ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

CLASIFICACIÓN AUTOMÁTICA DE PARÁSITOS DE REPTILES MEDIANTE REDES NEURONALES CONVOLUCIONALES

TRABAJO DE INTEGRACIÓN CURRICULAR PRESENTADO COMO REQUISITO PARA LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN TELECOMUNICACIONES

BRYAN PATRICIO NÚÑEZ ALVERCA
bryan.nunez01@epn.edu.ec

DIRECTOR: Ph.D. FELIPE LEONEL GRIJALVA ARÉVALO
felipe.grijalva@epn.edu.ec

Quito, febrero 2022

CERTIFICACIONES

Yo, BRYAN PATRICIO NÚÑEZ ALVERCA declaro que el trabajo de integración curricular aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

BRYAN PATRICIO NÚÑEZ ALVERCA

Certifico que el presente trabajo de integración curricular fue desarrollado por BRYAN PATRICIO NÚÑEZ ALVERCA, bajo mi supervisión.

Ph.D. FELIPE LEONEL GRIJALVA ARÉVALO
DIRECTOR

DECLARACIÓN DE AUTORÍA

A través de la presente declaración, afirmamos que el trabajo de integración curricular aquí descrito, así como el (los) producto(s) resultante(s) del mismo, son públicos y estarán a disposición de la comunidad a través del repositorio institucional de la Escuela Politécnica Nacional; sin embargo, la titularidad de los derechos patrimoniales nos corresponde a los autores que hemos contribuido en el desarrollo del presente trabajo; observando para el efecto las disposiciones establecidas por el órgano competente en propiedad intelectual, la normativa interna y demás normas.

BRYAN PATRICIO NÚÑEZ ALVERCA

Ph.D. FELIPE LEONEL GRIJALVA ARÉVALO

DEDICATORIA

A mis padres quienes me han dado
todo lo que soy como persona, mis principios,
mis valores, mi perseverancia y mi empeño,
y todo ello con una gran dosis de amor y
sin pedir nunca nada a cambio.

A mis hermanos por su cariño y apoyo incondicional
durante todo este proceso, por estar conmigo
en todo momento.

A toda mi familia porque con sus oraciones,
consejos y palabras de aliento hicieron
de mi una mejor persona y de una u otra
forma me acompañan en todos mis sueños y metas.

Bryan Núñez

AGRADECIMIENTOS

Agradezco a Dios quien con su bendición llena siempre mi vida y a toda mi familia por estar siempre presentes.

A mis padres Patricio y Verónica por ser mi fuente de inspiración, por guiarme con amor y sabiduría, por su inmenso sacrificio y dedicación por lo que no existen palabras para agradecer todo lo imposible que han hecho por mi bienestar.

A mi hermano Diego quien me apoyo y acompaño desde el principio, quien presencio todo el esfuerzo que realice y supo comprenderme y aconsejarme.

A mi hermana Karla quien ha sido un ejemplo a seguir y eje principal ya que sin ella no habría podido desarrollar mi trabajo de titulación.

Al PhD. Felipe Grijalva quien gracias a sus conocimientos, sus orientaciones, su manera de trabajar, su paciencia y su motivación han sido fundamentales para mi formación durante toda mi carrera.

A mis amigos que he tenido la oportunidad de conocer ya que nos hemos apoyado incondicionalmente durante toda la carrera y con los que he compartido momentos inolvidables.

Finalmente, a la Escuela Politécnica Nacional y sus docentes por todas las oportunidades que me ha brindado para mi desarrollo personal y profesional con un excelente nivel académico.

ÍNDICE DE CONTENIDO

RESUMEN

Los centros de rescate y cuidado de vida silvestre dejan abierta la posibilidad de que sus animales se conviertan en foco de enfermedades e infecciones. Los avances tecnológicos de redes neuronales convolucionales han abierto un nuevo campo para la detección y clasificación de enfermedades que muestran un gran potencial para superar los problemas y errores de la detección manual. Razón por la cual se ha planteado el realizar una investigación que sirva de herramienta en la identificación de agentes parasitarios que afectan en este caso a reptiles en cautiverio, mediante la utilización de algoritmos de segmentación, aumento de datos y transfer learning. Este último se ha realizado mediante la red pre entrenada MobileNet entrenada con la base de datos obtenida mediante la recolección de imágenes de parásitos por la médica veterinaria Alejandra Núñez. Los resultados serán comparados con una red no pre entrenada. Se tiene como resultado que la red pre entrenada tiene mejor desempeño que la red no pre entrenada. Se obtiene un excelente desempeño en el aprendizaje de características mediante la gráfica t-SNE y en la clasificación mediante las curvas ROC, métrica AUC y matriz de confusión.

PALABRAS CLAVE: Aprendizaje Automático, Parasitos, Redes Neuronales Convolucionales, Transferencia de aprendizaje, ROC

ABSTRACT

Wildlife rescue and care centers leave open the possibility that their animals become diseases and infections focus . Technological advances in convolutional neural networks have opened a new field for detection and classification of diseases that show potential to overcome the problems and errors in manual detection. Reason for it has been proposed to carry out an investigation that serves as a tool in the parasitic agents identification that affect in this case reptiles in captivity, through the use of segmentation algorithms, data augmentation and transfer learning. The latter has been carried out using the pre-trained MobileNet network trained with the database obtained by collecting images of parasites by the veterinary doctor Alejandra Nuñez. The results will be compared with a non pre-trained network. The result is that the pre-trained network performs better than the non-pre-trained network. Excellent performance is obtained in learning features using the t-SNE plot and in classification stage using ROC curves, AUC metrics, and confusion matrix.

KEYWORDS: Machine Learning, Parasites, Convolutional Neural Networks, Transfer Learning, ROC curve

1. INTRODUCCIÓN

La investigación de la parasitología de reptiles no se ha explorado completamente en la literatura científica [1]. Los parásitos son uno de los agentes infecciosos más comunes y se propagan fácilmente dentro de los centros de cuidado y manejo de vida silvestre [2], que pueden causar lesiones o inmunosuprimir a los reptiles, aumentando la tasa de mortalidad o abriendo las puertas a enfermedades secundarias [3]. Se ha demostrado que algunos tipos de parásitos causan hepatitis en serpientes [4]. Otros afectan el comportamiento y la fisiología de las lagartijas, e incluso algunos parásitos pueden causar enteritis crónica con edema y mucosa intestinal hemorrágica [4]. Además, los ácaros en los reptiles causan debilidad debido a la pérdida de sangre, neumonía e incluso septicemia [5]. La necesidad de controlar el parasitismo en las especies cautivas es importante porque pueden ser una fuente de transmisión de enfermedades zoonóticas, es decir, de animal a humano [6]. Los métodos manuales de clasificación de parásitos implican el análisis de imágenes de microscopio de heces por expertos humanos. Sin embargo, esta tarea requiere de un esfuerzo y tiempo importante ya que, en un centro donde se refugian los reptiles, pueden presentar una carga parasitaria muy alta de diferentes tipos, que afectan a su salud. Las herramientas de clasificación automática para agentes parasitarios pueden ayudar a los investigadores o cuidadores a operar de manera mucho más rápida y eficiente. En consecuencia, es posible realizar diagnósticos más rápidos y elaborar protocolos de desparasitación adecuados para prevenir enfermedades en reptiles en cautiverio.

