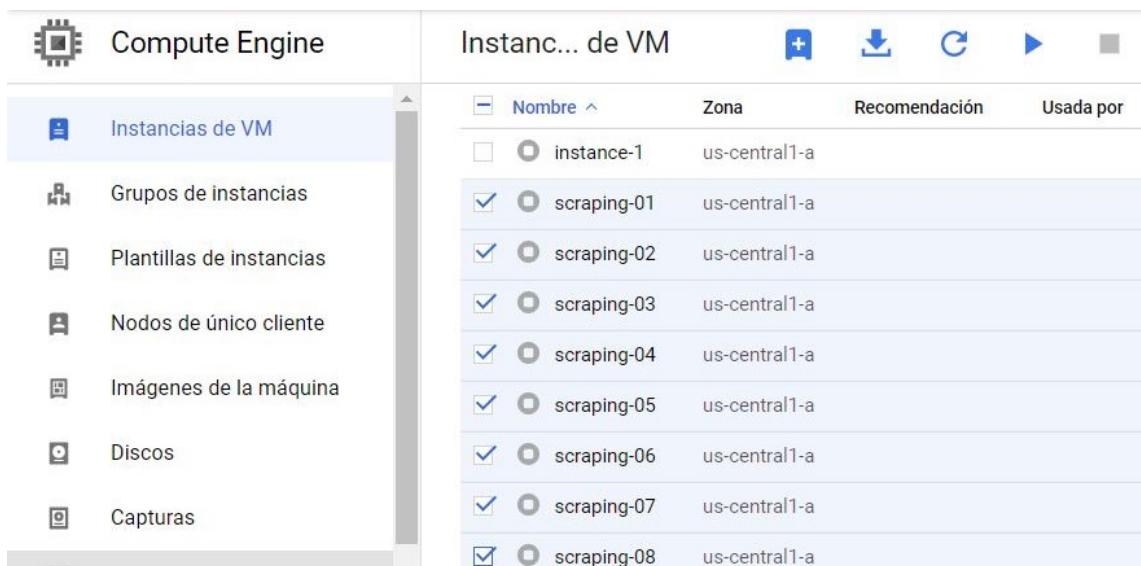
mo de 13 veces esperando 2 segundos entre cada clic, busca todos los comentarios bajo el nombre del elemento CSS “**div.w100p**”, elimina aquellos que pertenezcan al creador del proyecto identificados con etiqueta verde en el extremo superior derecho del recuadro del comentario con el nombre del elemento “**span.bg-ksr-green-700.white.px1.type-14.mr1**” (asignados con el valor de «Creator»), y almacena los restantes que pertenecen a los patrocinadores. En caso no encuentre ningún ítem de comentarios en la sección, se asignará un valor aleatorio no relacionado con proyectos. Luego de eliminar mensajes de la página acerca de comentarios ocultos o eliminados, el algoritmo culmina creando la lista de comentarios separados por autor y cerrando el web driver. Esta función fue ejecutada en 16 partes, asignando el id respectivo a los comentarios extraídos por cada proyecto en un archivo .csv, como se observa en la Figura 50.

```
count_proj=0
for id_proj, url_proj in zip(data_final["id"][:], urls_list[:]):
    comments_txt = getPageText(url_proj)
    df_comments = {"ids": [id_proj], "comments": [comments_txt]}
    data_comments = pd.DataFrame(df_comments)
    data_comments.to_csv(r'data_comentarios.csv', mode = 'a', sep = ',', index = False, header=False)
    count_proj += 1
    del data_comments, df_comments, comments_txt
    print(count_proj)
print('Ya terminó el scraping!')
```

Figura 50. Ejecución de la función de extracción de comentarios y almacenamiento.

Fuente: Elaboración propia.

Para optimizar la descarga, se crearon 8 instancias en Google Cloud (Figura 51).



The screenshot shows the Google Cloud Platform Compute Engine interface. On the left, there is a sidebar with icons for Compute Engine, VM instances, groups, templates, nodes, images, disks, and snapshots. The main area is titled 'Instanc... de VM' and displays a list of 8 instances. The table has columns for Nombre, Zona, Recomendación, and Usada por. All instances are in the 'us-central1-a' zone. The instances are: instance-1 (unchecked), scraping-01 (checked), scraping-02 (checked), scraping-03 (checked), scraping-04 (checked), scraping-05 (checked), scraping-06 (checked), scraping-07 (checked), and scraping-08 (checked). There are also icons for creating a new instance (+), deleting (-), cloning (C), and navigating (arrow).

Nombre	Zona	Recomendación	Usada por
instance-1	us-central1-a		
scraping-01	us-central1-a		
scraping-02	us-central1-a		
scraping-03	us-central1-a		
scraping-04	us-central1-a		
scraping-05	us-central1-a		
scraping-06	us-central1-a		
scraping-07	us-central1-a		
scraping-08	us-central1-a		

Figura 51. Instancias lanzadas en paralelo para la extracción de comentarios.

Fuente: Elaboración propia.

Cada instancia contenía dos copias del algoritmo, con la cantidad de proyectos fraccionada en 16 partes para que sean ejecutados en paralelo. Si bien el tiempo total de la consolidación de esta base de información duró aproximadamente un mes debido a percances de la conexión interna de las instancias y algunos problemas de ineficiencia de la primera versión del algoritmo, durante el transcurso dentro de este lapso de tiempo fueron solucionados hasta lograr optimizar el algoritmo de web scraping y tener el conjunto final de datos tomó menos de 48 horas. Este se encuentra disponible públicamente en Kaggle y se visualiza en la Figura 52.

id	comments
48.1k	71% [@Creator: What wi... 0% Other (7857) 29%
1001265769	['Waste of money ', 'Battery is Sound Quality ', "What's with the squares?", 'My battery started to...
1001502333	["First to market, last to deliver MVP. Just here to put my monthly DO NOT BUY PIMAX. Nothing but re...
1001565620	[]

Figura 52. Visualización del archivo de comentarios subido a Kaggle.

Fuente: Elaboración propia.

Actividad 4: Realizar análisis exploratorio y estadístico de variables considerados

En esta sección, se analizaron estadísticamente las variables de cada modalidad, tanto las distribuciones de sus datos para la metainformación y contenido textual, así como estadísticos para las variables cuantitativas de la metainformación, entre ellos el rango de sus valores, la media, la mediana, la moda, la desviación estándar y la varianza. En la variable dependiente estado de financiamiento, los 27,251 proyectos se distribuyen mediante el gráfico de pie de la Figura 53. Se observa que casi 20 mil proyectos entre 2009 y 2019 no llegaron a ser financiados, es decir, aproximadamente el 72 % del total fracasaron.

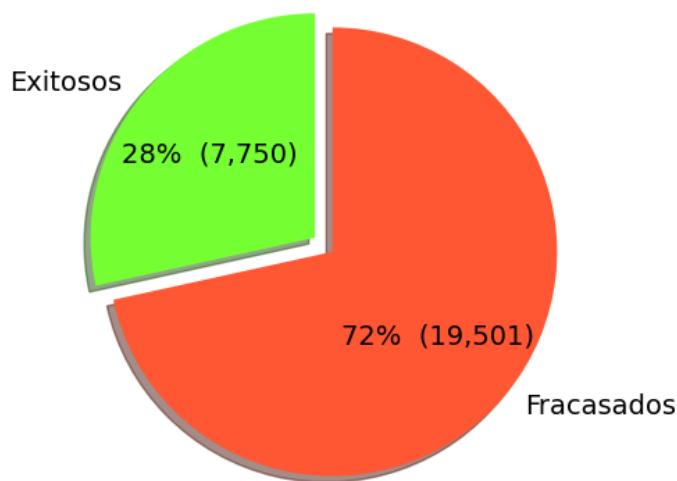


Figura 53. Distribución de proyectos tecnológicos según su estado.

Fuente: Elaboración propia.

De acuerdo a la distribución por año de la Figura 54, 2015 fue el periodo en donde se registraron más campañas de proyectos tecnológicos en la plataforma.

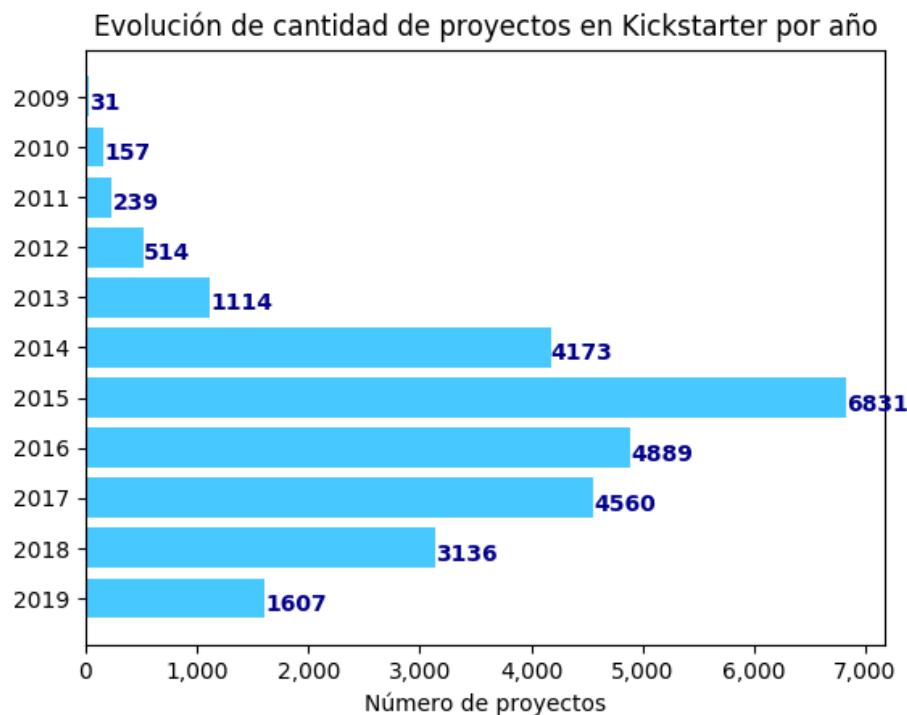


Figura 54. Evolución de cantidad de proyectos tecnológicos por año.

Fuente: Elaboración propia.

El anterior gráfico abierto por estado de financiamiento, como se representa en la Figura 55, muestra que el 2015 resultó ser el año más disparate, donde el 78 % fueron fracasados.

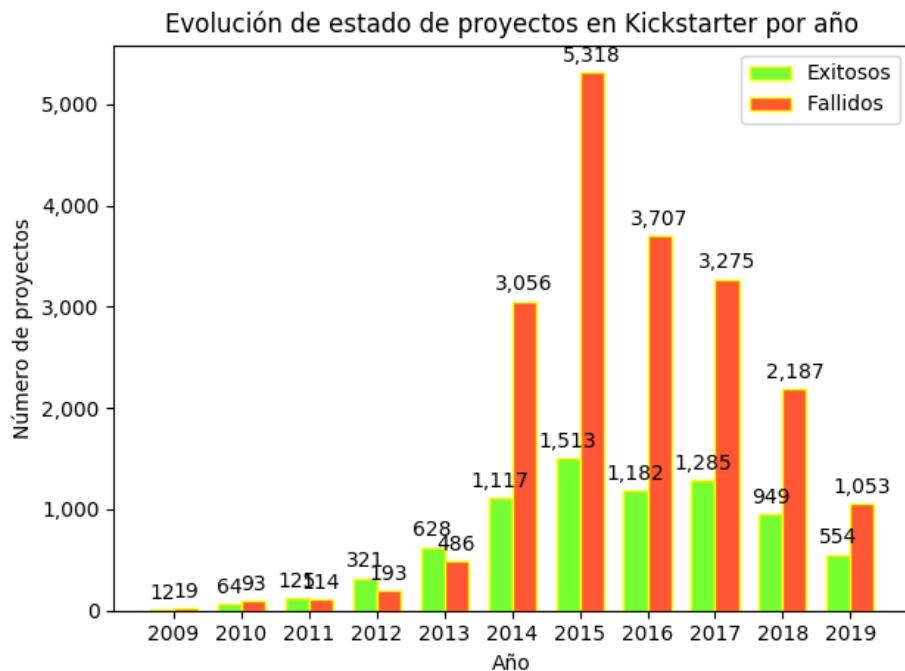


Figura 55. Evolución de proyectos tecnológicos, por su estado y año.

Fuente: Elaboración propia.

Por el lado de Metainformación, de las 19 variables de la Tabla 1, se consideraron como potenciales variables independientes a 3 categóricas, 5 numéricas y 1 lista compuesta por números (*pledge_amounts*). La distribución del primer grupo se ilustra en la Figura 56.

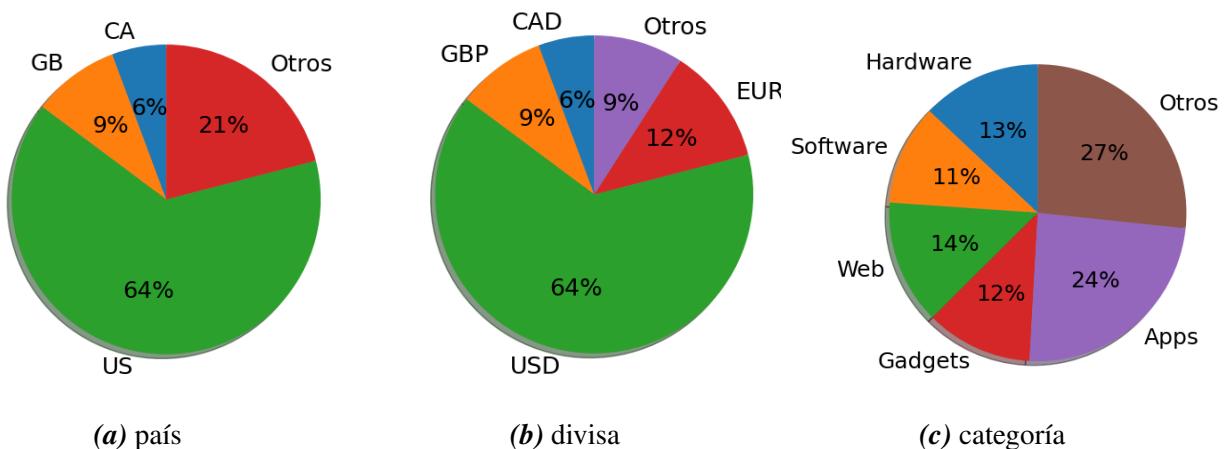


Figura 56. Distribución de las variables categóricas de Metainformación.

Fuente: Elaboración propia.

De las 3 variables categóricas (*country*, *currency* y *category*), se observa que más de la mitad de creadores proyectos provienen de los Estados Unidos (64 %) e invierten en dólares. A ellos los acompañan personas de Gran Bretaña (9 %), que invierten en libras esterlinas, y de Canadá (6 %), que invierten en dólares canadienses. El 21 % restante provienen de otros países, donde el 12 % de ellos invierten en euros. Por el lado de las categorías, las más resaltantes son Apps, Web, Hardware, Software y Gadgets.

Para las potenciales variables numéricas de Metainformación (*backers_count*, *goal*, *pledged*, *usd_pledged* y *duration*), se calcularon sus datos estadísticos de rango de valores, media, mediana, desviación estándar y varianza, con ayuda de diagramas de caja y bigote que se muestran a continuación:

- Número de patrocinadores de la campaña (*backers_count*):

- Rango de valores: [0; 105,857]
- Media: 208.710469340575
- Mediana: 9.487
- Desviación estándar: 1,179.68237749203
- Varianza: 1,391,650.51176525

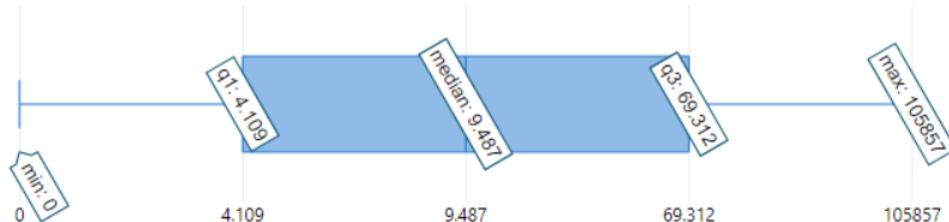


Figura 57. Diagrama de caja y bigote de patrocinadores.

Fuente: Elaboración propia.

- Monto meta de la campaña (*goal*):

- Rango de valores: [1; 100,000,000]
- Media: 91,263.9666162825
- Mediana: 15,762.614
- Desviación estándar: 1,259,282.1587922
- Varianza: 1,585,791,555,452.35

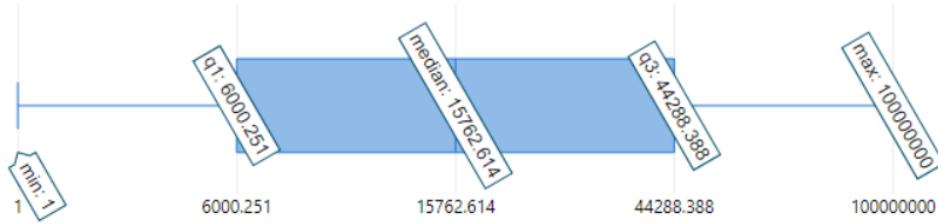


Figura 58. Diagrama de caja y bigote de meta.

Fuente: Elaboración propia.

- Monto patrocinado al final de la campaña (*pledged*):
 - Rango de valores: [0; 17,406,300]
 - Media: 34,668.5134710787
 - Mediana: 1,382.933
 - Desviación estándar: 226,763.900313481
 - Varianza: 51,421,866,485.3822

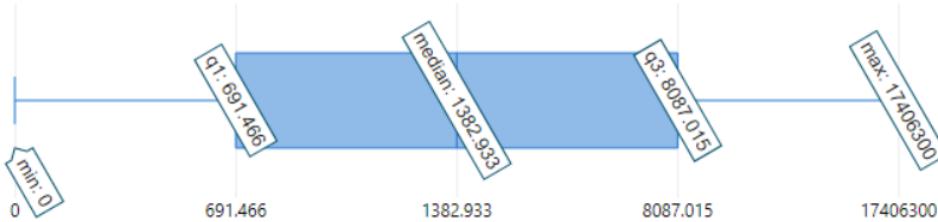


Figura 59. Diagrama de caja y bigote de monto patrocinado.

Fuente: Elaboración propia.

- Duración de la campaña (*duration*).
 - Rango de valores: [1; 92]
 - Media: 35.4654141132436
 - Mediana: 30
 - Desviación estándar: 11.84570862999998
 - Varianza: 140.320812946853

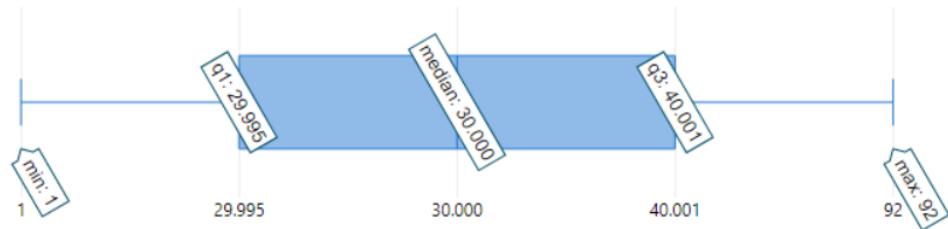


Figura 60. Diagrama de caja y bigote de duración.

Fuente: Elaboración propia.

Posterior a este entendimiento de datos, se elaboró una matriz de correlaciones (Figura 61) para encontrar correlaciones entre ellas y determinar la existencia de alguna variable redundante y descartarla para no afectar el rendimiento del modelo.

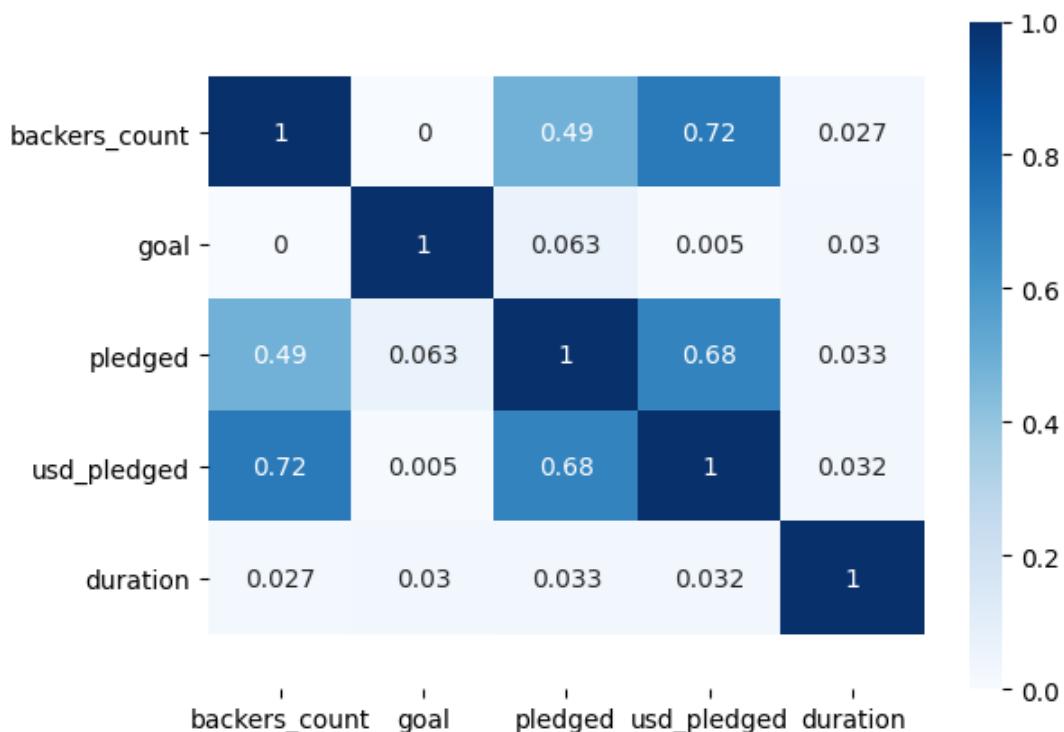


Figura 61. Matriz de correlaciones entre variables independientes.

Fuente: Elaboración propia.

Como se puede apreciar en la figura anterior, la variable *usd_pledged* está altamente correlacionada con las variables *backers_count* y *pledged* (ambas con un aproximado de 70%). Esto quiere decir que dicha variable no es significativa porque explicaría de manera muy similar a las otras dos.

Asimismo, si se observan los registros desde una matriz que contiene, además de gráficos de dispersión de las correlaciones, histogramas de las variables independientes como en la Figura 62, se confirma y concluye no utilizar las observaciones comentadas.

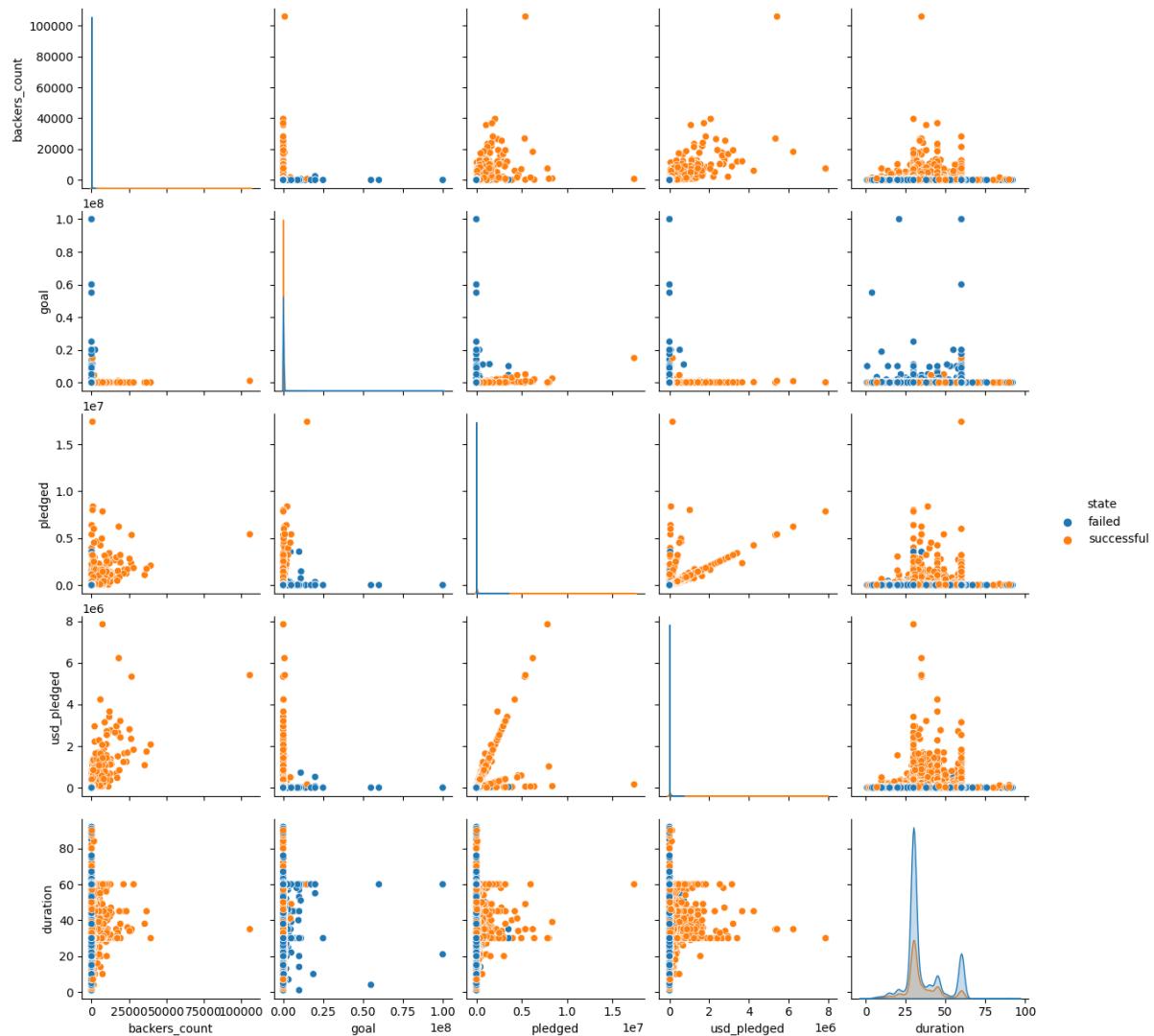


Figura 62. Gráfico de dispersión de correlaciones entre variables independientes.

Fuente: Elaboración propia.

La variable *duration* es la única que sigue una distribución normal debido a la forma de campana de su silueta. Asimismo, los registros de proyectos exitosos y fracasados para las otras 4 variables se encuentran mezcladas en los mismos grupos al cruzarse entre ellas. También se visualizan observaciones en los extremos de cada gráfico, tal y como se detalló en sus valores estadísticos individuales.

Por el lado de Descripción, solo 640 proyectos (2% del total) no presentaron descripciones por razones externas durante el proceso de extracción de datos. De ellos, 512 (80% de proyectos sin descripciones) fracasaron en ser financiados. Por el lado de proyectos con descripciones, casi el 30% fueron exitosos como se grafica en la Figura 63.

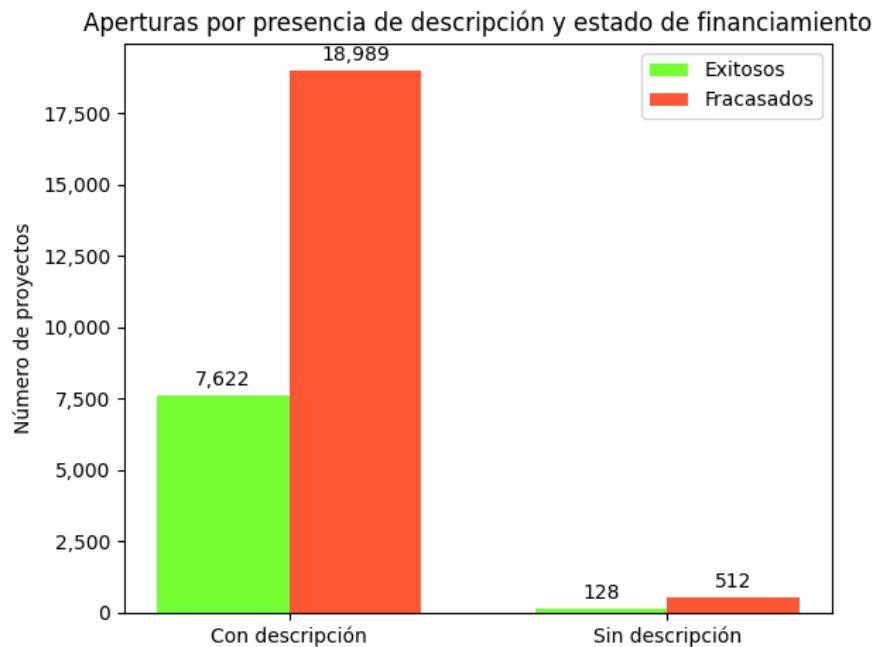


Figura 63. Distribución de proyectos por presencia de descripciones y estado final.

Fuente: Elaboración propia.

Asimismo, el registro con descripción de mayor longitud presentó 5,152 palabras y, a nivel total de proyectos, el vocabulario fue de 165,683 palabras.

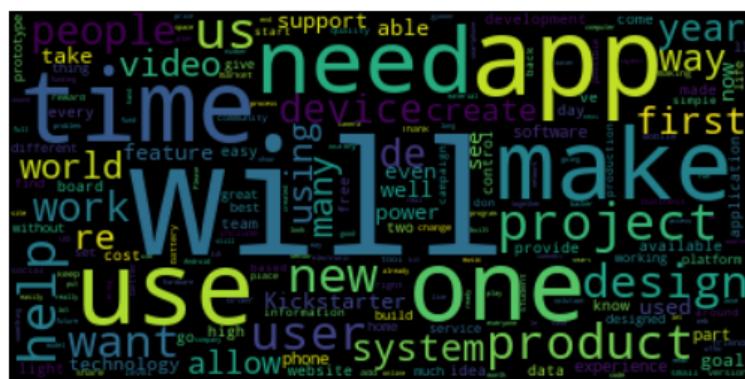


Figura 64. Nube de palabras de descripciones.

Fuente: Elaboración propia.

En la nube de palabras de la anterior Figura 64, los términos más frecuentes en ellas se relacionan con las funcionalidades del producto (*system, app, need, help, product*, etc).

Por el lado de Comentarios, al analizar los proyectos exitosos y fracasados por la presencia de comentarios (Figura 65), se observa que el 60 % de proyectos con comentarios (4,626 registros) fueron exitosos, mientras que el 84 % de los proyectos sin comentarios fracasaron.

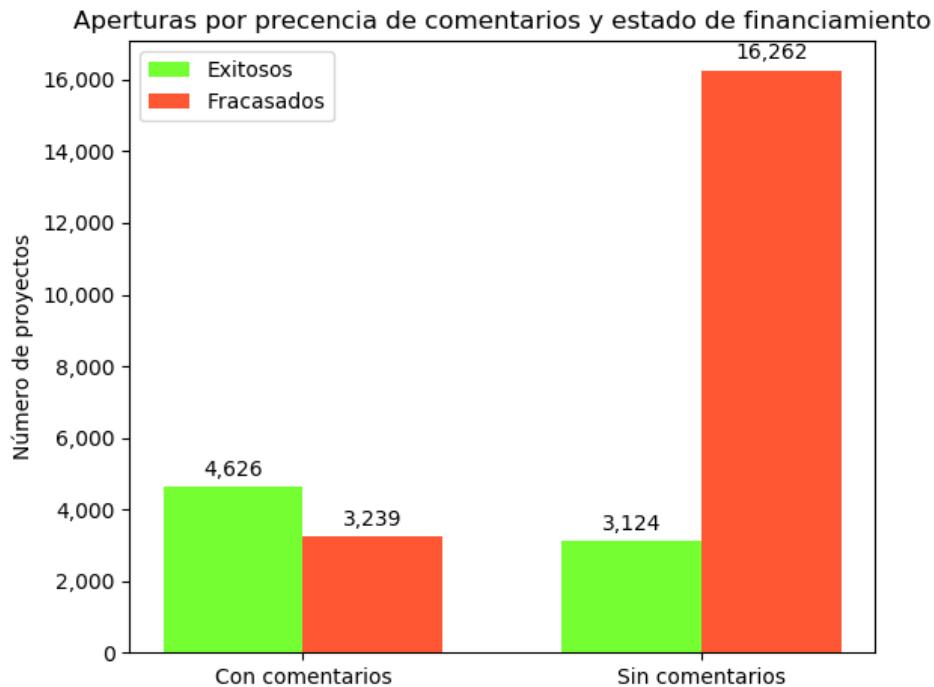


Figura 65. Distribución de proyectos por presencia de comentarios y estado final.

Fuente: Elaboración propia.

Esto señala que es más probable que un proyecto sin recibir comentarios tiende a fracasar por la diferencia notable entre ambas categorías (68 %). Por el contrario, para el caso de aquellos que presentan comentarios, no se puede formular una hipótesis sobre su comportamiento ya que la diferencia de proporciones no es muy alta (20 %) en comparación con el grupo sin comentarios. Por ello, para conocer un poco más a este último grupo, se analizó el impacto de las cantidades de comentarios independientes realizados por patrocinadores exclusivamente en el resultado final de la meta de financiamiento.

De aquellos proyectos con comentarios, el 96% que fueron financiados exitosamente recogieron más de 475 mil comentarios, como se aprecia en la Figura 66.

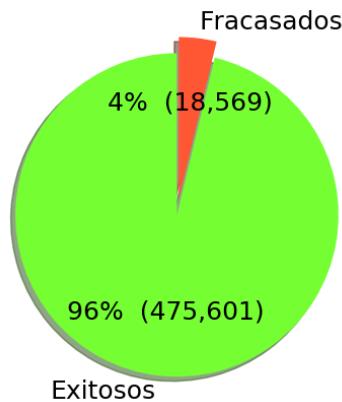


Figura 66. Distribución de comentarios en proyectos exitosos y fracasados.

Fuente: Elaboración propia.

Respecto al contenido, en la Figura 67 se ilustra la nube de palabras más frecuentes, respectivamente, luego de quitar URLs, emoticonos y términos en idioma distinto al inglés.

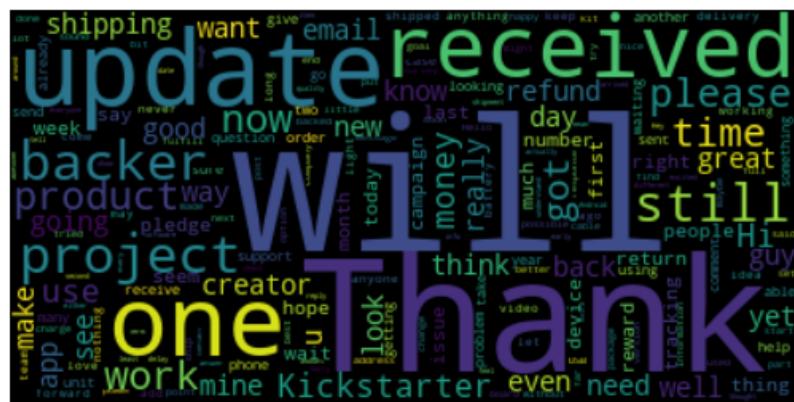


Figura 67. Nube de palabras de comentarios más frecuentes.

Fuente: Elaboración propia.

De esta imagen, se observa que las palabras más frecuentes en los comentarios (tamaño de fuente más grande) se relacionan con términos clave respecto a la campaña (*update, project, backer, product* y *Kickstarter*) y palabras relacionadas a su interacción con el público (*thank, will, received, time, money*). Algunos de estos términos, tanto en su forma raíz como en conjugaciones, suelen aparecer solitarias o acompañados de otros para formar frases recurrentes.

4.3 Preparación de los datos

Actividad 1: Pre-procesar base de datos de Metainformación

De acuerdo a los autores K. Chen et al. (2013), S.-Y. Chen et al. (2015) y Jin et al. (2019), a las 5 potenciales variables numéricas se adicionaron 7 variables basadas en el mecanismo financiero (mediana (*pledges_median*), promedio (*pledges_mean*), valor máximo (*pledges_max*), valor mínimo (*pledges_min*), variación estándar (*pledges_std*) y cantidad de montos disponibles para contribuir (*pledges_num*)) y efecto progresión (porcentaje de financiamiento o completitud (*completeness*) del monto prometido). Esta última se calcula dividiendo el monto alcanzado en el tiempo t, sobre la meta de la campaña, multiplicado por 100%. La nueva matriz de correlación se observa en la Figura 68.

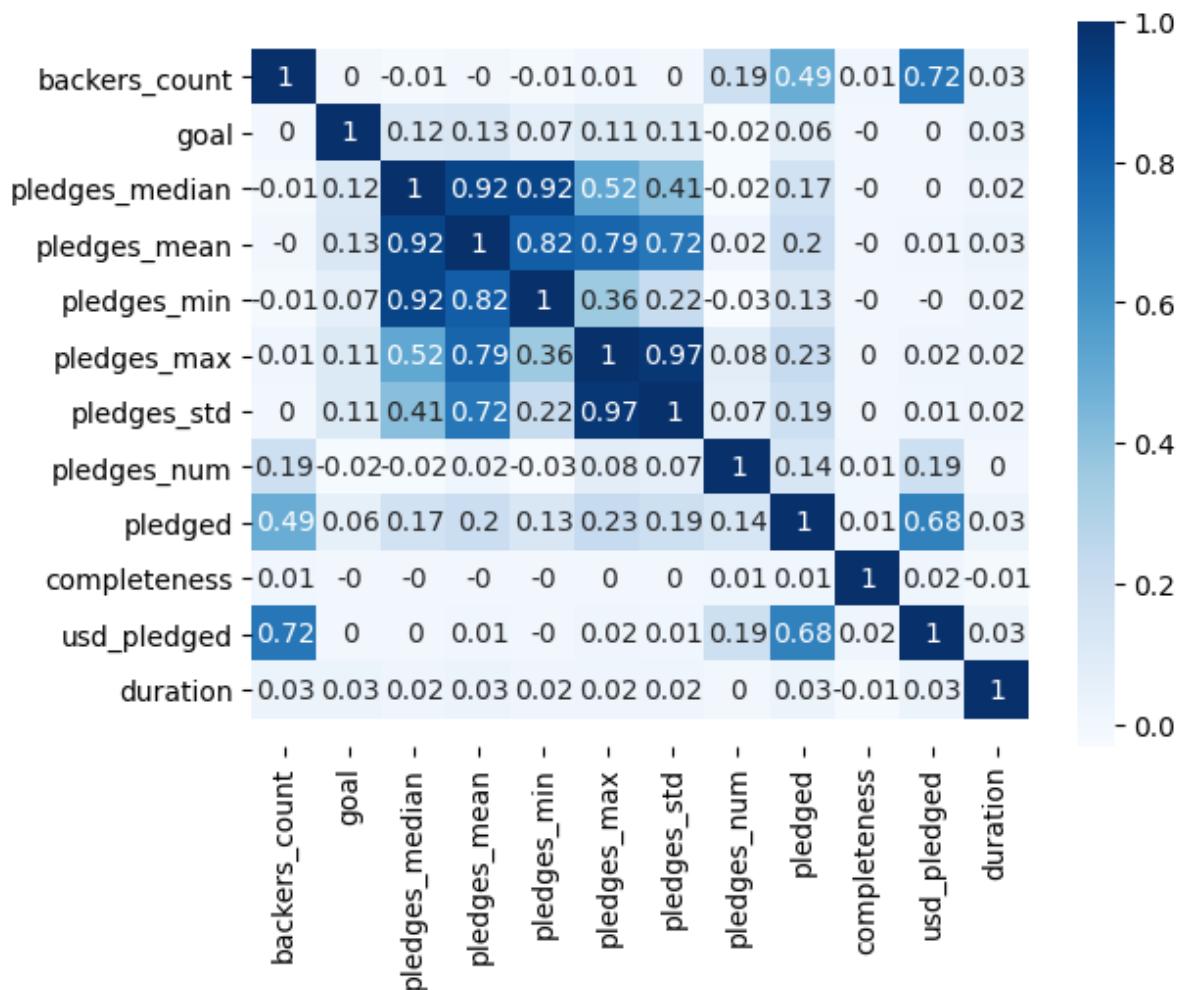


Figura 68. Matriz de correlaciones entre variables independientes considerando adicionales.

Fuente: Elaboración propia.

Para la selección de variables, se consideró a aquellas con correlación clasificada como insignificante, es decir, menor o igual a 0.30 (Mukaka, 2012). Las únicas que cumplen son *goal*, *pledges_num*, *completeness* y *duration*. Sin embargo, algunas de las restantes pueden ser consideradas condicionándose a excluir otras. En la Tabla 2 se listan las 8 combinatorias posibles de variables que se pueden formar.

Tabla 2
Potenciales combinatorias de variables de metainformación.