En este contexto, las redes neuronales convolucionales (RNCs) han demostrado ser una valiosa herramienta para encontrar y clasificar patrones en imágenes más fácilmente que los especialistas, que invierten largos tiempos en la identificación y son propensos a error. Las RNC se han estudiado durante varios años, obteniendo buenos resultados en tareas de clasificación de imágenes, y mediante el uso de modelos similares, es posible generalizar algoritmos para resolver diferentes tipos de problemas [7]. Por ejemplo, estas herramientas de aprendizaje automático se probaron recientemente en muchos campos, incluidos la medicina y la biología [8]. Hay varios factores [9] que influyen en la eficacia de las RNC (Ej., Arquitectura, procesamiento de datos, segmentación, etc.). Además, la complejidad que implica entrenar una RNC desde cero, en general, no es factible cuando los datos son insuficientes. Por lo tanto, esto ha llevado al uso de métodos de apoyo como el aumento de datos y el transferencia de aprendizaje [10]. Este último usa una RNC previamente entrenada con un conjunto de datos y una tarea base, y luego las características aprendidas

(pesos de red) se transfieren a una segunda red para ser entrenadas en un nuevo conjunto de datos y una tarea objetivo [11].

Por lo cual, este trabajo propone un nuevo enfoque para identificar agentes parasitarios a partir de imágenes microscópicas de heces que afectan a los reptiles en cautiverio, mediante algoritmos de segmentación, estrategias de aumento de datos y RNC bajo un esquema de transferencia de aprendizaje.

En [12] se propone un modelo para la detección de mosca blanca adulta (*Bemisia tabaci*) y trips (*Frankliniella occidentalis*) en invernaderos. Un sistema de adquisición de imágenes mediante trampas adhesivas permitió la recopilación de la base de datos. La segmentación se realizó utilizando el algoritmo OTSU y otros métodos de procesamiento de imágenes digitales. Finalmente, la clasificación se llevó a cabo con la ayuda de una red neuronal feed-forward. Un trabajo similar es abordado por [13], donde se extrajeron y analizaron varios rasgos morfológicos relacionados con el tamaño y color de los especímenes para clasificar-los.

En [14] los autores presentan un método para la detección y clasificación binaria de célu-las infectadas por el parásito de la malaria. Proponen una etapa de segmentación basada en operadores morfológicos de sombrero de copa [15], y la etapa de clasificación utiliza diferentes conjuntos de características de textura y forma que alimentan una red neuronal.

Además, en [16], se porpone un sistema de monitoreo remoto de trampas de insectos en tiempo real que emplea IoT y un método para clasificar insectos basado en Faster RNC (R-CNN) y ResNet 50 aplicando la transferencia de aprendizaje. Los resultados muestran que el sistema podría identificar insectos automáticamente con una precisión del 94 %.

Finalmente, en [17] se propone un enfoque para clasificar los organismos protozoarios y metazoarios. Específicamente, comparan el análisis discriminante, redes neuronales y árboles de decisión. Descubrieron que el análisis discriminante y el rendimiento de la red neuronal eran bastante similares, mientras que la técnica del árbol de decisión era menos eficiente.

Como en trabajos anteriores, utilizaremos técnicas tradicionales de procesamiento digitales de imágenes como binarización y segmentación [12,14,17]. Luego, usaremos un clasificador basado en RNC bajo un esquema de transferencia de aprendizaje como en [16].

Cabe mencionar que se ha realizado una versión extendida de este trabajo, la cual fue entregada a la revista científica Plos One. Esta versión esta bajo revisión hasta la fecha de realización de este trabajo.

1.1. Objetivo general

Clasificar automáticamente parásitos de reptiles usando redes neuronales convolucionales.

1.2. Objetivos específicos

- Estudiar la teoría de las redes neuronales convolucionales y el aprendizaje automático.
- Estudiar los aspectos más importantes de la parasitología en reptiles y su clasificación.
- Segmentar las imágenes para aislar la región de interés donde se encuentra el parásito.
- Implementar una red neuronal convolucional como clasificador y extractor de características de parásitos.
- Determinar el rendimiento del clasificador mediante métricas de aprendizaje automático

1.3. Alcance

Para este proyecto primero se obtendrá la base de datos provista por la Médica Veterinaria Alejandra Núñez, la cual en su proyecto de investigación “Identificación de parásitos con diferentes métodos coprológicos en muestras de reptiles en el vivarium de Quito”, extrajo las imágenes de los parásitos mediante microscopio y los identificó de acuerdo a su morfología [18]. La base de datos cuenta con 4116 imágenes distribuidas en 42 tipos diferentes de parásitos. Para el propósito de este trabajo de titulación se utilizarán las imágenes de 6 parásitos (Figura 1.1) que poseen la mayor cantidad de imágenes como se muestra en la Tabla 1.1.

Tabla 1.1: Distribución de parásitos

Parásitos (etiquetas)	**Imágenes Totales**
Ácaro - *ophionyssus natricis* (ophi)	245
Blastocystis sp (blas)	950
Oxiurdo egg (oxi)	345
Rhytidoides similis (rhyti)	1072
Strongyloides (strong)	648
Taenia (tae)	356
Total	3616

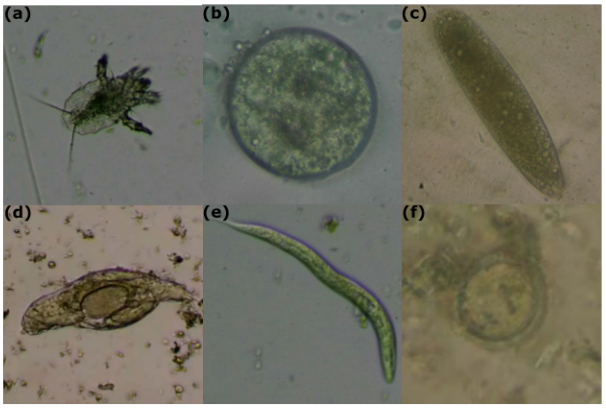


Figura 1.1: Ejemplos de cada parásito. (a) Ácaro (*Ophionyssus natricis*); (b) *Blastocystis sp*; (c) *Oxiurdo egg*; (d) *Rhytidoides similis*; (e) *Strongyloides*; (f) *Taenia*.

La Figura 1.2 muestra un diagrama de bloques de implementación del sistema de clasificación automática de parásitos de reptiles basado en redes neuronales convolucionales. Este modelo requiere de dos fases: Entrenamiento y prueba.

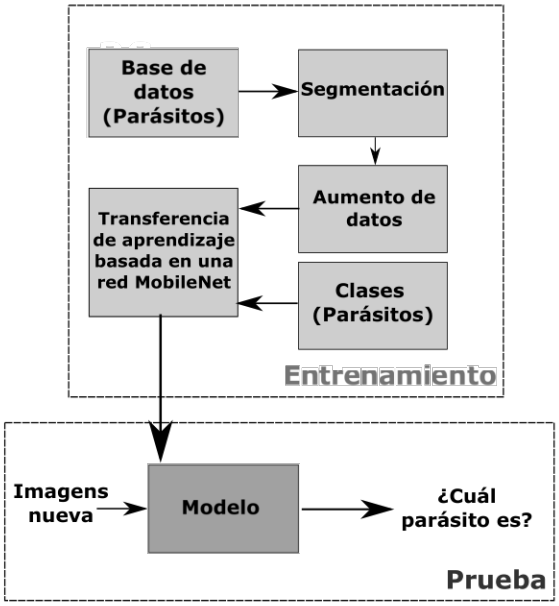


Figura 1.2: Diagrama de bloques de implementación del sistema de clasificación automática de parásitos de reptiles basado en redes neuronales convolucionales.

En la fase de entrenamiento la base de datos pasará por un proceso segmentación mediante procesamiento digital de imágenes donde se eliminará el ruido y se recortará la imagen para que sea visible únicamente el parásito, luego las imágenes pasarán por un proceso de aumento de datos para hacer nuestro sistema más robusto [19], luego esta nueva base de datos aumentada pasará por una red neuronal convolucional pre-entrenada, en la cual se utilizará transferencia de aprendizaje para ajustar los pesos de la misma.

También, este modelo será validado gracias a la técnica de K Fold Cross Validation [20].

Con esto, calcularemos las métricas como la eficacia de entrenamiento, de validación y de prueba. Luego se realizará un gráfico ROC , el cual nos permitirá calcular el Área bajo la curva (AUC) y matriz de confusión. Como último resultado, se realizará la gráfica t-SNE que permite observar el aprendizaje de características.

Por otro lado, se utilizará un modelo de red no pre-entrenada, para poder comparar los resultados de MobileNet con dicha red y ofrecer un análisis de cuán precisas son las redes neuronales pre-entrenadas. Este proyecto de titulación no tiene producto final demostrable.

1.4. Marco teórico

1.4.1. Segmentación de imagenes

1.4.1.1. CLAHE

EL método de la ecualización de histograma adaptativo limitado por contraste CLAHE (Contrast Limited Adaptive Histogram Equalization) se formula en función de dividir la imagen en varias regiones que no se superponen de tamaños casi iguales. Para imágenes de 512 × 512, para lograr una buena estimación estadística, el número de regiones se selecciona generalmente para que sea igual a 64 dividiendo por igual la imagen por 8 en cada dirección. Esta partición da como resultado tres grupos diferentes de regiones. Un grupo, que consta solo de las regiones de las esquinas que en total son 4. El segundo grupo, que consta de 24 regiones, es la clase de regiones de borde que son las regiones de las aristas de la imagen. Las demás regiones que no son de borde o esquinas pertenecen a la clase regiones internas y en este caso constan de 36 regiones. En este enfoque, primero, se calcula el histograma de cada región. Luego, basándose en un límite deseado para la expansión del contraste, se obtiene un límite de recorte para histogramas de recorte. A continuación, cada histograma se redistribuye sin sobrepasar el límite de recorte. En la técnica CLAHE, los píxeles se mapean combinando linealmente los resultados de las asignaciones de las cuatro regiones más cercanas [21]. Un ejemplo de esto se puede visualizar en la Figura 1.3

1.4.1.2. Escala de grises

Una operación muy próxima a la binarización de la imagen que se debe usar es la conversión RGB a gris, una imagen coloreada se convierte en una imagen gris [22]. Es necesario realizar la siguiente operación 1.1 en todos los píxeles de una imagen para realizar dicha conversión.

$$GRIS = 0{,}2989 \times R + 0{,}5870 \times G + 0{,}1140 \times B \tag{1.1}$$

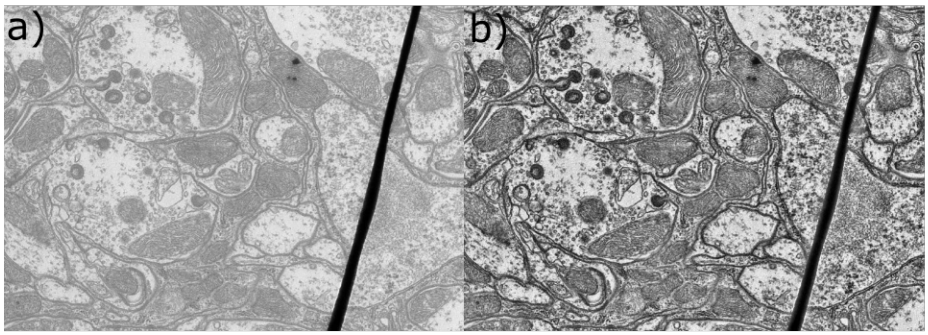


Figura 1.3: Ejemplo del método CLAHE en una imagen TEM. a) Imagen original. b) Imagen procesada mediante CLAHE.

1.4.1.3. Binarizacion mediante OTSU

La creación de umbrales es una herramienta muy utilizada en la segmentación de imágenes cuando uno está interesado en identificar los diferentes componentes homogéneos de la imagen. En un caso ideal, el histograma tiene un valle profundo y agudo entre dos picos que representan los objetos y el fondo, respectivamente, de modo que el umbral se puede elegir en el fondo de este valle. El método Otsu es un método no paramétrico y no supervisado de selección automática de umbrales para la segregación de imágenes. Un umbral óptimo se selecciona mediante el criterio discriminante, a saber, para maximizar la separabilidad de las clases resultantes en niveles de gris [23]. Un ejemplo de esto se puede visualizar en la Figura 1.4.

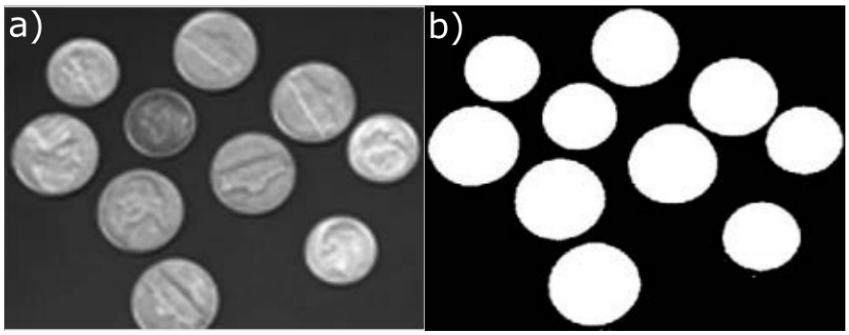


Figura 1.4: Ejemplo del método OTSU en una imagen de monedas. a) Imagen original. b) Imagen procesada mediante OTSU.

1.4.1.4. Operaciones Morfológicas

Las operaciones morfológicas son procedimientos simples que se aplican a imágenes que están en un formato binario. Es necesario dos elementos, la primera es la imagen ya binarizada, y la segunda es el elemento estructurante o núcleo, esta ultima decide la naturaleza de la transformación [24]. Dos operadores morfológicos básicos son Erosión y Dilatación.