Combinación 1	Combinación 2	Combinación 3	Combinación 4
goal	goal	goal	goal
completeness	completeness	completeness	completeness
duration	duration	duration	duration
pledges_num	pledges_num	pledges_num	pledges_num
backers_count	backers_count	backers_count	backers_count
pledges_median	pledges_mean	pledges_min	pledges_max
		pledges_std	
Combinación 5	Combinación 6	Combinación 7	Combinación 8
goal	goal	goal	goal
completeness	completeness	completeness	completeness
duration	duration	duration	duration
pledges_num	pledges_num	pledges_num	pledges_num
pledged	pledged	pledged	pledged
pledges_median	pledges_mean	pledges_min	pledges_max
		pledges_std	

Fuente: Elaboración propia.

Una vez generadas las variables independientes (X) y dependiente (Y), el conjunto de datos es separado en subconjuntos de entrenamiento y prueba, con proporciones de 80% y 20% respectivamente (L.-S. Chen & Shen, 2019; Mitra & Gilbert, 2014; Sawhney et al., 2016; Yu et al., 2018; Yuan et al., 2016) y se fija un valor de aleatoriedad. Dentro de los parámetros de separación, se establece el argumento de estratificación según la variable Y, es decir, cada subconjunto mantendrá la distribución 72% exitosos y 28% fracasados.

```
train_ratio = 0.80
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size = 1 - train_ratio, stratify = Y, random_state=0)
```

Figura 69. Función para dividir base de datos en subconjuntos de entrenamiento y prueba.

Fuente: Elaboración propia.

Luego, utilizando el escalador Mínimo Máximo (*Min-Max scaler*) de la librería Scikit-learn, se normalizaron las variables independientes a un nuevo rango conformando valores entre 0 y 1. La función creada para este proceso se muestra en la Figura 70.

```
# Escalado de variables
from sklearn.preprocessing import StandardScaler # Importan el modulo
from sklearn import preprocessing
sc_X = preprocessing.MinMaxScaler() # Definir la funcion
X_train = sc_X.fit_transform(X_train) # Ajuste
X_test = sc_X.transform(X_test) # Aplicacion
```

Figura 70. Función para normalizar variables.

Fuente: Elaboración propia.

Para calcular los nuevos valores normalizados usando el anterior escalador, se sigue la siguiente fórmula (Ciaburro & Joshi, 2019; Pedregosa et al., 2011; Scikit-learn, s.f.):

$$x_{escalado} = \frac{x - x_{min}}{x_{max} - x_{min}} \quad (4)$$

Donde x representa el valor original de un dato, x_{min} el valor mínimo existente para dicha variable y x_{max} , el valor máximo.

Por ejemplo, tomando como referencia las estadísticas de la variable *duration* en la Figura 60, se desea transformar una duración de 30 días dentro del rango [0; 1]. Para aplicar la fórmula, los valores serían $x = 30$, $x_{min} = 1$ y $x_{max} = 92$. Entonces, el nuevo resultado sería $x_{escalado} = \frac{30-1}{92-1} = 0.32$.

Actividad 2: Pre-procesar base de datos de Descripción

Se realizó la limpieza de texto basándose en los trabajos de los autores Mitra y Gilbert (2014), Yuan et al. (2016) y L.-S. Chen y Shen (2019) y además, se agregaron pasos de lematización y supresión de palabras de parada siguiendo el proceso dictado en el curso de Procesamiento de Lenguaje Natural en la Escuela Superior de Economía de la Universidad Nacional de Investigación, Rusia (Zimovnov, 2018). Antes de ejecutarse el proceso, los registros sin descripciones (*NaN*) fueron convertidos en cadena (*string*) para evitar problemas de procesamiento de texto. Se remueven las contracciones, caracteres especiales, enlaces externos y contenidos en otros idiomas. Este resultado fue separado en palabras o tokens para eliminar palabras de parada en inglés, lematizar las restantes y finalmente juntarlas en una lista por su proyecto.

Cada iteración se pudo lograr gracias a elementos de la biblioteca para procesamiento de lenguaje natural Natural Language Toolkit (NLTK), como por ejemplo *word_tokenize*, *stopwords* y *WordNetLemmatizer*. La descripción de mayor longitud pasó a presentar 3,671 palabras y a nivel general de proyectos, el nuevo vocabulario tuvo 165,526 palabras.

Las nubes de palabras reflejan las palabras más frecuentes dentro de un conjunto de datos. La Figura 71 representa aquellas palabras que más aparecen en las descripciones.

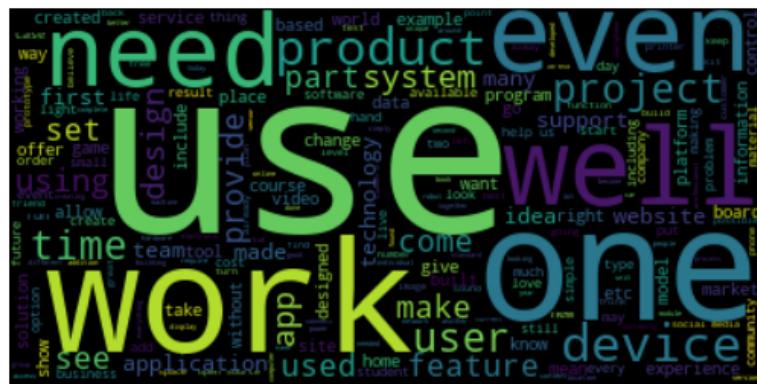


Figura 71. Nube de palabras de descripciones posterior a la limpieza de texto.

Fuente: Elaboración propia.

Luego de la limpieza de textos, la variable independiente *description*, como en el caso de Metainformación, fue dividida en subconjuntos de 80% para entrenamiento y 20% para prueba, estratificados según la distribución de la variable *state*. A continuación, en la Figura 72 se ejecuta el proceso para representar las palabras de las descripciones en vectores.

```
# función para entrenar tokenizador
def create_tokenizer(lines):
    tokenizer = Tokenizer(oov_token=<OOV>")
    tokenizer.fit_on_texts(lines)
    return tokenizer

# tokenizar conjunto de entrenamiento
tokenizer = create_tokenizer(training_sentences)

# función para calcular máxima longitud de un documento (sin palabras de parada)
def max_length(lines):
    return max([len(s.split()) for s in lines])

# calculando máxima longitud de un documento
length_long_sentence = max_length(training_sentences)

# función para codificar lista de líneas
def encode_text(tokenizer, lines, length):
    # codificación
    encoded = tokenizer.texts_to_sequences(lines)
    # rellenar secuencias codificadas
    padded = pad_sequences(encoded, maxlen=length, padding='post')
    return padded

# codificar datos
training_sentences = encode_text(tokenizer, training_sentences, length_long_sentence)
testing_sentences = encode_text(tokenizer, testing_sentences, length_long_sentence)
```

Figura 72. Proceso de representación de palabras en vectores codificados.

Fuente: Elaboración propia.

De acuerdo al algoritmo de la figura anterior, se usó la función *Tokenizer* de la librería **tensorflow.keras.preprocessing.text**, para separar las palabras únicas o *tokens* de una línea de texto asignada. En caso se encuentre un término no identificado dentro del diccionario a entrenar, se asignará a dicho token el valor de *<OOV>*. Esta función se aplicó al subconjunto de entrenamiento para elaborar un diccionario a partir de sus tokens. Luego, se creó una función para determinar la mayor longitud de palabras de las descripciones del dataset. Esta cantidad representó 3,671 términos. A continuación, se desarrolló una función para crear una secuencia de las palabras codificadas, homologar hasta la máxima longitud de descripciones y llenar con ceros a la derecha (parámetro *padding='post'*) en caso un vector no alcance esta longitud. Se aplicó este proceso a los subconjuntos originales de entrenamiento y prueba.

Una vez obtenido el vocabulario de palabras únicas y asignado el tamaño de cada arreglo (en este caso, se asignó el de la descripción de mayor longitud), se procedió a elaborar la matriz de características usando incrustaciones de GloVe como se observa en la Figura 73. Para la presente investigación, se seleccionó la opción Wikipedia 2014 + Gigaword 5, con una matriz de 100 columnas, donde cada una contendrá las incrustaciones de palabras GloVe para las palabras del corpus, cuyos índices se corresponderán con cada número de fila (Malik, 2019).

```
from numpy import array, asarray, zeros
# abrir archivo GloVe y crear diccionario
glove_file = open('Glove/glove.6B.100d.txt', encoding="utf8")
embeddings_dictionary = dict()
# completar diccionario con palabras entrenadas de GloVe
for line in glove_file:
    records = line.split()
    word = records[0]
    vector_dimensions = asarray(records[1:], dtype='float32')
    embeddings_dictionary[word] = vector_dimensions
# cerrar archivo GloVe
glove_file.close()
# crear matriz de incrustación de palabras
embedding_matrix = zeros((vocab_size, embedding_dim))
for word, index in tokenizer.word_index.items():
    embedding_vector = embeddings_dictionary.get(word)
    if embedding_vector is not None:
        embedding_matrix[index] = embedding_vector
```

Figura 73. Proceso de creación de matriz de incrustaciones de palabras.

Fuente: Elaboración propia.

En la figura anterior, luego de abrirse el archivo GloVe, se completa un diccionario creado con los registros extraídos del algoritmo GloVe. Luego de terminarse este proceso y cerrarse el archivo, se crea la matriz de incrustación de palabras que será utilizada más adelante en la capa de incrustación del modelo predictivo.

Actividad 3: Pre-procesar base de datos de Comentarios

La base de datos de comentarios está conformada a nivel de 1 proyecto con una lista de comentarios separados en sublistas. De los 7,750 proyectos con comentarios (4,626 exitosos), se removieron aquellos que presentaron términos en idioma distinto al inglés, URLs, emoticonos, emojis, números y caracteres especiales, quedando 7,658 registros (4,574 exitosos).

Debido a la gran cantidad de registros carecientes de comentarios, se propuso rellenarlos con un término aleatorio no relacionado con las temáticas principales: *kuwagatabaizan*. Posteriormente, se repitió el mismo ejercicio para las descripciones. Sin considerar este nuevo término, la nube de palabras se representa en la Figura 74.



Figura 74. Nube de palabras de comentarios posterior a la limpieza de texto.

Fuente: Elaboración propia.

Al igual que el caso de Descripción, se siguieron los procesos de la Figura 72 para obtener la representación de palabras en vectores codificados y la Figura 73 para crear la matriz de incrustación de palabras que se usarán en la capa de incrustaciones, con la diferencia en que el tamaño de la secuencia de palabras codificadas no será la máxima longitud de comentarios ya que estos, inicialmente separados por autor, al ser concatenados en un solo registro por proyecto, ampliaron su longitud exponencialmente. La máxima longitud de todos los registros fue de 30,072 palabras; por ello, se limitó el tamaño de secuencias a 5,000 términos.

Capítulo V: Análisis y Discusión de Resultados

En este capítulo, se continuaron las 3 últimas fases de la metodología seleccionada.

5.1 Modelamiento

Antes de crear los modelos correspondientes, y después de definir los valores de entrada y parámetros (las subsecciones que se detallarán a continuación), se asigna una semilla inicial con un valor fijado por el usuario con el fin de evitar resultados aleatorios para futuras iteraciones. Se establece, además, una ruta local en donde se almacena cada punto de control basado en la mejora de la pérdida del subconjunto de validación con respecto a su iteración anterior. En caso de un estancamiento de esta última durante 10 épocas, es decir, si el valor de la pérdida no decrementa, el modelo dejará de entrenar. A esta regla se le añade la reducción de la tasa de aprendizaje luego de 5 épocas en caso el valor de la exactitud del subconjunto de validación no refleje un incremento. El objetivo de estas condiciones es evitar el sobreajuste en los modelos durante el entrenamiento.

Por último, es importante asignar un peso distinto para cada una de las dos clases de la variable dependiente *state*. Con el fin de evitar un mal entrenamiento, los pesos de ambas clases se balancean y se almacenan en un diccionario con su etiqueta correspondiente.

Actividad 1: Desarrollar modelo predictivo de Metainformación

Se diseñó el modelo de descripciones basada en un Perceptrón Multicapa (MLP por sus siglas en inglés) bajo la arquitectura de la Figura 75 y teniendo como referencia a los autores Yu et al. Se asignaron 100 épocas y el número de lotes fue 32.

La arquitectura comienza con la capa de entrada alimentadas por las 6 variables consideradas, que representan la cantidad de neuronas, tanto de entrada como de salida.

Si bien no existe alguna regla general para definir el número de capas óptimas, así como los hiperparámetros que se deben configurar en ellas, se puede utilizar como referencia algunas metodologías como las Reglas del Pulgar según Ranjan (2019).

De acuerdo a una de ellas, el número de capas ocultas comienza con 2 sin contar la última. La primera capa densa continúa a la capa de entrada, mientras que la segunda aparece después de la primera capa de desactivación.

Otro punto considerado fue el número de nodos o neuronas de las capas intermedias. Estas deben seguir una progresión geométrica de 2, donde la primera capa debe ser la mitad del número de variables en la capa de entrada. Dado que la mitad de 6 es un valor que no cumple, un número potencial puede ser 4.

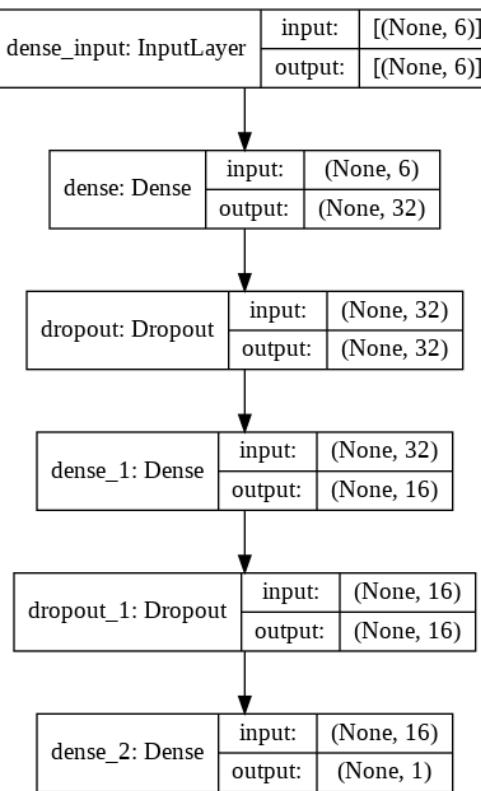


Figura 75. Arquitectura de modelo MLP para la metadata.

Fuente: Elaboración propia.

El autor también menciona tener en consideración utilizar la función de activación *relu* para las capas intermedias, una tasa de abandono de por lo menos 0.5 para las capas de desactivación, tamaño de salida de 1 neurona y función de activación *sigmoide* por tratarse de un problema de clasificación binaria, utilizar el optimizador *adam*, comenzar con 20 épocas en adelante de acuerdo al progreso de los resultados y fijar un tamaño de lote bajo progresión geométrica de 2; además de otros requerimientos previamente establecidos como la ponderación de clases para la variable dependiente en caso de datos desbalanceados y escalado de datos antes del entrenamiento.

Estas opciones fueron probadas en el modelo y evaluadas con las métricas correspondientes. Sin embargo, al calibrar el modelo y comparar distintos resultados, se obtuvo que la mejor cantidad de neuronas para la primera capa densa era de 32. De este modo, la siguiente capa intermedia se le asignó la mitad (16). La función de activación *tanh* para la segunda capa oculta presentó mejores resultados, así como tasas de abandono entre 0.25 y 0.3 para las capas de desactivación. El criterio para elegir esta función se explica en el modelo de descripción, en el cual también fue aplicado. Por último, además de *adam*, se realizó experimentos con otros optimizadores como por ejemplo *RMSprop* siendo este el resultado más cercano. Al final, *adam* fue escogido pero con una tasa de aprendizaje baja como 0.005 debido a que el modelo tenía a

aprender muy rápido durante el transcurso de las épocas. Al ratio de decaimiento se le asignó, entonces, el valor de 0.00005 y para evaluar el modelo se usó la exactitud.

El resumen de la explicación anterior, desde la configuración de parámetros para entrenar hasta los elementos presentes en cada capa, se encuentra en el Anexo F.

Actividad 2: Desarrollar modelo predictivo de Descripción

Se diseñó el modelo de descripciones basada en una Red Neuronal Convolutacional unidimensional (Conv1D) bajo la arquitectura de la Figura 76, así como de referencia un trabajo de análisis de sentimientos de películas (Malik, 2019). Se asignaron 100 épocas para entrenar y número de lotes de 128.

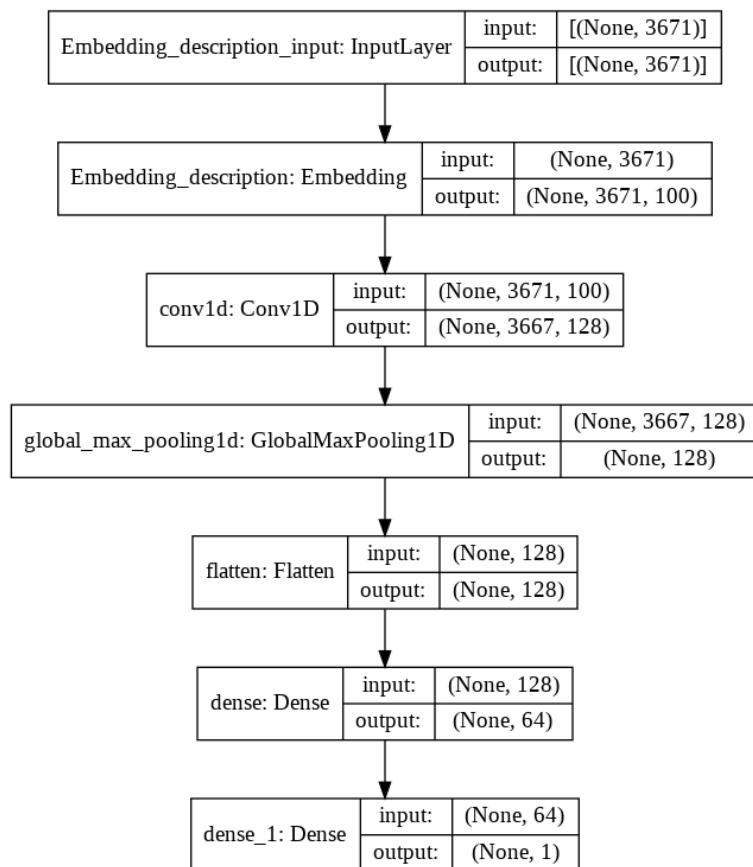


Figura 76. Arquitectura de modelo CNN para las descripciones.

Fuente: Elaboración propia.

Esta red se compone de una capa de incrustación de palabras o *Embedding* alimentada por los datos de entrada en la primera capa *InputLayer* de dimensión de 3,671 vectores de palabras (la mayor longitud de palabras de todas las descripciones), la cual genera como salida una matriz de 3,671 por 100 (número de columnas de incrustaciones de GloVe). Para esta capa se entrenarán 14,827,000 parámetros como resultado del producto de las 100 columnas mencionadas y 148,270 como el tamaño del vocabulario entrenado.

La siguiente capa es la Convolución en 1 dimensión o *Conv1D* (usado frecuentemente para extraer características de datos de textos por ser unidimensionales) que, con 128 características, 5 de tamaño de kernel y función de activación *relu*, generó una salida de 3,667 por 128; así como 64,128 parámetros entrenables.

A continuación, le sigue la capa de reducción *GlobalMaxPooling*. Al igual que en la convolución, esta también fue unidimensional y se caracteriza por realizar agrupamiento global basado en el valor máximo de los bloques seleccionados para reducir el tamaño del vector generado.

Esta nueva salida pasa por la capa de aplanamiento o *Flatten*, en donde se multiplican las filas y columnas y tener un solo vector. Esta sirve para conectar con las 64 neuronas de la nueva capa densa y función de activación *tanh*, agregada con el fin de mejorar la performance del modelo. Según Brownlee (2019), se puede considerar el uso tanto de una función *relu* como una función *tanh* cuando se presente la desaparición de gradientes al propagar hacia atrás a mayor cantidad de capas, hecho presentado en los experimentos. Si bien menciona que el uso de la función tangente hiperbólica en capas ocultas resultó una buena práctica durante las décadas de 1990 y 2000, teniendo mejor rendimiento que la función logística, afirma que ambas son dos opciones válidas para problemas de redes neuronales profundas. Por lo tanto, el criterio para considerar una función *tanh* en la investigación obedece a un mejor desempeño en 2 % más de exactitud en el entrenamiento que utilizando la función *relu*.