- Erosión: esta transformación realiza el desplazamiento del núcleo a través de la imagen. Dependiendo el tamaño del núcleo, la erosión irá descartando todos los píxeles cerca de los bordes de los objetos que se encuentren en la imagen binarizada. En efecto, el volumen de los objetos se reduce. Esta operación es beneficiosa si se desea eliminar ruido u objetos pequeños. También permite separar los objetos que se encuentren conectados de la imagen [24].Un ejemplo se muestra en la Figura 1.5

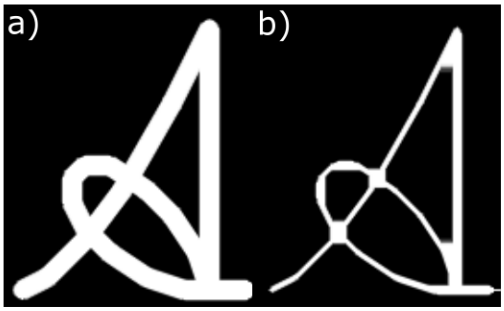


Figura 1.5: Ejemplo de la erosión en una imagen. a) Imagen original. b) Imagen erosionada

- Dilatación: esta transformación realiza la operación contraria de la erosión. Dependiendo el tamaño del núcleo, la dilatación aumentando el volumen de los objetos. Esta operación se la realiza para restablecer el tamaño de la imagen luego de la erosión [24]. Un ejemplo se muestra en la Figura 1.6

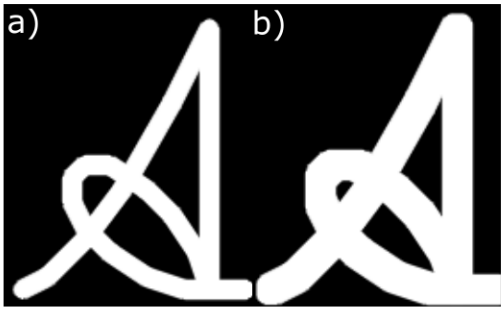


Figura 1.6: Ejemplo de la dilatación en una imagen. a) Imagen original. b) Imagen dilatada

- Gradiente Morfológico: esta operación realiza la diferencia entre las tareas de Erosión y Dilatación, teniendo como resultado los contornos de los objetos mas significativos de la imagen [24]. Un ejemplo se muestra en la Figura 1.7

1.4.2. Aprendizaje Automático

El Aprendizaje Automático se deriva de la Inteligencia Artificial permitiendo que computadoras puedan captar o aprender características directamente de experiencias o bases de datos. El aprendizaje automático puede crear un modelo matemático con el fin de hacer pre-

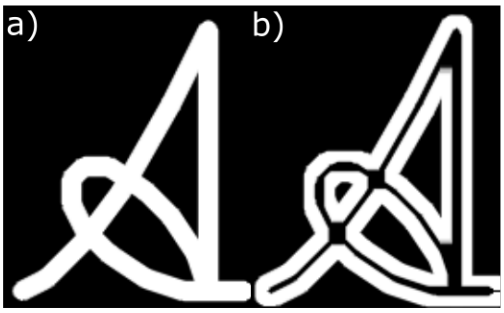


Figura 1.7: Ejemplo de una imagen procesada con el gradiente morfológico. a) Imagen original. b) Imagen procesada con el gradiente morfológico

dicciones o decisiones sin estar programado explícitamente para realizar la tarea [25].
Según la naturaleza de los datos de entrenamiento, es posible clasificar el aprendizaje automático de la siguiente manera:

- **Aprendizaje Supervisado:** Cuando se habla de algoritmos de aprendizaje supervisado se refiere a programas que reciben un conjunto de datos ya etiquetados llamado conjunto de entrenamiento e inducen modelos matemáticos que pueden ser usados para clasificar otros datos sin etiquetar [26].

- **Aprendizaje no supervisado:** A diferencia de los algoritmos de aprendizaje no supervisado, los algoritmos de aprendizaje no supervisado reciben conjuntos de entrenamiento con datos sin etiquetar generando modelos matemáticos que realizan de acuerdo a los criterios encontrados en el entrenamiento. Esto puede generar una cierta dificultad al momento de evaluar el desempeño del algoritmo. Entre los tipos de aprendizaje no supervisado destacan la reducción de la dimensionalidad y la agrupación (clustering) [25].

1.4.2.1. Evaluación de modelo

- **Precisión:** La precisión es una de las métricas más utilizadas para estimar el desempeño de los sistemas de aprendizaje en problemas de clasificación.Se utiliza para describir la cercanía de una medición al valor verdadero [27].

- **Curva ROC/AUC:** La curva ROC se obtiene trazando la sensibilidad o la tasa de verdaderos positivos (TPR) en el eje Y contra la especificidad o la tasa de falsos positivos (FPR) en el eje X para diferentes valores de decisión de umbral que varían de 0 a 1 (ver Figura 1.8) [28].

- **Matriz de confusión:** Una matriz de confusión es un resumen de las predicciones

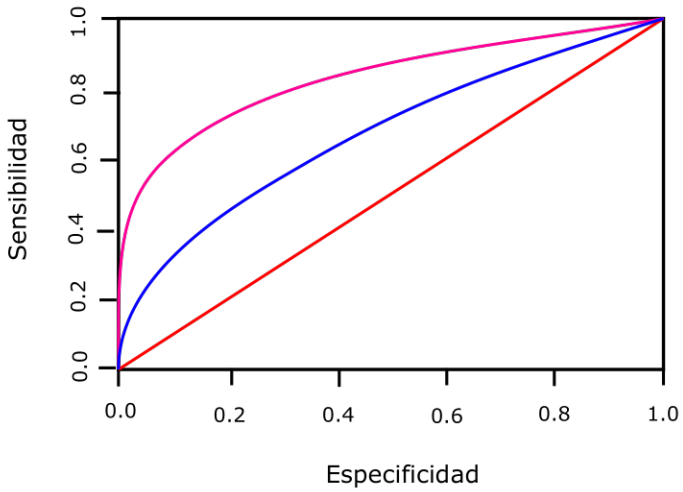


Figura 1.8: Ejemplo de una curva ROC/AUC

correctas e incorrectas en forma de recuento sobre un problema de clasificación. En otras palabras se puede decir que la matriz de confusión muestra el grado de confusión del clasificador cuando hace predicciones. No indica solamente los errores que está cometiendo su clasificador, sino, sobre los tipos de errores que está cometiendo (ver Figura 1.9) [29].

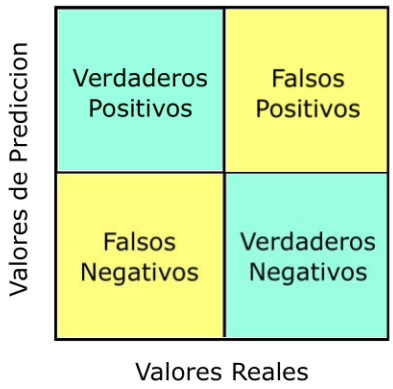


Figura 1.9: Ejemplo de una matriz de confusión.

- **Validación cruzada de K-fold:** La validación cruzada es un procedimiento de reselección que se utiliza para evaluar modelos de aprendizaje automático en una muestra de datos limitada [30].

  Este proceso tiene un parámetro llamado k, que se refiere al número de grupos en los que se dividirá la muestra de datos total. Por ejemplo, k = 5 se convierte en 5 grupos (ver Figura 1.10).

  En general esta técnica realiza k veces la etapa de entrenamiento con un conjunto de validación diferente en cada una, generando k numero de modelos.

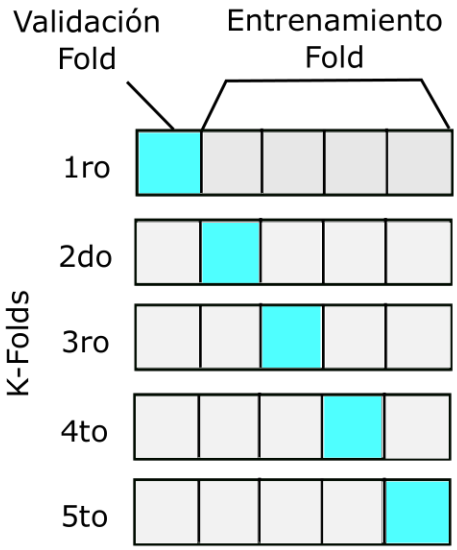


Figura 1.10: Representación de validación cruzada de K-fold

Este método es utilizado porque porque es fácil de entender y generalmente da como resultado evaluaciones mas realistas de las capacidades del modelo entrenado.

- **T-SNE:** Esta tecnica ha sido una herramienta muy utilizada en el campo del aprendizaje automático, ya que tiene una gran capacidad para crear mapas bidimensionales convincentes a partir de datos con cientos o incluso miles de dimensiones. El objetivo es tomar un conjunto de puntos en un espacio de alta dimensión y encontrar una representación fiel de esos puntos en un espacio de menor dimensión, típicamente el plano 2D. El algoritmo no es lineal y se adapta a los datos subyacentes, realizando diferentes transformaciones en diferentes regiones (ver Figura 1.11) [31].

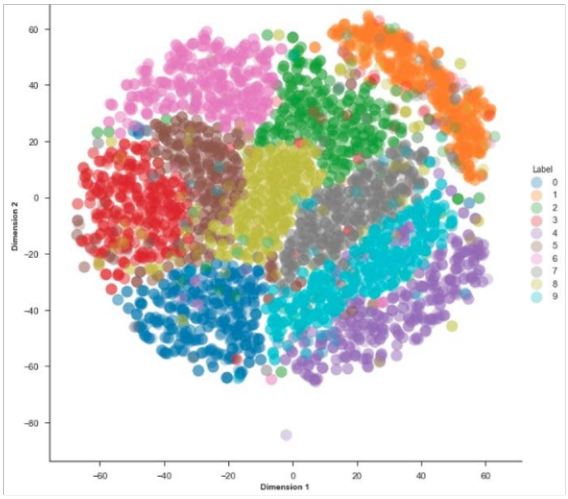


Figura 1.11: Representación de un mapa bidimensional T-SNE.

1.4.3. Redes neuronales convolucionales

Una red neuronal convolucional (RNC) es uno de los tipos de redes neuronales artificiales más utilizados. Conceptualmente, una RNC se parece a un perceptrón multicapa (MLP). Cada neurona del MLP tiene una función de activación que asigna las entradas ponderadas a la salida. Asimismo, la RNC se considera un MLP con una estructura especial. Esta estructura especial de la RNC le permite descubrir y extraer características profundas de los datos sin procesar, mediante operaciones de convolución y agrupación [32]. Las capas mas utilizadas son:

- **Capa convolucional:** La capa convolucional realiza una operación lineal que implica la multiplicación de un conjunto de pesos con la entrada. Debido que la técnica fue diseñada para objetos bidimensionales, la multiplicación se realiza entre una matriz bidimensional de datos de entrada y una matriz bidimensional llamada núcleo (kernel) [33].

  El núcleo suele ser de dimensiones mas pequeñas que la entrada para poder realizar la operación lineal la cual es el producto escalar. Esto permite que el mismo núcleo realice el producto escalar por la matriz de entrada varias veces en diferentes secciones de la entrada. Específicamente, el núcleo se aplica sistemáticamente a cada parte superpuesta del tamaño de un filtro de los datos de entrada, de izquierda a derecha y de arriba a abajo [34]. En la Figura 1.12 se muestra un ejemplo de una convolución.

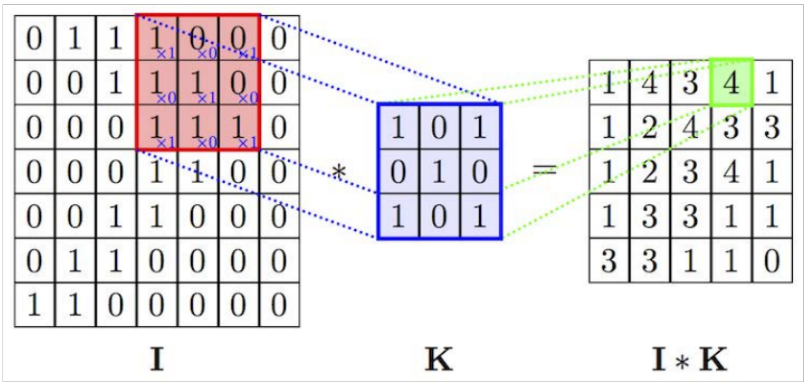


Figura 1.12: Ejemplo de operación de convolución con un tamaño de kernel de 3 × 3

  Juntar varias capas convolucionales permite que las capas más alejadas a la entrada aprenden características de alto orden como objetos específicos o formas más abstractas. En cambio las capas mas cercanas a la entrada aprendan características de bajo nivel como texturas, bordes, líneas y esquinas [33].

- **Capa de back normalization (BN):** Una capa BN actúa como un regularizador para evitar el ajuste excesivo y permite que en la etapa de entrenamiento se pueda utilizar tasas de aprendizaje mayores, lo que es la causa de una convergencia más rápida y una mejor generalización [35].

- **Capa Leaky ReLU:** Leaky ReLU Layer es la función de activación que, a diferencia de otras funciones de activación, como sigmoide, RelU Layer tiene una velocidad de cálculo y tasa de convergencia más rápidas, gracias a su linealidad. También esta capa se introdujo para evitar el problema del gradiente de desaparición, ya que esta capa no causa saturación para entradas negativas y positivas [36]. En todas las pruebas se usó una pendiente de 0.3 para la función LeakyReLU.

- **Capa max pooling:** esta capa permite reducir la dimensionalidad de los mapas de características resumiendo la presencia más activada de una característica [33].