Finalmente, la arquitectura culmina con la última capa con función de activación *sigmoid* para regular el valor de salida entre 0 y 1, ya que, al tratarse de un problema de clasificación binaria (predecir si un proyecto será financiado: exitoso, de lo contrario: fracasado), cuenta con solo 1 neurona y su parámetro de pérdida es *binary_crossentropy*. El resumen de todo lo anterior explicado se encuentra en el Anexo G.

Actividad 3: Desarrollar modelo predictivo de Comentarios

Al igual que en el modelo de descripciones de proyectos, se creó un diccionario de palabras con la data luego de tokenizar y codificarlas con las mismas funciones y librerías. Sin embargo, a diferencia del anterior modelo, la longitud de la matriz para el relleno de ceros con el fin de homogenizar el tamaño de cada vector de incrustaciones se limitó a las 5,000 últimas palabras de una oración (parámetro **padding='post'** de la función *pad_sequences*) en lugar de la longitud máxima dado que ésta representa una cantidad considerable (30,072) como para utilizar todos los recursos del entorno de ejecución.

De igual manera, se creó una matriz de incrustaciones de palabras usando GloVe y se diseñó una arquitectura basada en una Red Neuronal Recurrente (RNN por sus siglas en inglés) ilustrada en la Figura 77.

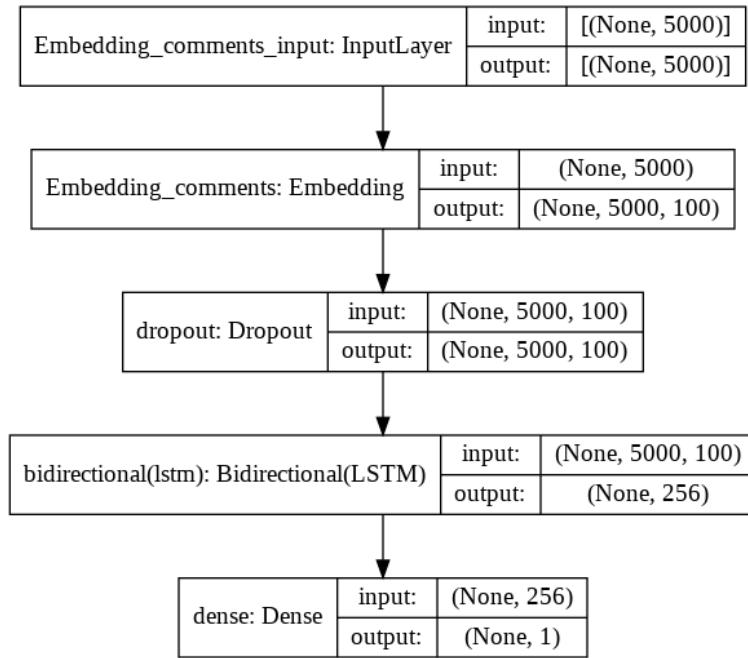


Figura 77. Arquitectura de modelo RNN para los comentarios.

Fuente: Elaboración propia.

Existe una similitud de estructura en las 2 primeras capas con las del modelo de descripción. Luego de la capa de incrustaciones o *Embedding*, para reducir las conexiones entre neuronas se añadió 1 capa de desactivación o *Dropout*.

A continuación, los vectores de palabras ingresan a la capa de la red neuronal recurrente *LSTM*. Para este caso, se consideró envolverla dentro de una Red Bidireccional de 128 neuronas, ya que como se explicó en el Marco Teórico del Capítulo II sobre las RNN Bidireccionales, este tipo es una mejora de la LSTM tradicional al entrenar 2 juntas (la segunda representa una copia invertida de la secuencia de entrada) en donde cada capa ahora puede considerar también información de las capas siguientes junto con la información de las previas que ya tenía en cuenta, es decir, toma información de 2 direcciones (Brownlee, 2017).

Finalmente, el modelo culmina con una capa densa en donde recibe 256 valores de entrada (2×128 de la capa Bidireccional LSTM) y utiliza la función de activación *sigmoid* para transformar el valor final entre 0 y 1. El resumen de lo anterior se encuentra en el Anexo H.

Antes de continuar con el desarrollo del modelo de Aprendizaje Profundo Multimodal, en la Tabla 3 se presentan las variables de las modalidades que se usaron para entrenarlo. Estas se seleccionaron de acuerdo al Benchmarking aplicado a los antecedentes en el Capítulo II.

Los autores citados por cada variable utilizada se mencionan a continuación:

Tabla 3
Diccionario de datos del conjunto final entrenado.

Variable	Detalle	Tipo de dato
Variables independientes		
goal	Monto de la meta de financiamiento del proyecto.	float64
completeness	Porcentaje de financiamiento o completitud.	float64
duration	Duración de la campaña (en días).	int64
pledges_num	Cantidad de montos disponibles para contribuir.	int64
pledged	Monto contribuido en la campaña.	float64
pledges_median	Mediana de montos disponibles para contribuir.	float64
description	Descripción del proyecto.	object
comments	Comentarios de patrocinadores sobre el proyecto.	object
Variable dependiente		
state	Estado de financiamiento del proyecto.	object

Fuente: Elaboración propia.

- **goal:** K. Chen et al. (2013), Mitra y Gilbert (2014), Zhou et al. (2015), S.-Y. Chen et al. (2015), Li et al. (2016), Yuan et al. (2016), Sawhney et al. (2016), Kaur y Gera (2017), Kamath y Kamat (2018), Yu et al. (2018), Jin et al. (2019), Cheng et al. (2019).
- **completeness:** S.-Y. Chen et al. (2015).
- **duration:** Mitra y Gilbert (2014), Zhou et al. (2015), Li et al. (2016), Sawhney et al. (2016), Kaur y Gera (2017), Kamath y Kamat (2018), Yu et al. (2018), Jin et al. (2019).
- **pledges_num:** K. Chen et al. (2013), Mitra y Gilbert (2014), S.-Y. Chen et al. (2015), Yuan et al. (2016), Jin et al. (2019).
- **pledged:** K. Chen et al. (2013), Li et al. (2016), Kamath y Kamat (2018).
- **pledges_median:** S.-Y. Chen et al. (2015)*, Jin et al. (2019)*.
- **description:** Mitra y Gilbert (2014), Zhou et al. (2015), Yuan et al. (2016), Sawhney et al. (2016), Kamath y Kamat (2018), Lee et al. (2018), Jin et al. (2019), Cheng et al. (2019), L.-S. Chen y Shen (2019), Chaichi y Anderson (2019).
- **comments:** Li et al. (2016), Kaur y Gera (2017), Lee et al. (2018), Jin et al. (2019).

Si bien en los respectivos antecedentes marcados en (*) figuran el promedio de los montos disponibles para patrocinar, se usó la mediana en vez de la media ya que presentó mejor performance en los experimentos.

Actividad 4: Desarrollar modelo ensamblado apilado

Una vez construidos los modelos para cada modalidad (metainformación, descripción y comentarios), se construyó un modelo de Aprendizaje Profundo Multimodal ilustrado en la Figura 78.

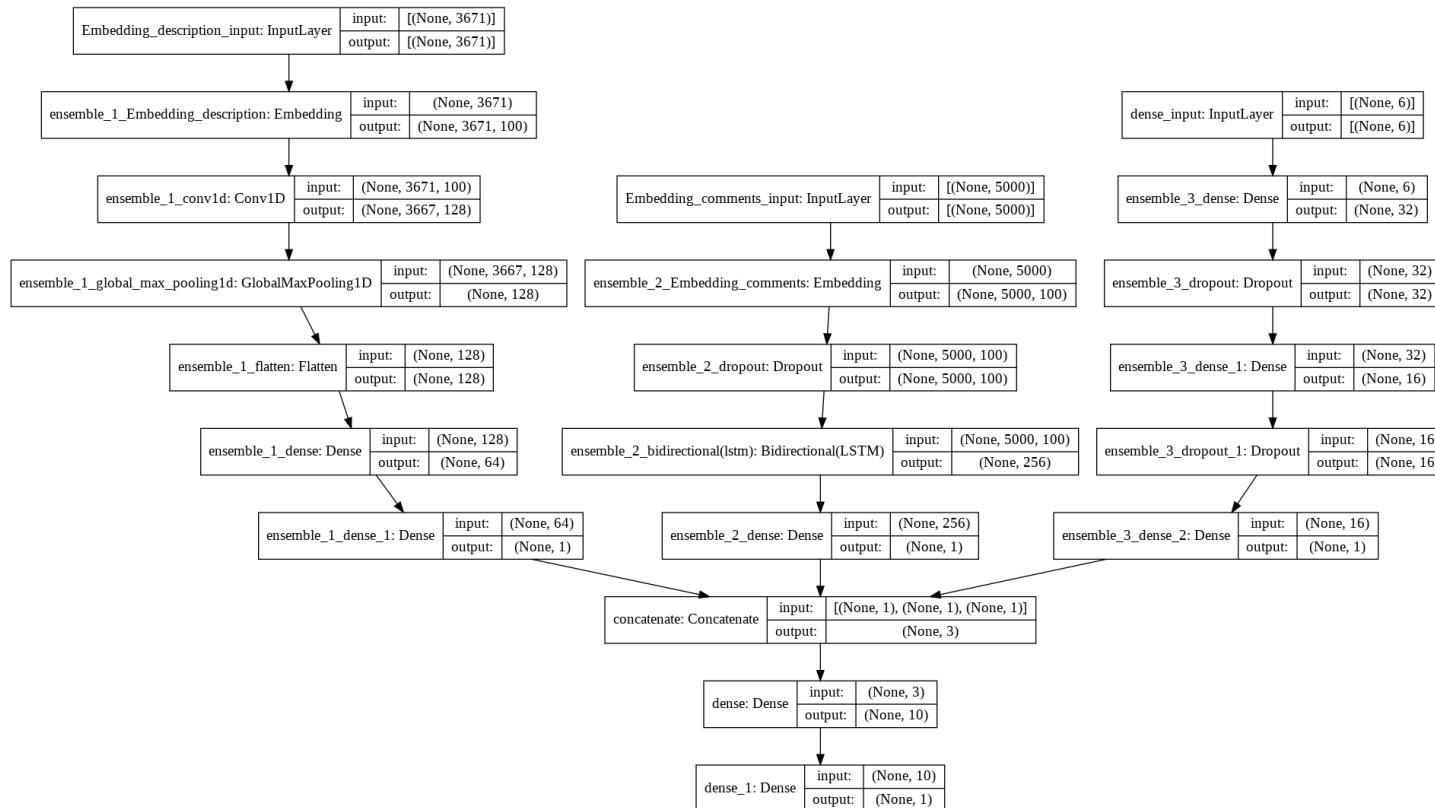


Figura 78. Arquitectura del modelo apilado final The Hydra.

Fuente: Elaboración propia.

La finalidad de este modelo apilado de múltiples cabezas es aprender la mejor manera de combinar las predicciones de cada submodelo para lograr un mayor rendimiento y clasificar mejor la variable dependiente que cada modalidad.

A este modelo se le denominó “*The Hydra*” (La Hidra por su traducción al español) en referencia al monstruo mitológico del lago de Lerna, con 7 cabezas que renacían a medida que se cortaban (Real Academia Española, s.f.).

Al tratarse de un modelo ensamblado apilado, las salidas de cada modelo se concatenaron en una capa debajo de estos, generando 3 valores de entrada para una penúltima capa densa con 10 neuronas de salida y una función de activación *relu*. El modelo apilado culmina con una capa densa de 1 salida y asignándose la función Sigmoide para generar probabilidades entre 0 y 1, los valores de Fracasado o Exitoso respectivamente.

Previo a la compilación del modelo final, se repitió el ejercicio de cada modelo cargado asignar los parámetro de pérdida *binary_crossentropy* para la clasificación binaria, *accuracy* (exactitud) para la métrica del entrenamiento, pesos balanceados para las clases de la variable *state* (0.6987077585764833 para 0 y 1.7581290322580645 para 1), y optimizador *Adam* con la variante de asignarle el ratio de aprendizaje y también de decaimiento de 0.00005. El resumen de todo los parámetros anteriores se encuentra en el Anexo I.

5.2 Evaluación

Como parte de la aplicación de la metodología CRISP-DM, explicada en el sexto subcapítulo del Capítulo III, se mencionaron las métricas usadas en la literatura. La más recurrente fue la exactitud. Dado que la librería Scikit-learn cuenta con un reporte de clasificación con esta métrica y otras 4 más como la precisión, sensibilidad, puntaje F1 y AUC, además que la distribución de proyectos por su estado de financiamiento es desbalanceada y se necesita más de un indicador para poder evaluar y comparar, se decidió usar estas 5 teniendo como referencias a los autores Beckwith (quinto antecedente), Yuan et al.* (séptimo antecedente), Kaur y Gera (noveno antecedente), Cheng et al. (decimocuarto antecedente), y L.-S. Chen y Shen** (decimoquinto antecedente).

En el antecedente marcado en (*), los modelos no fueron evaluados por AUC; mientras que en (**), las métricas precisión y AUC no fueron tomadas en cuenta.

Actividad 1: Evaluar desempeño del modelo predictivo de Metainformación

Las 8 combinaciones de variables de la Tabla 2 para el submodelo de Metainformación, luego de ser pre-procesadas utilizando el escalador Min-Max, fueron entrenadas durante 100 épocas cada una. En la Tabla 4 se presentan los resultados obtenidos.

Tabla 4

Exactitud de los conjuntos de datos de validación para las 8 combinaciones.

Combinación 1	Combinación 2	Combinación 3	Combinación 4
0.88405	0.89855	0.89763	0.88369
Combinación 5	Combinación 6	Combinación 7	Combinación 8
0.93102	0.92588	0.92478	0.92478

Fuente: Elaboración propia.

El mejor rendimiento lo produjo la combinación 5 (compuesta por las variables *goal*, *completeness*, *duration*, *pledges_num*, *pledged* y *pledges_median*), la cual alcanzó un valor de exactitud de 0.93102 luego de 19 épocas, con un promedio de 3 segundos de entrenamiento cada una. Al acabar este proceso, el modelo dejó de entrenar dado que durante 7 épocas no registró una reducción en el valor de la pérdida del subconjunto de validación, a pesar de que hace 3 épocas se redujo su tasa de aprendizaje.

Se concluye que la ventaja del grupo de las 4 últimas combinatorias frente a las primeras se da por la inclusión de la variable del monto recaudado al final de la campaña (*pledged*) en lugar del número de patrocinadores alcanzados (*backer_count*). Además, la combinación 5 supera a las siguientes 3 por considerar la mediana de los montos disponibles para contribuir en la campaña, frente a la media, variación estándar, valor máximo y valor mínimo de estos montos.

Así, de acuerdo a la Figura 79, en la época 11 se registran los mejores valores de exactitud y pérdida para el subconjunto de validación, alcanzando 0.9523 y 0.1246 respectivamente.

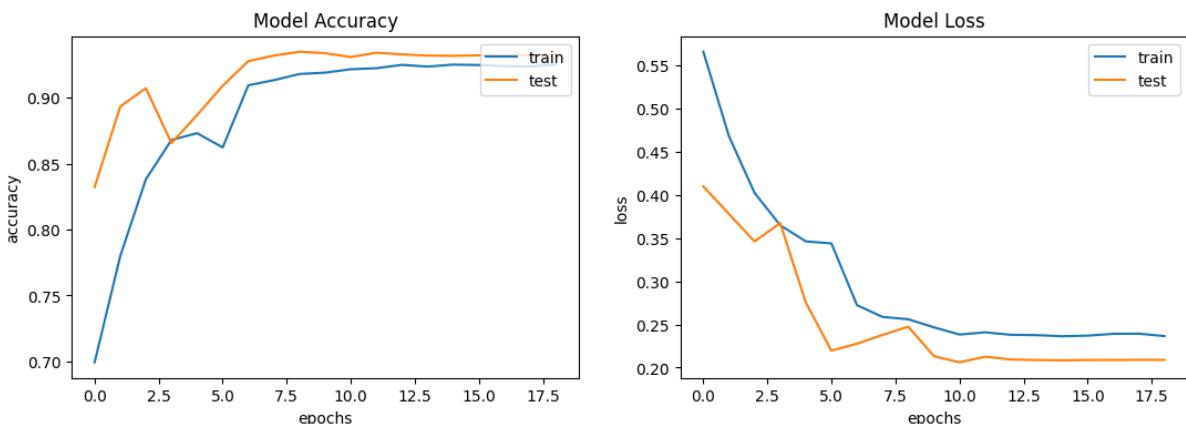


Figura 79. Exactitud y pérdida respectivamente de los subconjuntos de entrenamiento y validación para el modelo MLP de metadata con 100 épocas.

Fuente: Elaboración propia.

La matriz de confusión resultante se representa en la Figura 80.

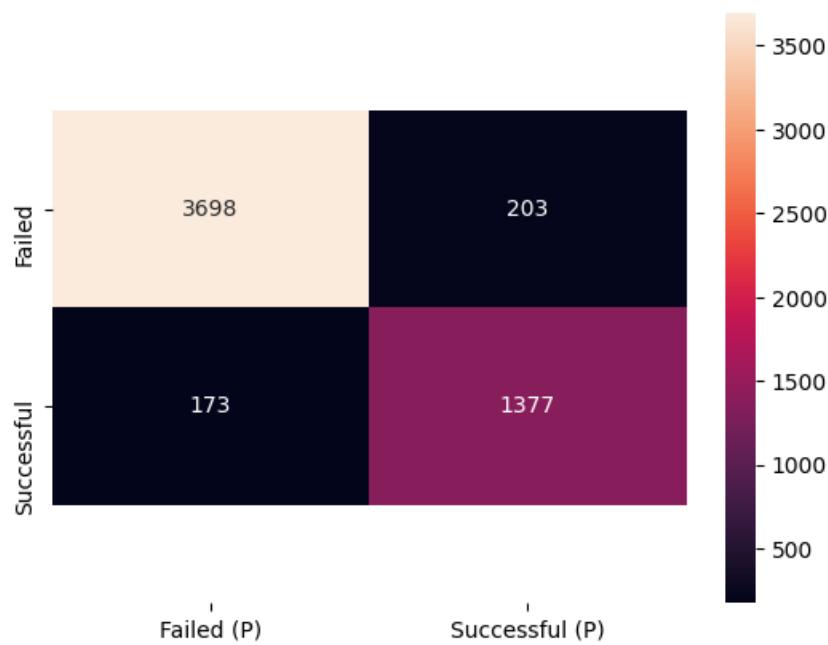


Figura 80. Matriz de confusión para el modelo de metadata.

Fuente: Elaboración propia.

De esta matriz, se derivan los resultados de la Tabla 5 y el AUC en la Figura 81.

Tabla 5

Informe de clasificación para el modelo de metadata.

Valor	Precisión	Sensibilidad	Puntaje F1	Muestras
Fracasado	0.96	0.95	0.95	3,901
Exitoso	0.87	0.89	0.88	1,550
Exactitud			0.93	5,451
Promedio macro	0.91	0.92	0.92	5,451
Promedio ponderado	0.93	0.93	0.93	5,451

Fuente: Elaboración propia.

- El ratio de exactitud se interpreta como: El 93 % de los proyectos de la muestra fueron predichos correctamente.
- El ratio de precisión para los positivos (*Successful*) se interpreta como: El 87 % de los proyectos exitosos predichos de la muestra fueron clasificados correctamente.
- El ratio de sensibilidad para los positivos (*Successful*) se interpreta como: El 89 % de los proyectos exitosos reales de la muestra fueron clasificados correctamente.
- El ratio de Puntaje F1 para los positivos (*Successful*) representa un balance entre las 2 métricas anteriores. En la tabla se observa que su valor es de 88 %, lo cual indica que en general, el modelo mantiene un alto rendimiento.

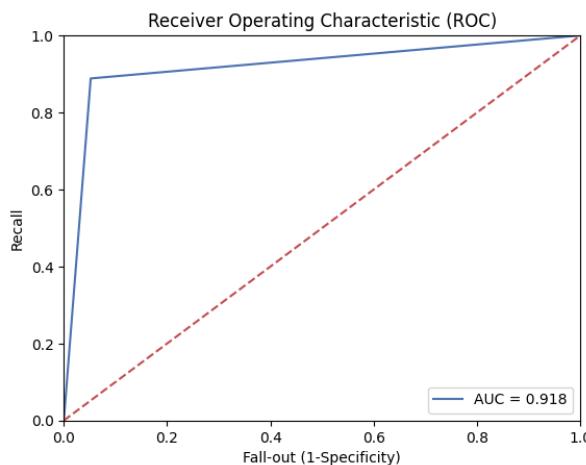


Figura 81. Área bajo la curva ROC de modelo de metadata.

Fuente: Elaboración propia.