- **Capa Dropout:** es un algoritmo que es utilizado en la etapa de entrenamiento que se basa en abandonar estocásticamente las neuronas para evitar la coadaptación de los detectores de características o sobreajuste [37].

1.4.3.1. Transferencia de aprendizaje con MobileNet

La transferencia de aprendizaje se refiere a la capacidad de reutilizar un modelo de red neuronal ya entrenado (denominado modelo previamente entrenado) para determinada tarea como punto de partida para un modelo en una segunda tarea. En el dominio de la visión por computadora, los modelos previamente entrenados se entrenan en conjuntos de datos desafiantes con enormes recursos informáticos.
Por ejemplo, MobileNet [38] es una red neuronal que se entrenó con la base de datos ImageNet que tiene alrededor de 14 millones de imágenes. La principal característica de esta RNC es la capacidad de generar modelos con rendimiento comparable a otras redes neuronales más robustas como ResNet [39], VGG16 [40], pero con la ventaja de que consume pocos recursos computacionales gracias a sus bloques de bloques de convolución separables en profundidad (DWSCB). Esto hace que este modelo sea perfecto para usar en dispositivos de recursos limitados como teléfonos móviles [38], de ahí su nombre.
Un DWSCB comprende un bloque de convolución en profundidad (DWC) y un bloque de convolución puntual (PWC) con su respectiva capa de normalización por lotes y función de activación ReLU (unidad lineal rectificada). En resumen, un DWC aplica un filtro a cada canal de entrada, como se muestra en la Figura 1.13, donde M es el número de entradas y

D_k es el tamaño del núcleo. Luego, alimenta estas salidas a una capa de normalización por lotes y finalmente a una función de activación de ReLU. Posteriormente, se realiza un 1×1 PWC para combinar las salidas (ver Figura 1.14), y nuevamente pasa a través de una capa de normalización por lotes y una función de activación de ReLU. Finalmente, a partir de las Figuras 1.13, 1.14 y 1.15, es posible observar cómo una operación de convolución estándar se factoriza en una convolución en profundidad y un 1×1 operaciones de convolución puntuales.

Posteriormente, las salidas pasan a través de un bloque de convolución estándar (SC), que filtra y combina los parámetros de entrada en un nuevo conjunto de parámetros de salida. Este bloque nuevamente hace uso de una capa de normalización por lotes y una capa con función de activación ReLU.

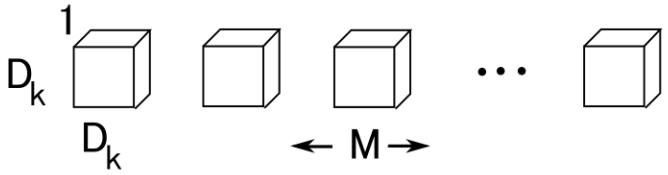


Figura 1.13: Capa Convolucional Depthwise. Adaptado de [38].

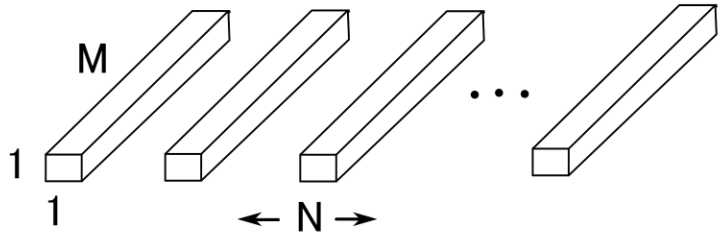


Figura 1.14: Capa Convolucional Pointwise. Adaptado de [38].

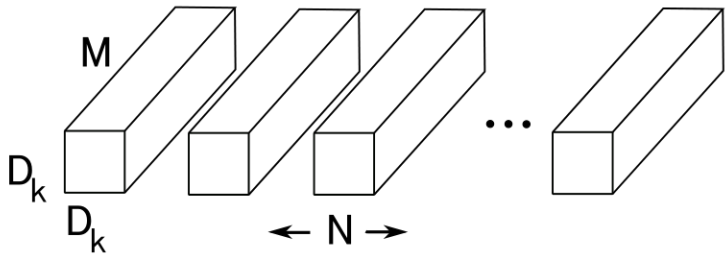


Figura 1.15: Proceso de Convolución Estándar. Adaptado de [38].

1.4.4. Herramientas de software

1.4.4.1. Jupyter Notebook

Jupyter Notebook es una herramienta que mediante su interfaz gráfica permite añadir código en diferentes lenguajes de programación, así también permite incluir imágenes, vídeo, texto y audio. Esta ejecución se realiza mediante la comunicación con un núcleo o kernel de calculo. Principalmente los desarrolladores incluyeron el kernel de cálculo de python, pero gracias a su carácter abierto se ha podido incluir varios kernels de calculo como Java, R, C, C++, etc. Esta interfaz, ha logrado que el lenguaje de programación no sea una limitante para mostrar el contenido de un proyecto sin la adopción de un único lenguaje [41].

1.4.4.2. Python

Python es un lenguaje de programación de código abierto, lo que hace que este siendo desarrollado continuamente con nuevas librerías y actualizaciones. Es un lenguaje que no necesita que sea compilado, sino que va interpretando mientras se ejecuta las lineas de código. Es uno de los códigos mas usados debido a su portabilidad y compatibilidad con múltiples plataformas. Entre sus usos mas característicos están: Netwoking, Ciencia de Datos, Desarrollo web o móvil, aprendizaje automático, entre otros [42].

1.4.4.3. Herramientas de aprendizaje automático mediante Python

- **TensorFlow:** TensorFlow es un sistema de codigo abierto utilizado principalmente para el uso de aprendizaje automático que opera en entornos heterogéneos y a gran escala. Permite asignar a diferentes máquinas como procesadores y tarjetas gráficas el procesamiento computacional de manera paralela. TensorFlow admite una variedad de aplicaciones, pero se enfoca particularmente en el entrenamiento y la predicción con redes neuronales profundas. Sirve como plataforma para la investigación y para implementar sistemas de aprendizaje automático en muchas áreas, como la la visión artificial, reconocimiento de voz, robótica y la recuperación de información. Mediante sus funciones permite optimizar la etapa de entrenamiento y luego aplicar esas funciones durante la etapa de prueba [43].

- **Keras:** Keras es una librería de Python de alto nivel, fácil de comprender y compacta. Es utilizado mayormente para el aprendizaje profundo que se puede ejecutar sobre TensorFlow. Permite una interacción directa con los conceptos principales del aprendizaje profundo, como la creación de capas para redes neuronales, mientras se ocupa

en segundo plano de los detalles esenciales de los tensores, sus formas y sus detalles matemáticos [44].

- **Scikit-Learn:** Scikit-Learn es una librería de código abierto de Python destinada a la integración fácil y rápida de métodos de aprendizaje automático. Posee un amplio catalogo de métodos para clasificación, regresión, reducción de dimensionalidad estimación de matriz de covarianza, preprocesamiento de datos, entre otros [45].

- **SciPy:** SciPy es una biblioteca de rutinas numéricas para el lenguaje de programación Python que proporciona bloques de construcción fundamentales para modelar y resolver problemas científicos. SciPy incluye algoritmos para integración, problemas de valores propios, optimización, ecuaciones diferenciales y algebraicas, interpolación y muchas otras. También proporciona estructuras de datos especializadas, como matrices dispersas y árboles dimensionales k. SciPy se basa en Numpy, que es uno de los paquetes fundamentales de Python ya que proporciona estructuras de datos de matriz y rutinas algebraicas rápidas [46]. Así mismo, SciPy es la base sobre la que se basan las bibliotecas científicas de nivel superior, incluidas scikit-learn [47].

2. METODOLOGÍA

Figura 2.1 muestra el diagrama de bloques de nuestro enfoque propuesto. La primera etapa consiste en recolectar imágenes de heces de reptiles. Un veterinario especialista etiquetó las imágenes según el parásito presente en ellas. Estas imágenes deben pasar por un proceso de segmentación para reducir el ruido de fondo y poder apreciar el parásito con la mayor claridad posible. Además, deben pasar por un proceso de aumento de datos para hacer frente a las clases desequilibradas. La arquitectura MobileNet previamente entrenada se utiliza para entrenar un nuevo modelo en nuestras imágenes en la etapa de transferencia de aprendizaje. Finalmente, el modelo entrenado predice el tipo de parásito para imágenes de heces de reptiles nunca antes vistas.

Todo el procedimiento con el detalle de la ejecución de los programas y base de datos se encuentra en el Anexo A.

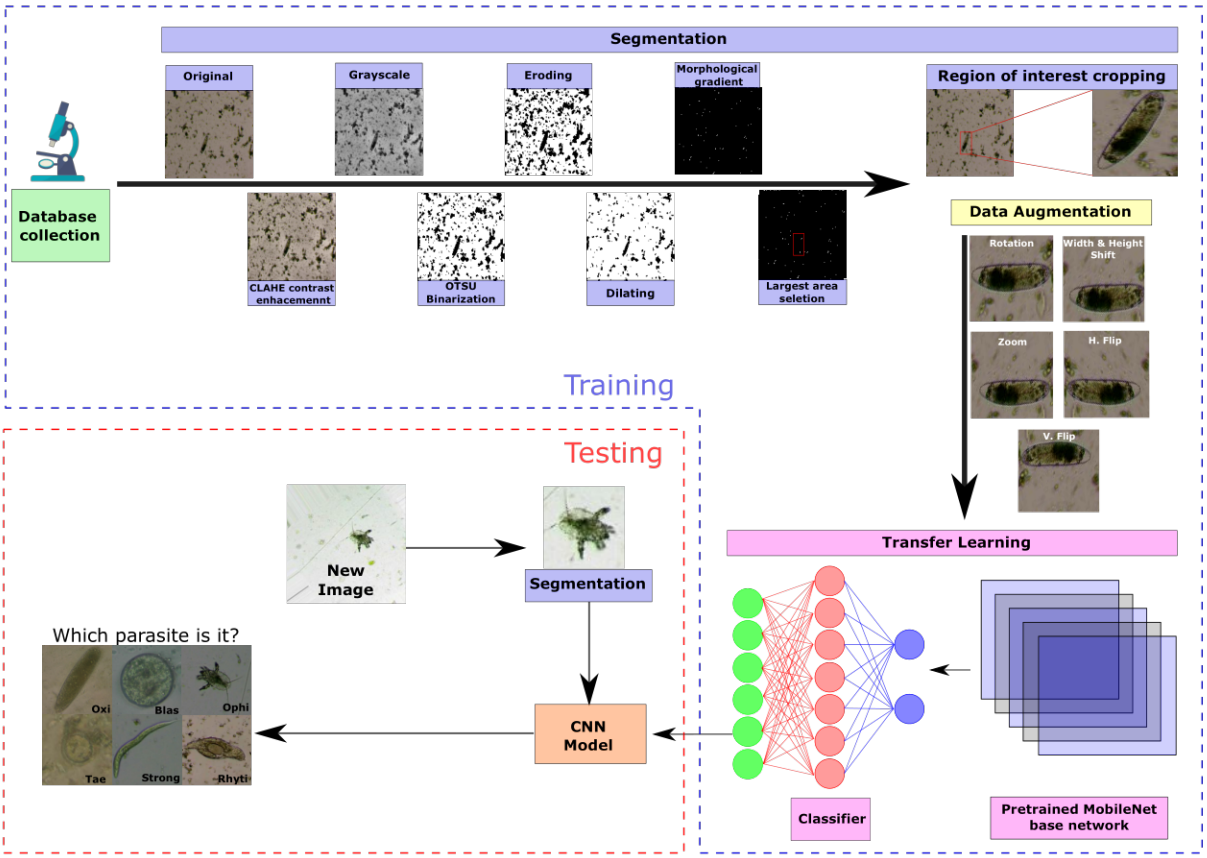


Figura 2.1: Diagrama de bloques del enfoque propuesto.

2.1. Conjunto de datos

La recogida de muestras de heces se realizó por el método indirecto, es decir, a partir de las muestras de heces depositadas en las zonas donde se ubica cada animal. Las muestras se recolectaron con un depresor de lengua para cada muestra para evitar la contaminación, y luego se almacenaron dentro de bolsas Ziploc selladas al vacío. Estas bolsas se etiquetaron y se colocaron en un recipiente hermético más fresco. En total, se recolectaron 118 muestras de heces.

Las muestras de heces se procesaron mediante tres exámenes de heces. Las pruebas de heces son los métodos más utilizados para diagnosticar infecciones parasitarias. El primero fue el método directo que investiga muestras de heces frescas en busca de formas parasitarias en su mayoría móviles bajo el microscopio. La segunda es la técnica de flotación, que hace que las formas parásitas floten hacia la superficie debido a su menor densidad que la solución en la que están sumergidas. La tercera es por sedimentación, que utiliza la gravedad para que cada parásito llegue a sedimentar de forma natural en un medio de menor densidad.

Dependiendo del método es posible identificar mejor unos parásitos que otros a partir de la misma muestra, por lo que cada muestra se analizó a través de un microscopio mediante las tres pruebas descritas anteriormente. Se obtuvieron imágenes digitales de parásitos utilizando un microscopio táctil digital (Better Scientific Led Q190A-LCD, aumento de X10, X40 y X100) para obtener varias imágenes y videos de los parásitos. Finalmente, cada imagen y video fueron anotados por un veterinario especialista usando las etiquetas que se muestran en la Tabla 2.1 de acuerdo con el parásito presente en ellos.

Utilizando el procedimiento descrito, se construyó un conjunto de datos de 3616 imágenes y 26 videos que contenían 4849 fotogramas de seis parásitos (ver Figura 2.2 para ver ejemplos de cada parásito) de acuerdo con la distribución que se muestra en la Tabla 2.1. Para este estudio, extrajimos todos los fotogramas de los 26 videos, que se utilizaron solo durante la etapa de entrenamiento de la red neuronal.