El área bajo la Curva ROC presenta un valor de aproximadamente 92 %, del cual se observa en el gráfico que su sensibilidad es muy alta y el ratio de Falsa Alarma es casi nulo. De acuerdo con Britos et al. (2006), el poder discriminante del modelo es excelente.

Actividad 2: Evaluar desempeño del modelo predictivo de Descripción

Luego de 82 épocas, con un promedio de 106 segundos de entrenamiento cada una, el modelo dejó de entrenar dado que durante 10 épocas no registró una reducción en el valor de la pérdida del subconjunto de validación, a pesar de que hace 5 épocas se redujo su tasa de aprendizaje.

Así, de acuerdo a la Figura 82, en la época 72 se registran los mejores valores de exactitud y pérdida para el subconjunto de validación, alcanzando 0.7683 y 0.4901 respectivamente.

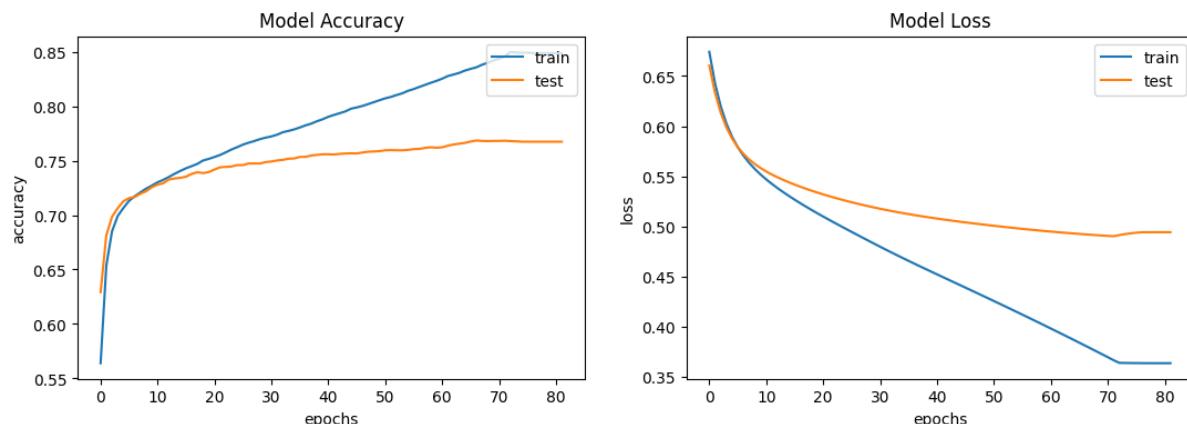


Figura 82. Exactitud y pérdida respectivamente de los subconjuntos de entrenamiento y validación para el modelo CNN de descripciones con 100 épocas.

Fuente: Elaboración propia.

La matriz de confusión resultante se representa en la Figura 83.

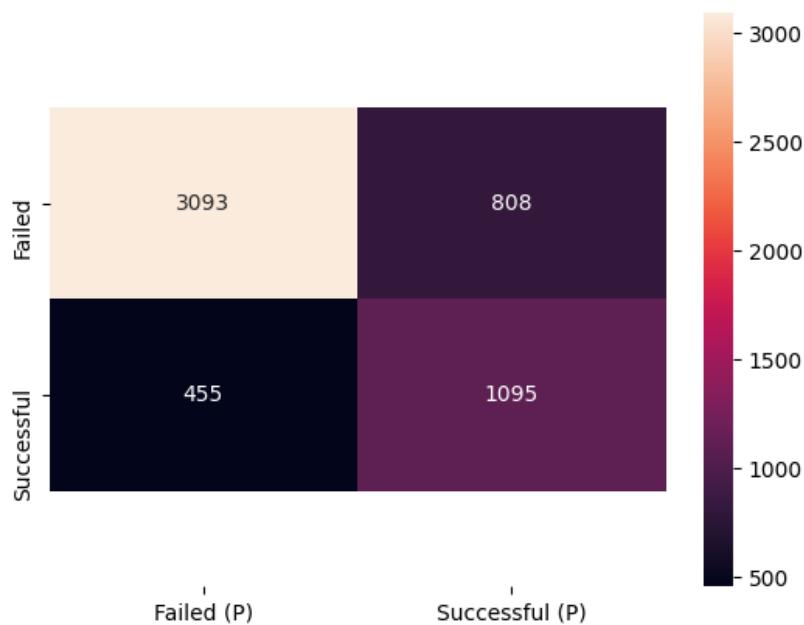


Figura 83. Matriz de confusión para el modelo de descripciones.

Fuente: Elaboración propia.

De esta matriz, se derivan los resultados de la Tabla 6 y el AUC en la Figura 84.

Tabla 6

Informe de clasificación para el modelo de descripciones.

Valor	Precisión	Sensibilidad	Puntaje F1	Muestras
Fracasado	0.87	0.79	0.83	3,901
Exitoso	0.58	0.71	0.63	1,550
Exactitud			0.77	5,451
Promedio macro	0.72	0.75	0.73	5,451
Promedio ponderado	0.79	0.77	0.77	5,451

Fuente: Elaboración propia.

- El ratio de exactitud se interpreta como: El 77 % de los proyectos de la muestra fueron predichos correctamente.
- El ratio de precisión para los positivos (*Successful*) se interpreta como: El 58 % de los proyectos exitosos predichos de la muestra fueron clasificados correctamente.
- El ratio de sensibilidad para los positivos (*Successful*) se interpreta como: El 71 % de los proyectos exitosos reales de la muestra fueron clasificados correctamente.
- El ratio de Puntaje F1 para los positivos (*Successful*) representa un balance entre las 2 métricas anteriores. En la tabla se observa que su valor es de 63 %, lo cual indica que en general, el modelo presenta un rendimiento regular.

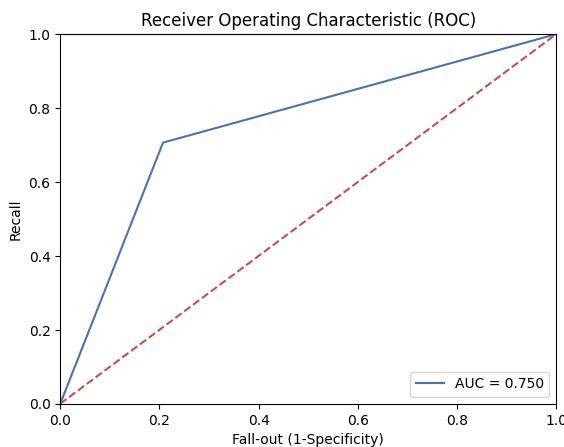


Figura 84. Área bajo la curva ROC de modelo de descripciones.

Fuente: Elaboración propia.

El área bajo la Curva ROC presenta un valor de aproximadamente 75 %, del cual se observa en el gráfico que su sensibilidad es medianamente alta y el ratio de Falsa Alarma es medianamente baja. De acuerdo con Britos et al. (2006), el poder discriminante del modelo es aceptable .

Actividad 3: Evaluar desempeño del modelo predictivo de Comentarios

Luego de 43 épocas, con 77 segundos en promedio de entrenamiento cada una, el modelo dejó de entrenar dado que durante 10 épocas no registró una reducción en el valor de la pérdida del subconjunto de validación, por más que 1 época antes se había reducido su tasa de aprendizaje.

Así, de acuerdo a la Figura 85, en la época 33 se registran los mejores valores de exactitud y pérdida para el subconjunto de validación, alcanzando 0.8510 y 0.4472 respectivamente.

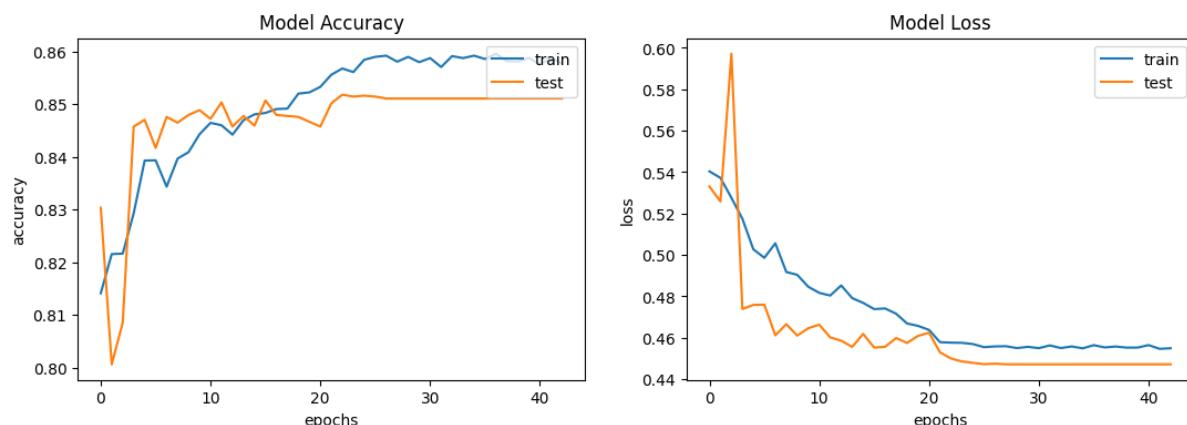


Figura 85. Exactitud y pérdida respectivamente de los subconjuntos de entrenamiento y validación para el modelo RNN de comentarios con 50 épocas.

Fuente: Elaboración propia.

La matriz de confusión resultante se representa en la Figura 86.



Figura 86. Matriz de confusión para el modelo de comentarios.

Fuente: Elaboración propia.

De esta matriz, se derivan los resultados de la Tabla 7 y el AUC en la Figura 87.

Tabla 7
Informe de clasificación para el modelo de comentarios.

Valor	Precisión	Sensibilidad	Puntaje F1	Muestras
Fracasado	0.84	0.98	0.90	3,901
Exitoso	0.91	0.53	0.67	1,550
Exactitud			0.85	5,451
Promedio macro	0.87	0.75	0.79	5,451
Promedio ponderado	0.86	0.85	0.84	5,451

Fuente: Elaboración propia.

- El ratio de exactitud se interpreta como: El 85% de los proyectos de la muestra fueron predichos correctamente.
- El ratio de precisión para los positivos (*Successful*) se interpreta como: El 91% de los proyectos exitosos predichos de la muestra fueron clasificados correctamente.
- El ratio de sensibilidad para los positivos (*Successful*) se interpreta como: El 53% de los proyectos exitosos reales de la muestra fueron clasificados correctamente.
- El ratio de Puntaje F1 para los positivos (*Successful*) representa un balance entre las 2 métricas anteriores. En la tabla se observa que su valor es de 67%, lo cual indica que en general, el modelo presenta un rendimiento regular.

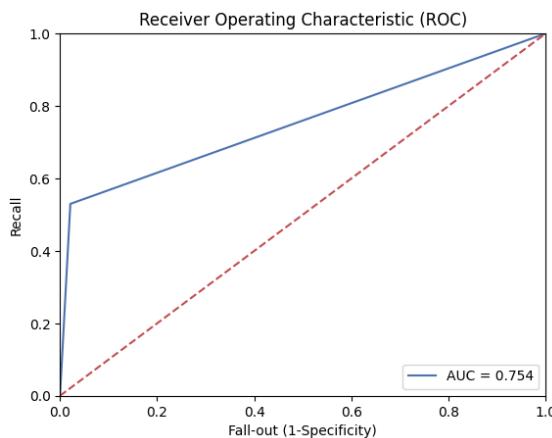


Figura 87. Área bajo la curva de modelo de comentarios.

Fuente: Elaboración propia.

El área bajo la Curva ROC presenta un valor de aproximadamente 75 %, del cual se observa en el gráfico que su sensibilidad es baja pero su ratio de Falsa Alarma es casi nulo. De acuerdo con Britos et al. (2006), el poder discriminante del modelo es aceptable.

Actividad 4: Evaluar desempeño del modelo ensamblado apilado

Luego de 13 épocas, con 152 segundos de entrenamientos cada una, el modelo dejó de entrenar dado que durante 10 épocas no registró una reducción en el valor de la pérdida del subconjunto de validación, a pesar de que hace 2 épocas se redujo su tasa de aprendizaje.

Así, de acuerdo a la Figura 88, en la época 3 se registran los mejores valores de exactitud y pérdida para el subconjunto de validación, alcanzando 0.9336 y 0.1810 respectivamente.

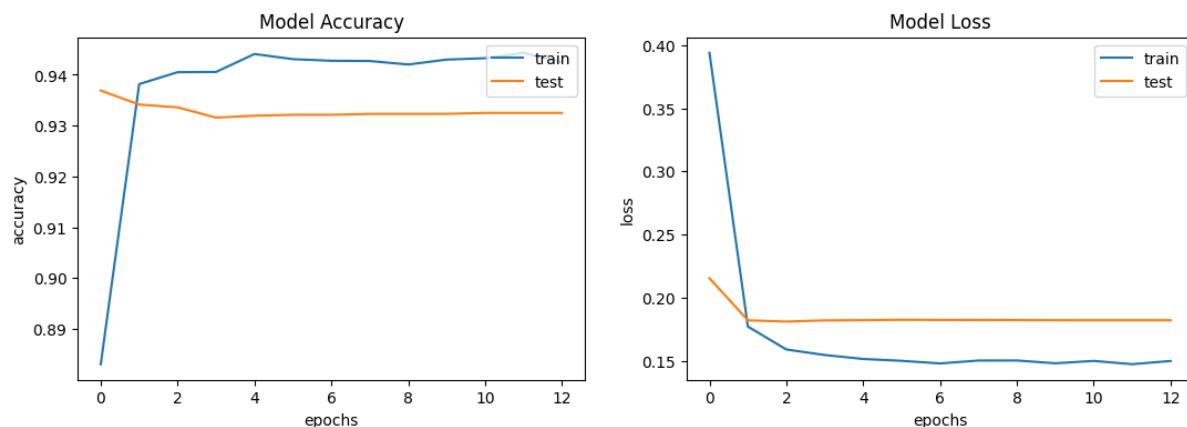


Figura 88. Exactitud y pérdida respectivamente de los subconjuntos de entrenamiento y validación para el modelo apilado con 200 épocas.

Fuente: Elaboración propia.

La matriz de confusión resultante se representa en la Figura 89.

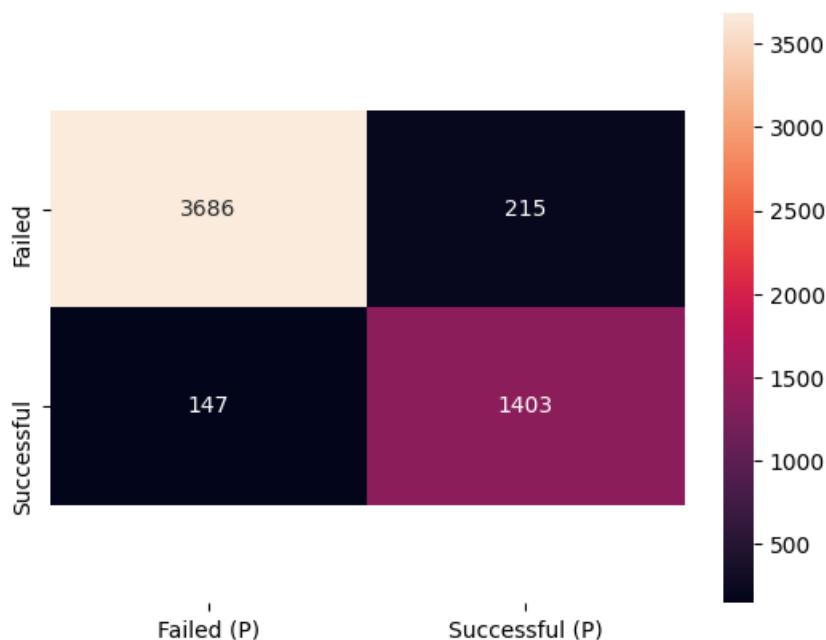


Figura 89. Matriz de confusión para el modelo apilado.

Fuente: Elaboración propia.

De esta matriz, se derivan los resultados de la Tabla 8 y el AUC en la Figura 90.

Tabla 8

Informe de clasificación para el modelo apilado.

Valor	Precisión	Sensibilidad	Puntaje F1	Muestras
Fracasado	0.96	0.94	0.95	3,901
Exitoso	0.87	0.91	0.89	1,550
Exactitud			0.93	5,451
Promedio macro	0.91	0.93	0.92	5,451
Promedio ponderado	0.93	0.93	0.93	5,451

Fuente: Elaboración propia.

- El ratio de exactitud se interpreta como: El 93 % de los proyectos de la muestra fueron predichos correctamente.
- El ratio de precisión para los positivos (*Successful*) se interpreta como: El 87 % de los proyectos exitosos predichos de la muestra fueron clasificados correctamente.
- El ratio de sensibilidad para los positivos (*Successful*) se interpreta como: El 91 % de los proyectos exitosos reales de la muestra fueron clasificados correctamente.

- El ratio de Puntaje F1 para los positivos (*Successful*) representa un balance entre las 2 métricas anteriores. En la tabla se observa que su valor es de 89 %, lo cual indica que en general, el modelo presenta un rendimiento regular.

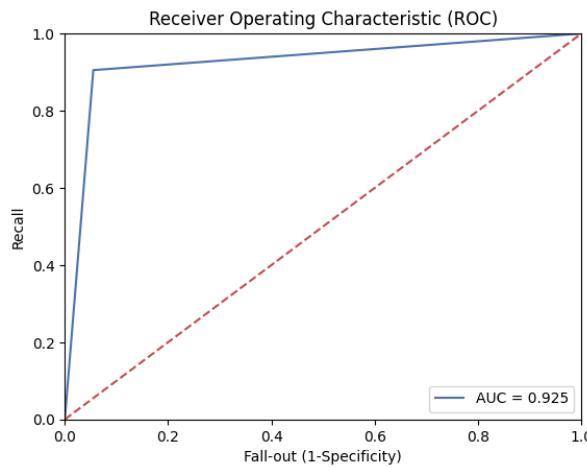


Figura 90. Área bajo la curva de modelo apilado.

Fuente: Elaboración propia.

El área bajo la Curva ROC presenta un valor aproximado de 93 %, del cual se observa en el gráfico que su sensibilidad es muy alta y su ratio de Falsa Alarma es casi nulo. De acuerdo con Britos et al. (2006), el poder discriminante del modelo es excepcionalmente bueno.

Finalmente, considerando los modelos independientes para cada modalidad, así como un trabajo previo del autor de la presente investigación cuyo trabajo sirvió de base (Puente, 2019), se armó el cuadro comparativo de la Tabla 9.

Tabla 9

Comparación de resultados de modelos propuestos con antecedentes.

Modelos		Exactitud	Precisión	Sensibilidad	Puntaje F1	AUC
Tesis de pregrado	Metainformación	0.89	0.86	0.72	0.75	0.84
	Descripción	0.75	0.59	0.53	0.35	0.68
Propuesta	Metainformación	0.93	0.91	0.92	0.92	0.92
	Descripción	0.77	0.72	0.75	0.73	0.75
	Comentarios	0.85	0.87	0.75	0.79	0.75
	The Hydra	0.93	0.91	0.93	0.92	0.93

Fuente: Elaboración propia.

Los modelos citados de la Tesis de pregrado para Metainformación y Descripción utilizaron una Máquina de Vectores de Soporte (SVM) y una SVM entrenada con el algoritmo TF-IDF, respectivamente.

Para comparar ambas investigaciones, se usaron las mismas bases de datos para las 2 modalidades, con proyectos tecnológicos de Kickstarter finalizados entre 2009 y 2019, así como también las mismas métricas para evaluar cada modelo. El tiempo de entrenamiento en el antecedente mencionado fue mayor (aproximadamente 16 segundos para la metainformación y 4 horas para la descripción), en contraste con los elaborados en este trabajo (aproximadamente 38 segundos para la metainformación y 2 horas y media para la descripción).

Como se observa en la Tabla 9, a nivel general, la performance de The Hydra fue mejor tanto contra los modelos individuales de cada modalidad (superando en más de 0.03 y más de 0.05 al modelo de comentarios y descripción respectivamente en las 5 métricas, y en 0.01 al de metainformación en AUC) como contra los modelos referenciados en los antecedentes (más de 0.05 en todas las métricas para el modelo de metainformación y más de 0.18 en todas las métricas para el modelo de descripción). El concepto (con otros modelos y variantes en el desarrollo) utilizado en la Tesis de pregrado se basó en el trabajo de los autores Cheng et al. (2019), el cual utilizó un marco de trabajo de Aprendizaje Profundo Multimodal (*Multimodal Deep Learning* en inglés), donde se combinan las características de metainformación, descripción e imagen principal del proyecto en la capa totalmente conectada. Sin embargo, en dicha ocasión no se alcanzó lograr los objetivos dado que el modelo de contenido visual presentó problemas para clasificar adecuadamente un proyecto según su estado. Esto se dio en parte a la variedad de imágenes dentro de la misma categoría Tecnología, que contiene asimismo 16 subcategorías, lo cual dificultó en su momento a la red a encontrar patrones a partir de su características. Se decidió, entonces, cambiar el criterio de reemplazar el contenido visual por otra modalidad respaldada por varios antecedentes (enunciados en la descripción del prototipo de investigación del Capítulo III) y que no fue tomada en cuenta en su momento, los comentarios realizados durante la campaña por los patrocinadores del proyecto.

5.3 Despliegue

Actividad 1: Diseñar prototipo de sistema con modelo propuesto

La última fase de la metodología CRISP-DM comienza con el diseño del prototipo que contemplará el sistema conformado por la captura de datos y la predicción del estado de financiamiento de un proyecto consultado. Para ello, previamente se deberán cargarse todos los complementos necesarios para que el modelo de Aprendizaje Profundo Multimodal funcione correctamente. Desde librerías de Python que incluyen elementos utilizados en la recolección de datos como Selenium y las librerías de Keras para la carga de las capas del modelo, hasta algoritmos usados para la limpieza de texto como NLTK y las métricas de clasificación por parte de Scikit-learn.

Luego, se diseñó el código para ejecutar el proceso de la Figura ??, el cual puede ser descargado desde el link <https://www.kaggle.com/alonsopuente/tesis-titulacion-demo>

Actividad 2: Ejecutar prototipo con proyectos tecnológicos vigentes

Una vez diseñado el proceso del prototipo, el software es ejecutado localmente desde Jupyter Notebook y recibe como dato de entrada el enlace web de un proyecto vigente en Kickstarter, representado en la Figura 91.

```
print("Por favor, ingrese el URL del proyecto que desea analizar:\n")
url_input = input()
url_input = url_input + '?'

Por favor, ingrese el URL del proyecto que desea analizar:

https://www.kickstarter.com/projects/2125914059/revopoint-pop-high-precision-3d-scanner-for-3d-printing?ref=discovery_category_ending_soon
```

Figura 91. Proyecto consultado para la demostración. Captura de pantalla: 15/02/21.

Fuente: Elaboración propia.

Uno de los experimentos hechos el 23 de enero del 2021 se realizó con la campaña vigente de ejemplo de la Figura 92.

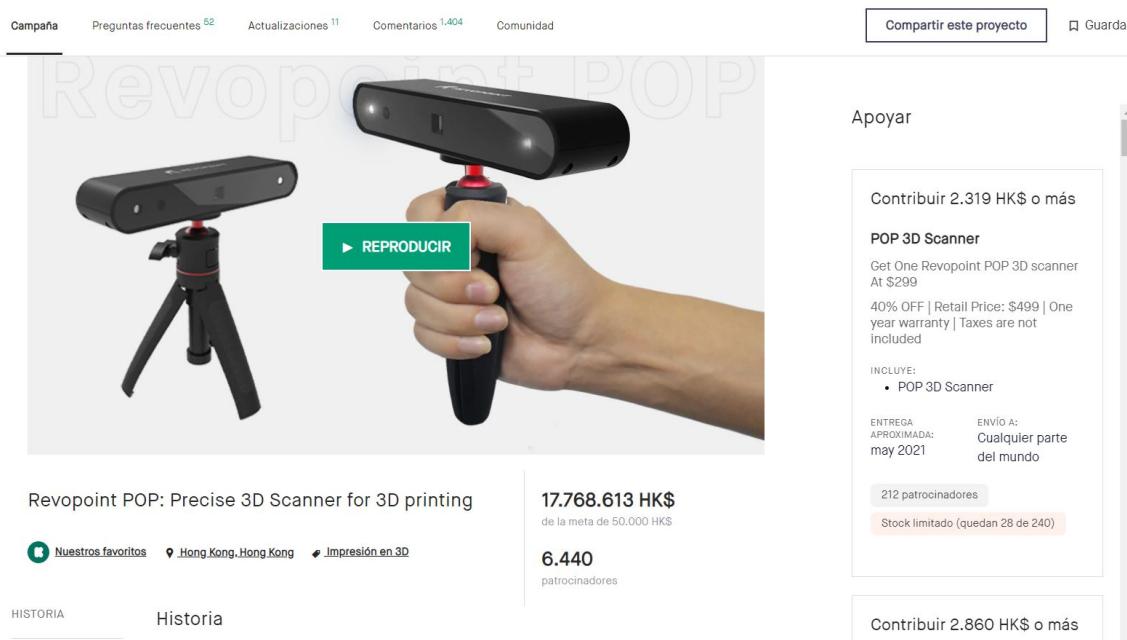


Figura 92. Campaña del proyecto consultado. Captura de pantalla: 15/02/21.

Fuente: Elaboración propia.

La primera acción hecha por el sistema fue extraer la metainformación, descripción y comentarios del proyecto desde el ingreso al URL por un navegador. Debido al cambio de políticas de acceso a la plataforma en 2020, Kickstarter detecta la presencia de bots y restringe la navegación usando CAPTCHAs para evitar su accionar. Algunas veces fue detectado el bot del sistema. Ante ello, la única acción manual por parte del usuario en el sistema se da en este paso presionando por 5 segundos el botón de “I'm a human” (Soy humano por su traducción

al español) que se muestra en la ventana. Desde este punto, luego de ingresar al enlace de la campaña del proyecto, el sistema primero se redirige a la sección de la metainformación y extrae las variables usadas para el entrenamiento del modelo.

Esta secuencia es repetida para las otras modalidades. En el caso de los comentarios, se encuentran en una sección distinta, para lo cual el sistema primero se redirige a ella a partir del dominio de la sección principal, en donde se encontraron la metainformación y la descripción.

Los datos extraídos de cada modalidad se muestran en la Figura 93 respectivamente.

Revopoint POP: Precise 3D Scanner for 3D printing
Cantidad de montos de contribución disponibles: 14
Mediana de montos de contribución disponibles: 2318800.0
Meta de la campaña: 50000 HKD
Monto contribuído en la campaña: 13741035 HKD
Porcentaje contribuído: 27482 %
Duración de la campaña: 43

(a) Metainformación

"[Is there more detail about the led light anywhere? I have some lighting already and need to know where about the light to set it if it is worth getting over my existing lighting options. Does it integrate or connect to the scanner in any way or is it set up stand alone light?]"
"Hi, I am new to UK, I have two questions. 1) Will you be shipping directly to the UK as If it is done through e.g. EU, UK bidders will have to incur additional VAT and duty on delivery. This can be 20% + fees etc. Please ship them direct to UK and avoid the extra fees. 2) I messaged you last week on Facebook messenger to claim my free markers but never got a reply."
"I am thinking of getting a scanner, what do you think?"
"I am thinking of getting a scanner, what do you think?"
"Other than painting what's needed to scan a different color (90% of the time I just can't do), how can the user solve the issue with the scanner being unable to see Black and having a very difficult time with dark colors like grays, blues, purples, browns, etc... This can be a deal breaker for me for a lot of what I am planning will be gray or black components. The CR is the best scanner I have seen, but I am open to other options."
"Hello I'm Bækcer #5358. I added 15400# to S8 marker. Please confirm. Thanks! How to get updated for the software later on?"
"You may consider a way to switch between having the two LR sensors their current distance apart and closer together for scanning items smaller than 50mm x 50mm x 50mm? It seems that there are a number of hackers that want accurate scans of small items..."
"It would be possible to accurately scan small objects with hollow inside like motorcycle intake pipe?"
"I just added \$25 for the LED light, but I can't find my backer number "anywhere"..."
"#9903 Thanks for the emails!"

(b) Descripción

(c) Comentarios

Figura 93. Variables extraídas por modalidad del proyecto consultado.

Fuente: Elaboración propia.

La información extraída es pre-procesada de la misma manera que la data usada para entrenar cada modelo. A continuación, el modelo cargado The Hydra recibe los datos procesados y concatenados para realizar la predicción. Si el umbral es por lo menos 0.50, el resultado será **EXITOSO** (*Successful* en inglés) como en la Figura 94.

Predicción de probabilidad de financiamiento: 99.86 %
Estado final de la campaña será: EXITOSO

Figura 94. Resultado de predicción de The Hydra para el proyecto consultado.

Fuente: Elaboración propia.

Capítulo VI: Conclusiones y Recomendaciones

6.1 Conclusiones

Luego de identificar y formular el problema general, plantear los objetivos y las hipótesis, se logró cumplir el objetivo de predecir el estado de financiamiento de un proyecto, ya sea detectando aquellos que podrían ser exitosos como los que podrían fracasar, bajo un nuevo criterio, solo considerar proyectos de tecnología, y un nuevo enfoque, implementar un modelo de Aprendizaje Profundo Multimodal que fue denominado «The Hydra».

Respecto a los objetivos específicos, se determinó que el análisis de las alternativas propuestas en los trabajos previos sí influyó en la selección de características y del desarrollo del marco de trabajo de la investigación, ya que se validaron algunas hipótesis de la literatura respecto al rendimiento de modelos de acuerdo a las variables, técnicas y parámetros establecidos en el desarrollo, como por ejemplo, la efectividad de modelos de redes neuronales frente a modelos convencionales de Aprendizaje Automático. Esto se puede corroborar en los resultados mostrados en la Tabla 9.

De la mencionada tabla, también se concluye que el modelo de Aprendizaje Profundo Multimodal se vio afectado por las características consideradas en su desarrollo, ya que presentó mejor rendimiento tanto contra sus submodelos como contra los modelos de tesis de pregrado evaluado por las 5 métricas, siendo 0.01 la diferencia contra el segundo mejor modelo bajo el valor AUC y 0.57 contra el peor modelo bajo la sensibilidad. Sin embargo, la desventaja de la propuesta de la investigación radica en que la ausencia o poca presencia de datos en alguna de las modalidades basadas en texto disminuye el ratio de éxito predicho para un proyecto. Esto se debe al comportamiento mencionado, con tendencia al fracaso, que se observó al probar el prototipo con proyectos de tecnología con dichas características, por ejemplo, descripción muy breve o pocos comentarios recibidos por patrocinadores.

Otra desventaja relacionada con el punto anterior fue el comportamiento de la modalidad de comentarios, ya que si bien hasta en 4 antecedentes se resalta su buena performance y el submodelo entrenado presentó niveles entre 0.75 y 0.87 en las 5 métricas evaluadas, solo el 29 % de proyectos de tecnología de este trabajo presentó comentarios, y de este universo, el 41 % fracasaron en ser financiados. Se encontró, además, una notable brecha entre el promedio de palabras de proyectos exitosos (aproximadamente 103 palabras en todos los comentarios) frente al de aquellos que fracasaron (en promedio 6 palabras en todos los comentarios). Como dato adicional, de las más de 74 mil palabras únicas en el diccionario desarrollado, se observaron términos que no fueron del todo lematizados dado a la complejidad de interpretación por parte del algoritmo ante los errores ortográficos encontrados en cada palabra (por ejemplo,

repeticiones de vocales o sustitución de consonantes), comúnmente expuestas en el lenguaje informal de las comunicaciones online. Esta observación no afectó significativamente en la etapa de entrenamiento del submodelo. Sin embargo, encontrar una manera de lidiar con ella, es decir, aumentar la valorización del procesamiento de textos sobre todo en los comentarios, sí hubiese permitido mejorar su actual rendimiento al utilizar su lema correcta que aparece con mayor frecuencia.

A nivel individual, The Hydra bajo cada métrica (desde las más usadas como la exactitud hasta aquellas más recomendadas para problemas con data desbalanceada como el puntaje F1) mantuvo niveles parejos y conllevó sin problemas su entrenamiento, pese a que tanto el modelo de descripción como de comentarios se obstaculizaron con el sobreajuste luego de muchas épocas. Entre una de las razones por las cuales ambos modelos no progresaban luego de una avanzada cantidad de épocas se encontró en el contenido textual, en especial, el de comentarios ya que la interacción social muchas veces no está sujeta a estrictas normativas de la gramática hacia los usuarios que expresan libremente su opinión. Por lo tanto, algunas palabras incorrectamente redactadas no pudieron ser lematizadas al 100% por la librería NLTK. El modelo de metainformación, en cambio, ayudó a mejorar el rendimiento del modelo apilado, ya que al combinar sus predicciones con los otros dos modelos, la nueva performance del conjunto se incrementó al evaluarse con las 5 métricas (un poco menos de 0.01 en la exactitud, precisión, sensibilidad y puntaje F1, y un poco más de 0.01 en el AUC).

A pesar de presentarse una data desbalanceada (72% proyectos fracasados y 28% exitosos), fraccionar la base total en subconjuntos de entrenamiento y pruebas de forma estratificada, es decir, mantener la distribución de 72% fracasados y 28% exitosos para cada subconjunto, y luego previo a la creación de cada modelo balancear los pesos de las clases (0.6987077585764833 para proyectos fracasados y 1.7581290322580645 para exitosos) fueron también determinantes para que los modelos eviten caer en sobreajuste tempranamente y presenten comportamientos de exactitud y pérdida en la validación cercanas al entrenamiento como se presentó en la Figura 88.

Asimismo, la evaluación de la factibilidad técnica del ambiente de desarrollo para las características del modelo de Aprendizaje Profundo Multimodal determinó la aplicabilidad de las condiciones propuestas en la literatura. Algunos de los trabajos del Capítulo II que inicialmente se tenía en mente implementar eran modelos Seq2seq o LDA. Por ejemplo, Shafqat y Byun (2019) planteó una arquitectura de este último tipo para resolver el problema de clasificación que encajaba con el marco de trabajo de la actual tesis de investigación, aplicando segmentaciones de comentarios según el tema de su contenido para luego alimentar a su sistema de recomendación de proyectos. Sin embargo, como se explica en las especificaciones de sus requerimientos para llevar a cabo estos experimentos, se necesitó tener al menos una

memoria RAM de 32 GB y una GPU potente como Nvidia GForce 1080 para llevar a cabo los experimentos con más de 504 mil comentarios filtrados provenientes de 600 proyectos de Kickstarter. Esto resultó inviable para las condiciones presentes en el entorno ya que, si bien la suscripción a Google Colab Pro permite utilizar GPU con hasta aproximadamente 26 GB de memoria, el conjunto recolectado de comentarios representó más de 10 veces (7,865 proyectos con comentarios) la cantidad mencionada con un total de más de 494 mil comentarios. De igual manera, el modelo Seq2seq implicaba el uso de una extensa memoria RAM y su otra desventaja se encontró en no poder alterar internamente una arquitectura ya modificada, es decir, modificar las capas y sus conexiones. Ante este escenario, se optó por la opción de un modelo LSTM Bidireccional, acortando el número de palabras del total de comentarios por proyecto a un valor estándar para poder diseñar la capa de incrustación correspondiente.

Finalmente, el último objetivo específico cumplido fue la implementación de una herramienta analítica en tiempo real para ayudar a los emprendedores y creadores de proyectos de tecnología en la toma de decisiones y estrategias de sus campañas. El prototipo del sistema funciona localmente en la computadora del investigador y fue puesto a prueba con al menos 1 proyecto vigente de tecnología.

6.2 Recomendaciones

Para futuros trabajos de investigación, se recomienda crear una plataforma web que contenga el prototipo del sistema descrito en la sección de despliegue del Capítulo V, que integre tanto la parte de extracción y pre-procesamiento del input como el modelo The Hydra, con una interfaz que permita al usuario aprender a utilizarla de manera autodidacta, siguiendo las buenas prácticas de experiencia del usuario y motivada por la aplicación de la extensión en Google Chrome que realizaron los autores K. Chen et al. (2013).

Para afinar el desarrollo y los resultados de este trabajo, se sugiere comenzar con continuar ajustando los hiperparámetros de los modelos de contenido textual para progresar en la etapa de entrenamiento, mediante técnicas como validación cruzada (*k-fold Cross Validation*), *Grid Search*, *Random Search*, entre otros. Además, se sugiere también buscar otras alternativas de optimizadores (por ejemplo, *RMSProp*, *SGD*) o alterar más parámetros de la opción usada *Adam*, usando otros inicializadores de kernel, entre otros, con el fin de optimizar los resultados de la predicción de los submodelos.

En caso se cuente con herramientas tecnológicas más potentes de hardware y software para el desarrollo de modelos predictivos más profundos como modelos Seq2seq, multimodales o híbridos del tipo DC-LDA, se recomienda limitar la extensión de palabras a un valor no mayor a la cantidad de comentarios presentada en el trabajo de Shafqat y Byun (2019).

Para lidiar con las oportunidades de mejora del tercer y cuarto párrafo de las conclusiones, se sugiere considerar otras variables y modalidades, como por ejemplo, la interacción social externa entre el creador y la comunidad, ya sea en la sección de comentarios como se consideró en esta investigación, así como también en las menciones del proyecto en las redes sociales para efectuar un análisis de sentimientos más profundo. Estas deberían considerarse en un segundo modelo de Aprendizaje Profundo Multimodal (como lo trabajaron los autores Shafqat y Byun (2019) en su investigación de predicción de temas en varios documentos) que luego sería concatenado con el primero, en las que el creador de la campaña interviene directamente (metainformación y descripción del proyecto), ya que la dependencia de una modalidad en la cual interviene una tercera parte (patrocinadores) podría afectar negativamente la poca o nula presencia de esta información, para lo cual se le podría asignar a este último grupo un menor peso en la fase de entrenamiento. Bajo este escenario, se podría tomar en cuenta opciones de limpieza de texto más avanzadas u otras disponibles (por ejemplo, realizar experimentos con *stemming*) para reducir lo máximo posible la aparición de palabras únicas de un mismo significado u origen pero reconocidas como distintas en el diccionario entrenado, y con esto implementar un clasificador de estados de financiamiento más sólido.

Se prefiere evitar usar la modalidad de imagen y/o video principal de la campaña, ya que como se comentó en la fase de Evaluación, en un trabajo previo desarrollado por el autor, no se encontraron características similares entre ellos para el caso particular de la categoría Tecnología. Otras alternativas potenciales también figuran la de predicción de las mejores opciones de valores que deberían contener las variables (tanto cuantitativas como cualitativas) para un proyecto antes del lanzamiento de su campaña. Este enfoque permitiría evaluar indefinidamente la información que un creador asigne a su campaña hasta encontrar la mejor combinación gracias a un modelo de recomendación.

Como penúltima sugerencia, debería seguirse alguna metodología para definir el valor del punto de corte o *threshold*, ya sea explorando más a detalle los puntos del Área bajo la curva ROC (AUC) u otra técnica, y poder implementar una clasificación más precisa.

Finalmente, como en varias referencias y libros sobre Aprendizaje Automático y Aprendizaje Profundo que se pueden encontrar, no existe una regla definida para asegurar el rendimiento excelente de cualquier modelo. El factor del logro de objetivos principalmente se debe a la continua experimentación y uso de distintas técnicas para alcanzar la performance esperada.

Referencias

- Ayana, G., Dese, K., Dereje, Y., Kebede, Y., Barki, H., Amdissa, D., Husen, N., Mulugeta, F., Habtamu, B., & Choe, S.-W. (2023). Vision-Transformer-Based Transfer Learning for Mammogram Classification. *Diagnostics*, 13(2). doi: 10.3390/diagnostics13020178.
- Beckwith, J. (2016). *Predicting Success in Equity Crowdfunding* [Tesis de grado]. Universidad de Pensilvania. Recuperado de http://repository.upenn.edu/joseph_wharton_scholars/25.
- Bhattacharya, S., Reddy Maddikunta, P. K., Pham, Q. V., Gadekallu, T. R., Krishnan S, S. R., Chowdhary, C. L., Alazab, M., & Jalil Piran, M. (2021). Deep learning and medical image processing for coronavirus (COVID-19) pandemic: A survey. *Sustainable Cities and Society*, 65. doi: 10.1016/j.scs.2020.102589.
- Binboga, S., Gemici, E., & Binboga, E. (2019). Thyroid Anatomy. En *Knowledges on Thyroid Cancer* (pp. 1-11). IntechOpen.
- Britos, P. V., García Martínez, R., Hossian, A., & Sierra, E. (2006). *Minería de Datos*. Nueva Librería.
- Brownlee, J. (2017). *How to Develop a Bidirectional LSTM For Sequence Classification in Python with Keras*. Machine Learning Mastery. Recuperado de <https://machinelearningmastery.com/develop-bidirectional-lstm-sequence-classification-python-keras/>.
- Brownlee, J. (2019). *How to Fix the Vanishing Gradients Problem Using the ReLU*. Machine Learning Mastery. Recuperado de <https://machinelearningmastery.com/how-to-fix-vanishing-gradients-using-the-rectified-linear-activation-function/>.
- Chaichi, N., & Anderson, T. (2019). Deploying Natural Language Processing to Extract Key Product Features of Crowdfunding Campaigns: The Case of 3D Printing Technologies on Kickstarter. *2019 Portland International Conference on Management of Engineering and Technology (PICMET)*, 1-9. doi: 10.23919/picmet.2019.8893839.
- Chen, K., Jones, B., Kim, I., & Schlamp, B. (2013). *KickPredict: Predicting Kickstarter Success* (Reporte técnico). Instituto de Tecnología de California. California, Estados Unidos de América. Recuperado de <http://courses.cms.caltech.edu/cs145/2013/blue.pdf>.
- Chen, L.-S., & Shen, E.-L. (2019). Finding the Keywords Affecting the Success of Crowdfunding Projects. *2019 IEEE 6th International Conference on Industrial Engineering and Applications (ICIEA)*, 567-571. doi: 10.1109/iea.2019.8714815.
- Chen, S.-Y., Chen, C.-N., Chen, Y.-R., Yang, C.-W., Lin, W.-C., & Wei, C.-P. (2015). Will Your Project Get the Green Light? Predicting the Success of Crowdfunding Campaigns. *Pacific Asia Conference on Information Systems (PACIS) 2015*, 79. Recuperado de <http://aisel.aisnet.org/pacis2015/79>.

- Cheng, C., Tan, F., Hou, X., & Wei, Z. (2019). Success Prediction on Crowdfunding with Multimodal Deep Learning. *Twenty-Eighth International Joint Conference on Artificial Intelligence*, 2158-2164. doi: 10.24963/ijcai.2019/299.
- Ciaburro, G., & Joshi, P. (2019). *Python Machine Learning Cookbook* (2^a ed.). Recuperado de https://subscription.packtpub.com/book/big_data_and_business_intelligence/9781789808452/1/ch01lvl1sec15/data-scaling.
- Deng, P., Han, X., Wei, X., & Chang, L. (2022). Automatic classification of thyroid nodules in ultrasound images using a multi-task attention network guided by clinical knowledge. *Computers in Biology and Medicine*, 150. doi: 10.1016/j.compbiomed.2022.106172.
- Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N. (2021). AN IMAGE IS WORTH 16X16 WORDS: TRANSFORMERS FOR IMAGE RECOGNITION AT SCALE. *ICLR 2021 - 9th International Conference on Learning Representations*.
- Felgueiras Carvalho, A. R. (2019). *3D Lung Nodule Classification in Computed Tomography Images* [Tesis de maestría]. Universidad de Oporto. Recuperado de <https://repositorio-aberto.up.pt/handle/10216/123391>.
- Fernandez-Blanco, A., Villanueva-Balsara, J., Rodriguez-Montequin, V., & Moran-Palacios, H. (2020). Key Factors for Project Crowdfunding Success: An Empirical Study. *Sustainability*, 12(2), 599. doi: 10.3390/su12020599.
- Géron, A. (2022). *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow* (2^a ed.). O'Reilly Media.
- 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.
- Herrera Gajardo, J. I. (2017). *Diseño e implementación de técnicas de procesamiento de imágenes para dispositivo de ultrasonido portátil* [Tesis de pregrado]. Universidad de Chile. Recuperado de <https://repositorio.uchile.cl/handle/2250/148434>.
- Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2261-2269. doi: 10.1109/cvpr.2017.243.

- Hurbans, R. (2020). *Grokking Artificial Intelligence Algorithms*. Manning Publications.
- 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.
- Jin, B., Zhao, H., Chen, E., Liu, Q., & Ge, Y. (2019). Estimating the Days to Success of Campaigns in Crowdfunding: A Deep Survival Perspective. *33rd AAAI Conference on Artificial Intelligence (AAAI'2019)*, 33, 4023-4030. doi: 10.1609/aaai.v33i01.33014023.
- Kamath, R. S., & Kamat, R. K. (2018). Supervised Learning Model For Kickstarter Campaigns With R Mining. *International Journal of Information Technology, Modeling and Computing (IJITMC)*, 4(1). doi: 10.5281/zenodo.1228716.
- Kang, Q., Lao, Q., Li, Y., Jiang, Z., Qiu, Y., Zhang, S., & Li, K. (2022). Thyroid nodule segmentation and classification in ultrasound images through intra- and inter-task consistent learning. *Medical Image Analysis*, 79. doi: 10.1016/j.media.2022.102443.
- Kaur, H., & Gera, J. (2017). Effect of Social Media Connectivity on Success of Crowdfunding Campaigns. *Procedia Computer Science*, 122, 767-774. doi: 10.1016/j.procs.2017.11.435.
- Keras. (s.f.). *Model training APIs*. Recuperado de https://keras.io/api/models/model_training_apis/.
- Kim, J., Gosnell, J. E., & Roman, S. A. (2020). Geographic influences in the global rise of thyroid cancer. *Nature Reviews Endocrinology*, 16(1), 17-29. doi: 10.1038/s41574-019-0263-x.
- Lee, S., Lee, K., & Kim, H.-c. (2018). Content-based Success Prediction of Crowdfunding Campaigns: A Deep Learning Approach. *Companion of the 2018 ACM Conference on Computer Supported Cooperative Work and Social Computing*, 193-196. doi: 10.1145/3272973.3274053.
- Li, Y., Rakesh, V., & Reddy, C. K. (2016). Project Success Prediction in Crowdfunding Environments. *Ninth ACM International Conference on web search and data mining*, 247-256. doi: 10.1145/2835776.2835791.
- Malik, U. (2019). *Python for NLP: Movie Sentiment Analysis using Deep Learning in Keras*. Recuperado de <https://stackabuse.com/python-for-nlp-movie-sentiment-analysis-using-deep-learning-in-keras/>.
- Manzari, O. N., Ahmadabadi, H., Kashiani, H., Shokouhi, S. B., & Ayatollahi, A. (2023). MedViT: A robust vision transformer for generalized medical image classification. *Computers in Biology and Medicine*, 157. doi: 10.1016/j.compbiomed.2023.106791.

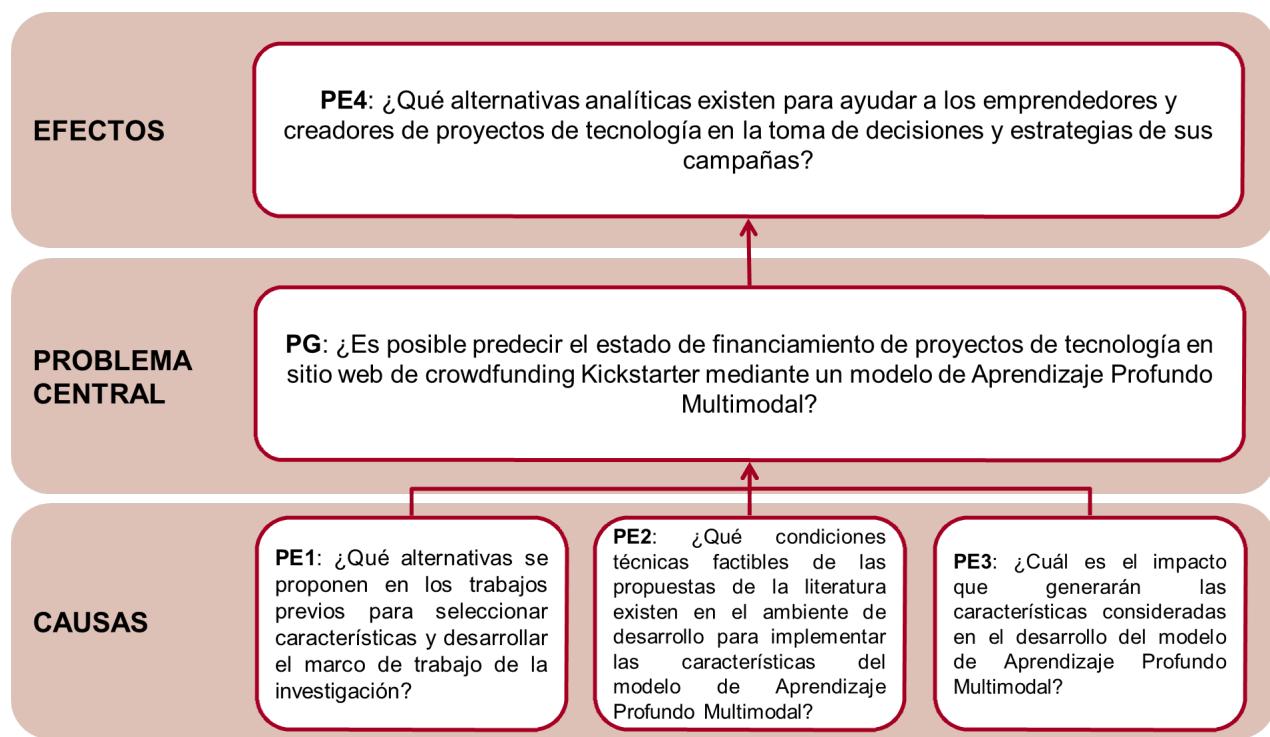
- Mitra, T., & Gilbert, E. (2014). The Language that Gets People to Give: Phrases that Predict Success on Kickstarter. *17th ACM conference on Computer supported cooperative work & social computing*, 49-61. doi: 10.1145/2531602.2531656.
- Monroy Malca, V. d. P. (2021). *Diseño de un sistema de visión 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>.
- Moreira Aresta, G. (2021). *Detection of lung nodules in computed tomography images* [Tesis de grado]. Universidad de Oporto. Recuperado de <https://repositorio-aberto.up.pt/bitstream/10216/134135/2/473080.pdf>.
- Moroney, L. (2020). *AI and Machine Learning for Coders*. O'Reilly Media. Recuperado de <https://books.google.com.pe/books?id=gw4CEAAAQBAJ>.
- Mukaka, M. M. (2012). Statistics Corner: A guide to appropriate use of Correlation coefficient in medical research. *Malawi Medical Journal*, 24(3), 69-71. Recuperado de https://www.researchgate.net/publication/236604665_Statistics_Corner_A_guide_to_appropriate_use_of_Correlation_coefficient_in_medical_research.
- 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.
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12, 2825-2830.
- Puente, A. (2019). *Predicción del estado de financiamiento de proyectos de tecnología en web de crowdfunding Kickstarter mediante modelo(s) de Aprendizaje Automático* [Tesis de pregrado]. Universidad ESAN.
- Ranjan, C. (2019, julio). *Rules-of-thumb for building a Neural Network*. Towards Data Science. Recuperado de <https://towardsdatascience.com/17-rules-of-thumb-for-building-a-neural-network-93356f9930af>.
- Real Academia Española. (s.f.). Hidra. Recuperado de <https://dle.rae.es/hidra>.
- Regmi, S., Subedi, A., Bagci, U., & Jha, D. (2023). Vision transformer for efficient chest X-ray and gastrointestinal image classification. doi: 10.48550/arxiv.2304.11529.
- Sampieri, R. H., Collado, C. F., & Lucio, P. B. (2014). *Metodología de la investigación* (6^a ed.). McGraw-Hill Education. Recuperado de <https://books.google.com.pe/books?id=oLbjQEACAAJ>.
- Sawhney, K., Tran, C., & Tuason, R. (2016). *Using Language to Predict Kickstarter Success* (Reporte técnico). Universidad Standford. California, Estados Unidos de América. Recuperado de <https://stanford.edu/~kartiks2/kickstarter.pdf>.

- Scikit-learn. (s.f.). *sklearn.preprocessing.MinMaxScaler*. Recuperado de <https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html>.
- Shafqat, W., & Byun, Y.-C. (2019). Topic Predictions and Optimized Recommendation Mechanism Based on Integrated Topic Modeling and Deep Neural Networks in Crowdfunding Platforms. *Applied Sciences*, 9(24), 5496. doi: 10.3390/app9245496.
- Shin, J. H., Baek, J. H., Chung, J., Ha, E. J., Kim, J.-H., Lee, Y. H., Lim, H. K., Moon, W.-J., Na, D. G., Park, J. S., Choi, Y. J., Hahn, S. Y., Jeon, S. J., Jung, S. L., Kim, D. W., Kim, E.-K., Kwak, J. Y., Lee, C. Y., Lee, H. J., ... Sung, J. Y. (2016). Ultrasonography Diagnosis and Imaging-Based Management of Thyroid Nodules: Revised Korean Society of Thyroid Radiology Consensus Statement and Recommendations. *Korean Journal of Radiology*, 17(3), 370-395. doi: 10.3348/kjr.2016.17.3.370.
- 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>.
- Singh, S. P., Wang, L., Gupta, S., Goli, H., Padmanabhan, P., & Gulyás, B. (2020). 3d deep learning on medical images: A review. *Sensors (Switzerland)*, 20(18), 1-24. doi: 10.3390/s20185097.
- Sun, J., Li, C., Lu, Z., He, M., Zhao, T., Li, X., Gao, L., Xie, K., Lin, T., Sui, J., Xi, Q., Zhang, F., & Ni, X. (2022). TNSNet: Thyroid nodule segmentation in ultrasound imaging using soft shape supervision. *Computer Methods and Programs in Biomedicine*, 215. doi: 10.1016/j.cmpb.2021.106600.
- Sun, J., Wu, B., Zhao, T., Gao, L., Xie, K., Lin, T., Sui, J., Li, X., Wu, X., & Ni, X. (2023). Classification for thyroid nodule using ViT with contrastive learning in ultrasound images. *Computers in Biology and Medicine*, 152. doi: 10.1016/j.combiomed.2022.106444.
- Supanta Zapata, C. A. (2021). *Desarrollo de un algoritmo orientado a la detección y extracción de características de nódulos pulmonares en imágenes radiográficas digitales* [Tesis de pregrado]. Universidad de San Martín de Porres. Recuperado de <https://hdl.handle.net/20.500.12727/7556>.
- Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. (2015). Going deeper with convolutions. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1-9. doi: 10.1109/cvpr.2015.7298594.
- Tampu, I. E., Eklund, A., Johansson, K., Gimm, O., & Haj-Hosseini, N. (2023). Diseased thyroid tissue classification in OCT images using deep learning: Towards surgical decision support. *Journal of Biophotonics*, 16(2). doi: 10.1002/jbio.202200227.

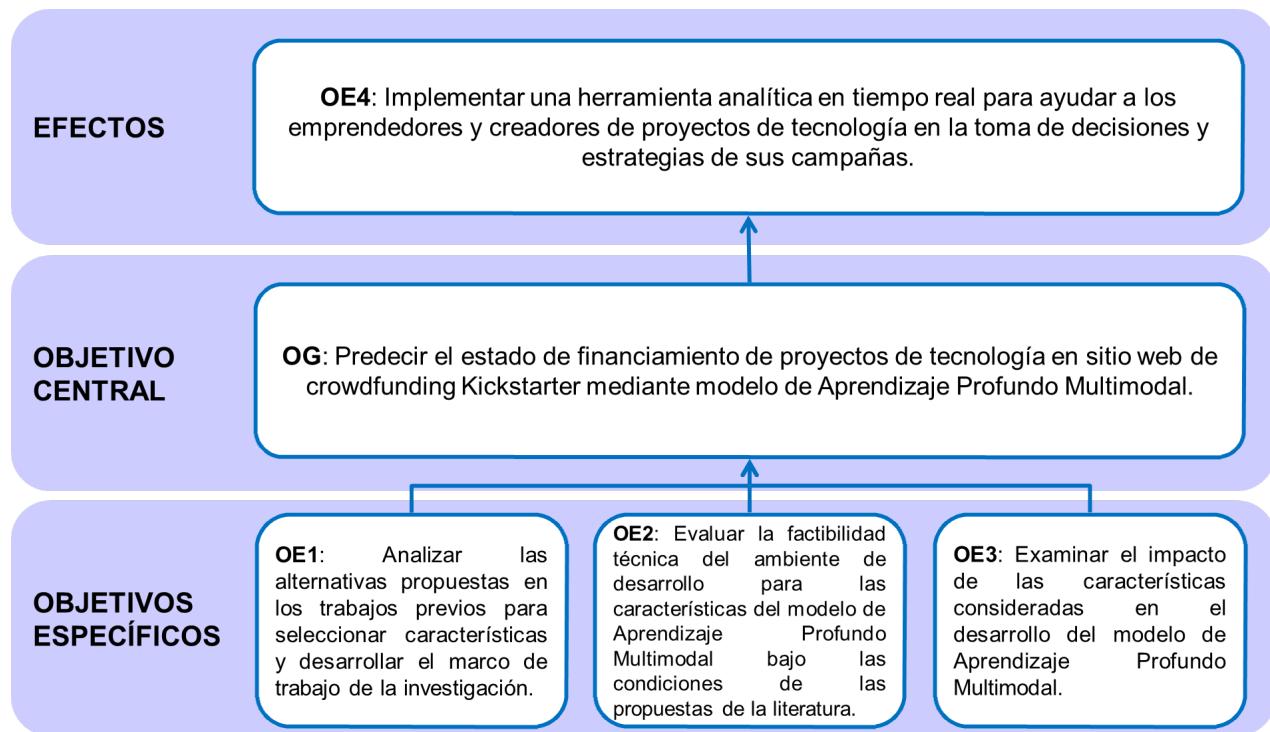
- Wang, L., Zhang, L., Zhu, M., Qi, X., & Yi, Z. (2020). Automatic diagnosis for thyroid nodules in ultrasound images by deep neural networks. *Medical Image Analysis*, 61. doi: 10.1016/j.media.2020.101665.
- Web Robots. (2019). Kickstarter Datasets. Recuperado de <https://webrobots.io/kickstarter-datasets/>.
- Yang, G., Luo, S., & Greer, P. (2023). A Novel Vision Transformer Model for Skin Cancer Classification. *Neural Processing Letters*, 55(7), 9335-9351. doi: 10.1007/s11063-023-11204-5.
- Yu, P.-F., Huang, F.-M., Yang, C., Liu, Y.-H., Li, Z.-Y., & Tsai, C.-H. (2018). Prediction of Crowdfunding Project Success with Deep Learning. *2018 IEEE 15th International Conference on e-Business Engineering (ICEBE)*, 1-8. doi: 10.1109/icebe.2018.00012.
- Yuan, H., Lau, R. Y., & Xu, W. (2016). The Determinants of Crowdfunding Success: A Semantic Text Analytics Approach. *Decision Support Systems*, 91, 67-76. doi: 10.1016/j.dss.2016.08.001.
- Zhang, J., Mazurowski, M. A., Allen, B. C., & Wildman-Tobriner, B. (2023). Multistep Automated Data Labelling Procedure (MADLaP) for thyroid nodules on ultrasound: An artificial intelligence approach for automating image annotation. *Artificial Intelligence in Medicine*, 141. doi: 10.1016/j.artmed.2023.102553.
- Zhou, M., Zhang, X., Wang, A. G., Du, Q., Qiao, Z., & Fan, W. (2015). Money Talks: A Predictive Model on Crowdfunding Success Using Project Description. *Twenty-first Americas Conference on Information Systems, Puerto Rico, 2015*. Recuperado de <http://aisel.aisnet.org/amcis2015/BizAnalytics/GeneralPresentations/37>.
- Zhu, Y.-C., AlZoubi, A., Jassim, S., Jiang, Q., Zhang, Y., Wang, Y.-B., Ye, X.-D., & DU, H. (2021). A generic deep learning framework to classify thyroid and breast lesions in ultrasound images. *Ultrasonics*, 110. doi: 10.1016/j.ultras.2020.106300.
- Zimovnov, A. (2018). *NLP - Text Preprocessing and Text Classification (using Python)* [Archivo de video]. Recuperado de <https://youtu.be/nxhCyeRR75Q>.

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 la clasificación de nódulos tiroideos a través de imágenes de ultrasonido			
Problema General	Objetivo General	Hipótesis General	Variables	Método
¿Es posible implementar un modelo de Deep Learning para el diagnóstico de nódulos tiroideos a través de imágenes de ultrasonido?	Diseñar un modelo de Deep Learning para el diagnóstico de nódulos tiroideos a través de imágenes de ultrasonido.	A través del diseño de un modelo de Deep Learning se logra diagnosticar los nódulos tiroideos a través de imágenes de ultrasonido.		
Problemas Específicos	Objetivos Específicos	Hipótesis Específicas		
¿Qué características del conjunto de datos a usar para entrenar y evaluar los modelos son ideales para obtener buenos resultados?	Determinar las características ideales del conjunto de datos a usar para entrenar y evaluar los modelos.	Las mejores características del conjunto de datos son usadas para entrenar y evaluar modelos permiten un mejor desarrollo del modelo de Deep Learning.	Dependiente: Diagnóstico de nódulos tiroideos Independiente: Modelo de Deep Learning	Tipo de Investigación: Experimental. Alcance de la investigación: Explicativo

¿Qué técnicas de preprocesamiento se deberían aplicar a las imágenes de ultrasonido del conjunto de datos?

Determinar las técnicas de preprocesamiento que se deben aplicar a las imágenes de ultrasonido del conjunto de datos.

Las técnicas de preprocesamiento a aplicar a las imágenes de ultrasonido del conjunto de datos mejoran el desempeño del modelo de Deep Learning.

¿De qué forma se puede evaluar el rendimiento de los modelos de Deep Learning en la clasificación de imágenes de ultrasonido?

Identificar las métricas de evaluación de rendimiento de los modelos de Deep Learning en la clasificación de imágenes de ultrasonido.

Las métricas de evaluación de rendimiento de los modelos de Deep Learning logran una correcta comparación de modelos en la tarea de clasificación de imágenes de ultrasonido.

¿Qué arquitecturas de Deep Learning tienen el más alto desempeño en la clasificación de imágenes de ultrasonido de la tiroides?

Determinar las arquitecturas de Deep Learning que tienen el más alto desempeño en la clasificación de imágenes de ultrasonido de la tiroides.

Las arquitecturas de Deep Learning con más alto desempeño en la clasificación de imágenes de ultrasonido demuestran superioridad frente a los demás modelos.

D Comparación de metodologías de antecedentes

Autor	Título de la Investigación	Metodología	Grupo
Chen, Jones, Kim & Schlamp	KickPredict: Predicting Kickstarter Success	<ul style="list-style-type: none"> – Recolección de datos. – Modelamiento. – Evaluación. – Despliegue. 	GG
Mitra & Gilbert	The Language that Gets People to Give: Phrases that Predict Success on Kickstarter	<ul style="list-style-type: none"> – Recolección de datos. – Pre-procesamiento de datos. – Modelamiento. – Evaluación. – Despliegue. 	GE
Zhou, Zhang, Wang, Du, Qiao & Fan	Money Talks: A Predictive Model on Crowdfunding Success Using Project Description	<ul style="list-style-type: none"> – Formulación del problema. – Recolección de datos. – Pre-procesamiento de datos. – Modelamiento. – Evaluación. 	GD
Chen, Chen, Chen, Yang, Lin & Wei	Will Your Project Get the Green Light? Predicting the Success of Crowdfunding Campaigns	<ul style="list-style-type: none"> – Recolección de datos. – Pre-procesamiento de datos. – Modelamiento. – Evaluación. 	GF
Beckwith	Predicting Success in Equity Crowdfunding	<ul style="list-style-type: none"> – Recolección de datos. – Pre-procesamiento de datos. – Modelamiento. – Evaluación. 	GF
Li, Rakesh & Reddy	Project Success Prediction in Crowdfunding Environments	<ul style="list-style-type: none"> – Formulación del problema. – Recolección de datos. – Pre-procesamiento de datos. – Modelamiento. – Evaluación. 	GD

		– Recolección de datos.
Yuan, Lau & Xu	The Determinants of Crowdfunding Success: A Semantic Text Analytics Approach	– Pre-procesamiento de datos. – Modelamiento. – Evaluación. – Despliegue.
Sawhney, Tran & Tuason	Using Language to Predict Kickstarter Success	– Recolección de datos. – Modelamiento. – Evaluación.
Kaur & Gera	Effect of Social Media Connectivity on Success of Crowdfunding Campaigns	– Recolección de datos. – Pre-procesamiento de datos. – Modelamiento. – Evaluación.
Kamath & Kamat	Supervised Learning Model For Kickstarter Campaigns With R Mining	– Formulación del problema. – Recolección y pre- procesamiento de datos. – Exploración y extracción de datos. – Modelamiento. – Evaluación.
Yu, Huang, Yang, Liu, Li & Tsai	Prediction of Crowdfunding Project Success with Deep Learning	– Recolección de datos. – Exploración de datos. – Pre-procesamiento de datos. – Modelamiento. – Evaluación.
Lee, Lee & Kim	Content-based Success Prediction of Crowdfunding Campaigns: A Deep Learning Approach	– Recolección de datos. – Modelamiento. – Evaluación.

Jin, Zhao, Chen, Liu & Ge	Estimating the Days to Success of Campaigns in Crowdfunding: A Deep Survival Perspective	<ul style="list-style-type: none"> – Formulación del problema. – Recolección de datos. – Pre-procesamiento de datos. – Modelamiento. – Evaluación. – Despliegue. 	GA
Cheng, Tan, Hou & Wei	Success Prediction on Crowdfunding with Multimodal Deep Learning	<ul style="list-style-type: none"> – Formulación del problema. – Recolección de datos. – Pre-procesamiento de datos. – Modelamiento. – Evaluación. – Despliegue. 	GA
Chen & Shen	Finding the Keywords Affecting the Success of Crowdfunding Projects	<ul style="list-style-type: none"> – Recolección de datos. – Pre-procesamiento de datos. – Selección de características. – Modelamiento. – Evaluación. 	GC
Chaichi & Anderson	Deploying Natural Language Processing to Extract Key Product Features of Crowdfunding Campaigns: The Case of 3D Printing Technologies on Kickstarter	<ul style="list-style-type: none"> – Recolección de datos. – Pre-procesamiento de datos. – Selección de características. – Modelamiento. – Evaluación. 	GC
Shafqat & Byun	Topic Predictions and Optimized Recommendation Mechanism Based on Integrated Topic Modeling and Deep Neural Networks in Crowdfunding Platforms	<ul style="list-style-type: none"> – Formulación del problema. – Recolección de datos. – Selección de características. – Modelamiento. – Evaluación. – Despliegue. 	GB

Fernández-		
Blanco,		
Villanueva-	Key Factors for Project	
Balsera,	Crowdfunding Success: An	CRISP-
Rodriguez-	Empirical Study	DM
Montequin &		
Moran-Palacios		

E Comparación de objetivos específicos de antecedentes

Año	Autor	Publicación	Título de la Investigación	Objetivo General	Objetivos Específicos	Item
2013	Chen, Jones, Kim, Schlamp	Reporte técnico	KickPredict: Predicting Kickstarter Success	Desarrollar un sistema para predecir si un proyecto de Kickstarter será financiado exitosamente o no antes de su culminación.	Desarrollar aplicaciones en Android y Google Chrome para predecir en tiempo real el estado de financiamiento y el porcentaje.	OE4
					Analizar las características más importantes que influyen en el éxito de financiamiento a partir de las propiedades de los proyectos.	OE3
					Estudiar el impacto de contribuciones durante el transcurso de la campaña en el estado de financiamiento.	
2014	Mitra, Gilbert	Acta de conferencia	The Language that Gets People to Give: Phrases that Predict Success on Kickstarter	Determinar los factores que conducen a financiar con éxito un proyecto de crowdfunding.	Desarrollar frases predictivas junto con variables de control para ayudar a posteriores estudios y creadores de proyectos.	OE4
					Evaluar el impacto del uso de ciertas frases de la descripción en el éxito de financiamiento.	OE3

	Zhou, Zhang, Wang, Du, Qiao, Fan	Acta de conferencia	Money Talks: A Predictive Model on Crowdfunding Success Using Project Description	Estudiar la influencia de las descripciones de proyectos en el éxito de financiamiento de proyectos crowdfunding.	Examinar el impacto de la calidad argumentativa y fuente de credibilidad de la descripción de un proyecto en su performance durante la campaña. Evaluar e implementar características previamente consideradas en trabajos previos.	OE3 OE1
2015	Chen, Chen, Chen, Yang, Lin, Wei	Acta de conferencia	Will Your Project Get the Green Light? Predicting the Success of Crowdfunding Campaigns	Predecir si una campaña de crowdfunding tendrá éxito a través de extracción y posterior uso de características estáticas y dinámicas.	Analizar relación entre exactitud del modelo y el uso de características dinámicas y/o estáticas.	OE3
2015	Beckwith	Tesis de grado	Predicting Success in Equity Crowdfunding	Determinar la relación entre las características de una empresa y su capacidad para recaudar fondos en una plataforma de crowdfunding de capital.	Evaluando efectividad de predicción en distintas etapas de campaña. Evaluar características estáticas y dinámicas de trabajos previos.	OE1

2016	Li, Rakesh, Reddy	Acta de conferencia	Project Success Prediction in Crowdfunding Environments	Formular la predicción del éxito del proyecto como un problema de análisis de supervivencia y aplicar el enfoque de regresión censurada.	Estudiar la distribución del tiempo de éxito del proyecto de los datos de crowdfunding. Demostrar que el desempeño de los modelos con proyectos exitosos y fracasados es mejor que aquellos que solo comprenden exitosos.
2016	Yuan, Lau, Xu	Artículo	The Determinants of Crowdfunding Success: A Semantic Text Analytics Approach	Implementar un marco de análisis textual para analizar y predecir el éxito de recaudación de proyectos de crowdfunding.	Identificar las características de temas a partir de las descripciones. Identificar las características discriminatorias que influyen en el éxito de financiamiento de proyectos.
2016	Sawhney, Tran, Tuason	Reporte técnico	Using Language to Predict Kickstarter Success	Predecir el éxito de una campaña a partir de su contenido, características lingüísticas y metainformación.	Ayudar a emprendedores a identificar las características textuales más influyentes que afectan los resultados de la recaudación de fondos. OE4 Estudiar relación entre información inicial de campaña (sin incluir variables del tiempo o externas) y estado de financiamiento.

2017	Kaur, Gera	Artículo	Effect of Social Media Connectivity on Success of Crowdfunding Campaigns	Analizar la relación entre la conectividad de las redes sociales y el desempeño de una campaña de crowdfunding.	Evaluar la correlación entre variables de conectividad y variables principales de la campaña. Examinar el impacto de variables principales de la en el desempeño del modelo predictivo. OE3
2018	Kamath, Kamat	Artículo	Supervised Learning Model For Kickstarter Campaigns With R Mining	Implementar un sistema con técnicas de Aprendizaje Automático aplicadas al conjunto de datos de campañas de Kickstarter para clasificar proyectos.	Evaluando el entorno técnico para el desarrollo de fases de metodología. OE2 Analizar propuestas de la literatura para el diseño del marco de trabajo de la investigación. OE1
2018	Yu, Huang, Yang, Liu, Li, Tsai	Acta de conferencia	Prediction of Crowdfunding Project Success with Deep Learning	Desarrollar un modelo de Aprendizaje Profundo para predecir el éxito de un proyecto de crowdfunding.	Determinar el impacto de las propiedades del proyecto para predecir su éxito de financiamiento. OE3 Analizar alternativas de algoritmos de Aprendizaje Automático con conjunto de datos utilizado.

				Implementar modelo más rápido, preciso y eficiente de recursos que línea base.
2018	Lee, Lee, Kim	Acta de conferencia	Content-based Success Prediction of Crowdfunding Campaigns: A Deep Learning Approach	Predecir el estado de financiamiento de proyectos de tecnología con DNN utilizando solo el contenido textual de los proyectos. Alcanzar nivel de rendimiento del estado del arte de por lo menos 89-91 % de exactitud.
2019	Jin, Zhao, Chen, Liu, Ge	Acta de conferencia	Estimating the Days to Success of Campaigns in Crowdfunding: A Deep Survival Perspective	Predecir la distribución de promesas y la duración para lograr el éxito de financiamiento implementando un modelo Seq2seq con arquitectura SMP. Estimar valor de variable dependiente en cualquier momento de la campaña con modelo adaptado a nuevos datos.
				Identificar la distribución de las contribuciones de acuerdo a la evolución del tiempo (días) de la campaña. Identificar el tiempo correcto de duración que debe tener la campaña a partir del análisis de supervivencia. Ayudar a los creadores de proyectos a identificar características relevantes para lograr alcanzar el financiamiento exitoso de sus proyectos.

OE4

2019	Cheng, Tan, Hou, Wei	Acta de conferencia	Success Prediction on Crowdfunding with Multimodal Deep Learning	Estudiar la influencia de interacciones sofisticadas entre modalidades textuales, visuales y metainformación en la predicción de éxito de proyectos.	Investigar esquemas de fusión con diferentes modalidades y evaluar arquitectura multimodal en el con- junto de datos de crowdfunding. OE1
2019	Chen, Shen	Acta de conferencia	Finding the Keywords Affecting the Success of Crowdfunding Projects	Analizar el impacto del contenido textual en un proyecto de Kickstarter a partir del análisis de sus palabras clave que determinen el éxito de financiamiento.	Investigar la contribución de imáge- nes al éxito del proyecto. OE3
2019	Chaichi, Anderson	Acta de conferencia	Deploying Natural Language Processing to Extract Key Product Features of Crowdfunding Campaigns: The Case of 3D Printing Technologies on Kickstarter	Analizar impacto de la predicción temprana en los resultados. OE4	Analizar alternativas de clasifica- ción a partir de distinta selección de características en trabajos previos. OE1

2019	Shafqat, Byun	Artículo	Topic Predictions and Optimized Recommendation Mechanism Based on Integrated Topic Modeling and Deep Neural Networks in Crowdfunding Platforms	Desarrollar un sistema integrado de recomendación de proyectos de crowdfunding a partir del análisis textual de sus comentarios.	Identificar potenciales proyectos fraudulentos analizando tendencias de discusión en los comentarios. Ayudar a encontrar proyectos seguros a inversores a partir de sistema OE4 de recomendación.
	Fernández-Blanco, Villanueva-				Diseñar proceso de arquitectura integrada de modelos de recomendación y de predicción de tendencia de temas de comentarios.
	Balsera, Rodríguez-Montequín, Moran-Palacios	Artículo	Key Factors for Project Crowdfunding Success: An Empirical Study	Evaluar ambiente de implementación y experimentación de modelos propuestos.	Evaluación ambiental de implementación y experimentación de modelos OE2
					Determinar la relación entre la distribución de variables entre clústeres.
					Estimar qué grupo potencial desarrollado sería el resultado de un nuevo proyecto.
					Ayudar al creador a definir una estrategia o reorientar un proyecto para llevarlo al éxito, en función de su posición en el sistema.
					OE4

F Parámetros para modelo predictivo de Metainformación

Hiperparámetro	Valor
Pesos de clases	<ul style="list-style-type: none"> – Exitoso (1): 1.7581290322580645 – Fracasado (0): 0.6987077585764833
Optimizador	<ul style="list-style-type: none"> – Nombre: ADAM – Ratio de aprendizaje: $5 * 10^{-3}$ – Ratio de decaimiento: $1 * 10^{-5}$
Parada temprana	<ul style="list-style-type: none"> – Monitor: <i>val_accuracy</i> – Modo: <i>auto</i> – Paciencia: 10
Reducción de ratio de aprendizaje	<ul style="list-style-type: none"> – Monitor: <i>val_accuracy</i> – Modo: <i>max</i> – Paciencia: 3 – Factor: 0.1
Punto de guardado del modelo	<ul style="list-style-type: none"> – Monitor: <i>val_loss</i> – Modo: <i>min</i>
Función de pérdida	<i>binary_crossentropy</i>
Número de épocas	100
Tamaño de lote	128
Datos de entrada	<i>X_train</i>
Datos de destino	<i>y_train</i>
Datos de validación	<i>X_test</i> , <i>y_test</i>
Longitud de datos de entrada - capa de entrada	6
Número de neuronas de salida - capa densa 1	32
Función de activación - capa densa 1	<i>relu</i>
Ratio de capa de desactivación 1	0.25
Número de neuronas de salida - capa densa 2	16
Función de activación - capa densa 2	<i>tanh</i>
Ratio de capa de desactivación 2	0.30
Función de activación - capa final	<i>sigmoid</i>

G Parámetros para modelo predictivo de Descripción

Hiperparámetro	Valor
Pesos de clases	<ul style="list-style-type: none"> – Exitoso (1): 1.7581290322580645 – Fracasado (0): 0.6987077585764833
Optimizador	<ul style="list-style-type: none"> – Nombre: ADAM – Ratio de aprendizaje: $1 * 10^{-5}$ – Ratio de decaimiento: $1 * 10^{-5}$
Parada temprana	<ul style="list-style-type: none"> – Monitor: <i>val_loss</i> – Modo: <i>min</i> – Paciencia: 10
Reducción de ratio de aprendizaje	<ul style="list-style-type: none"> – Monitor: <i>val_accuracy</i> – Modo: <i>max</i> – Paciencia: 5 – Factor: 0.01
Punto de guardado del modelo	<ul style="list-style-type: none"> – Monitor: <i>val_loss</i> – Modo: <i>min</i>
Función de pérdida	<i>binary_crossentropy</i>
Número de épocas	100
Tamaño de lote	128
Datos de entrada	<i>training_sentences</i>
Datos de destino	<i>training_labels</i>
Datos de validación	<i>testing_sentences</i> , <i>testing_labels</i>
Longitud de datos de entrada - capa de entrada	3,671
Tamaño de vocabulario - capa Embedding	148,270
Dimensión de incrustación - capa Embedding	100
Número de filtros - capa Conv1D	128
Tamaño de kernel - capa Conv1D	5
Número de neuronas de salida - capa densa 1	64
Función de activación - capa densa 1	<i>tanh</i>
Función de activación - capa final	<i>sigmoid</i>

H Parámetros para modelo predictivo de Comentarios

Hiperparámetro	Valor
Pesos de clases	<ul style="list-style-type: none"> – Exitoso (1): 1.7581290322580645 – Fracasado (0): 0.6987077585764833
Optimizador	<ul style="list-style-type: none"> – Nombre: ADAM – Ratio de aprendizaje: $1 * 10^{-3}$ – Ratio de decaimiento: $1 * 10^{-6}$
Parada temprana	<ul style="list-style-type: none"> – Monitor: <i>val_loss</i> – Modo: <i>min</i> – Paciencia: 10
Reducción de ratio de aprendizaje	<ul style="list-style-type: none"> – Monitor: <i>val_accuracy</i> – Modo: <i>max</i> – Paciencia: 5 – Factor: 0.01
Punto de guardado del modelo	<ul style="list-style-type: none"> – Monitor: <i>val_loss</i> – Modo: <i>min</i>
Función de pérdida	<i>binary_crossentropy</i>
Número de épocas	50
Tamaño de lote	64
Datos de entrada	<i>training_sentences</i>
Datos de destino	<i>training_labels</i>
Datos de validación	<i>testing_sentences</i> , <i>testing_labels</i>
Longitud de datos de entrada - capa de entrada	5,000
Tamaño de vocabulario - capa Embedding	74,096
Dimensión de incrustación - capa Embedding	100
Ratio de capa de desactivación	0.50
Número de neuronas de salida - capa Bidireccional LSTM	128
Función de activación - capa final	<i>sigmoid</i>

I Parámetros para modelo de Aprendizaje Profundo Multimodal

Hiperparámetro	Valor
Pesos de clases	<ul style="list-style-type: none"> – Exitoso (1): 1.7581290322580645 – Fracasado (0): 0.6987077585764833
Optimizador	<ul style="list-style-type: none"> – Nombre: ADAM – Ratio de aprendizaje: $1 * 10^{-5}$ – Ratio de decaimiento: $1 * 10^{-5}$
Parada temprana	<ul style="list-style-type: none"> – Monitor: <i>val_loss</i> – Modo: <i>min</i> – Paciencia: 10
Reducción de ratio de aprendizaje	<ul style="list-style-type: none"> – Monitor: <i>val_accuracy</i> – Modo: <i>max</i> – Paciencia: 5 – Factor: 0.01
Punto de guardado del modelo	<ul style="list-style-type: none"> – Monitor: <i>val_loss</i> – Modo: <i>min</i>
Función de pérdida	<i>binary_crossentropy</i>
Número de épocas	200
Tamaño de lote	32 ^a
Datos de entrada	<i>X_train_stacked</i> ^b
Datos de destino	<i>y_train</i>
Datos de validación	<i>X_test_stacked</i> ^c , <i>y_test</i>
Longitud de datos de entrada ^d	3,671, 5,000, 6
Número de neuronas de entrada - capa concatenada ^e	1, 1, 1
Número de neuronas de salida - capa concatenada	3
Número de neuronas de salida - capa densa 1	10
Función de activación - capa densa 1	<i>relu</i>
Función de activación - capa final	<i>sigmoid</i>

^a Al no especificarse, por defecto se asignó el valor de 32 (Keras, s.f.); ^b comprende la concatenación de *X_train_description*, *X_train_comments* y *X_train_metadata*; ^c comprende la concatenación de *X_test_description*, *X_test_comments* y *X_test_metadata*; ^d representan las longitudes de datos de entrada de cada modalidad; ^e representan las salidas de cada modalidad