2.2. Método propuesto

2.2.1. Segmentación

La segmentación es un procedimiento digital de imágenes que extrae la región de interés de la imagen original [48]. La etapa de segmentación fue crucial para lograr un buen ren-

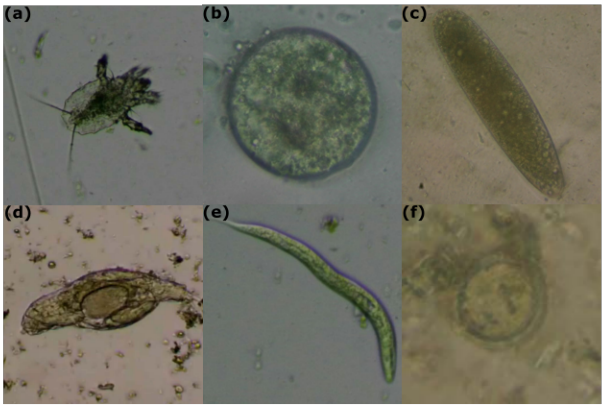


Figura 2.2: Ejemplos de cada parásito. (a) Acaro (*Ophionyssus natricis*); (b) *Blastocystis sp*; (c) *Oxiurdo egg*; (d) *Rhytidoides similis*; (e) *Strongyloides*; (f) *Taenia*.

Tabla 2.1: Distribución de la base de datos de parásitos

Parásitos (etiquetas)	Imágenes Totales	Videos
Ácaro - *ophionyssus natricis* (ophi)	245	2 (1236 frames)
Blastocystis sp (blas)	950	0
Oxiurdo egg (oxi)	345	6 (888 frames)
Rhytidoides similis (rhyti)	1072	6 (1048 frames)
Strongyloides (strong)	648	11 (764 frames)
Taenia (tae)	356	1 (913 frames)
Total	3616	4849

dimiento en la etapa de entrenamiento de la red, ya que la mayoría de las imágenes en la base de datos son muy ruidosas (por ejemplo, como se muestra en la Figura 2.3 a) ya que las imágenes se tomaron de heces de animales. Con este objetivo, las imágenes fueron procesadas mediante los siguientes pasos como se muestra en la Figura 2.3.

- CLAHE (Ecualización de histograma adaptable de contraste limitado) es una técnica de procesamiento de imágenes digitales que mejora el contraste de la imagen sin aumentar el ruido. Selecciona diferentes secciones de la imagen para redistribuir sus valores de brillo de píxeles. Como resultado, el contraste de la imagen se mejora mientras se preservan los contornos de los objetos [21], como se muestra en la Figura 2.3 b.

- Luego los píxeles con valores de color RVA (rojo, verde, azul) fueron convertidos a escala de grises (ver Figura 2.3 c)

- Después de convertir la imagen de color RVA (rojo, verde, azul) a escala de grises, se aplica la binarización de la imagen a través del método de Otsu [23], que determina el umbral de conversión más apropiado minimizando la variación intraclase entre dos

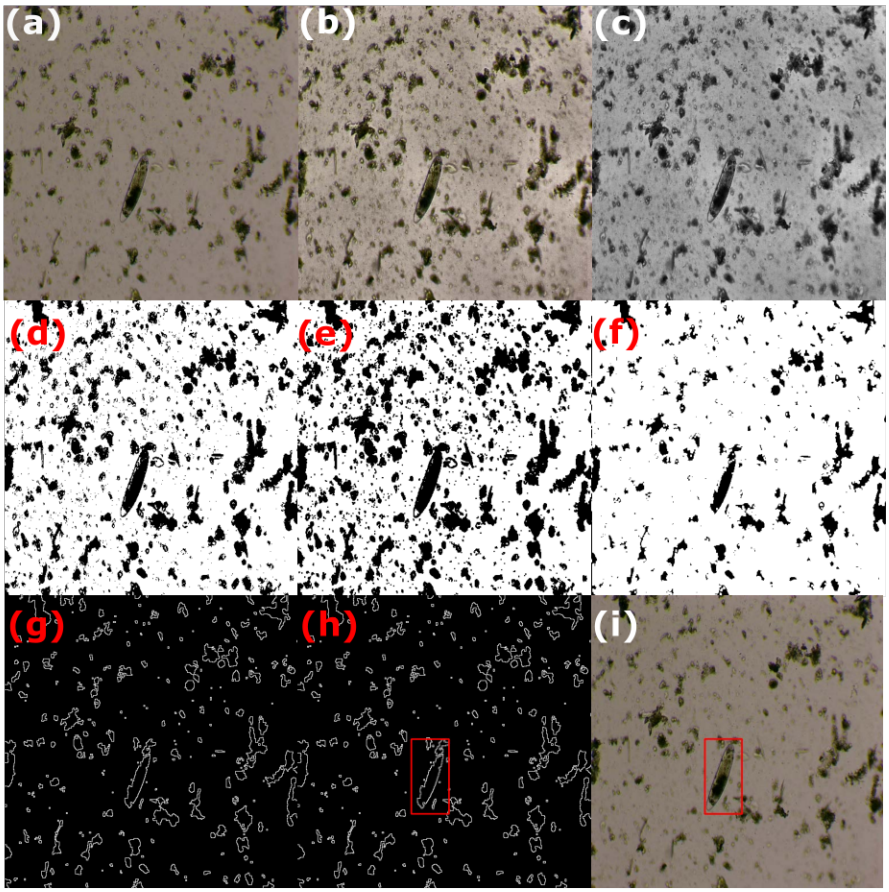


Figura 2.3: Detección de parásitos mediante técnicas de procesamiento de imágenes: (a) imagen original; (b) mejora del contraste de la imagen usando CLAHE; (c) conversión de imágenes de color a escala de grises; (d) binarización de imágenes utilizando el método OTSU; (e) operación morfológica basada en erosión; (f) operación morfológica basada en dilatación; (g) operación de gradiente morfológico; (h) selección del área más grande (región de interés) en la imagen procesada, (i) trazar la región de interés en la imagen original.

clases de píxeles asumidas (generalmente, blanco y negro), como se muestra en la Figura 2.3 d.

- Las operaciones morfológicas [24] (Figura 2.3 e y Figura 2.3 f) se realizan erosionando y luego dilatando los píxeles de la imagen para disminuir el ruido en la imagen a través de el kernel. Para nuestro análisis se utilizo una matriz de 3x3 dimensiones compuesta de unos como kernel (B).

$$B = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Observe que la erosión tiende a eliminar los objetos pequeños debido a los escombros o la basura, de modo que solo quedan los objetos sustantivos, mientras que la dilatación hace que los objetos como el parásito sean más visibles.

El gradiente morfológico (Figura 2.3 g) de una imagen I se obtiene calculando la diferencia entre las operaciones de dilatación ($\oplus$) y erosión ($\ominus$) del paso anterior usando el kernel B, de acuerdo con la ecuación 2.1.

$$\mathsf{G} = (\mathsf{I} \oplus \mathsf{B}) - (\mathsf{I} \ominus \mathsf{B}) \forall \mathsf{I} \in \mathbb{R}^2 \tag{2.1}$$

En la imagen resultante G, se enfatizan los contornos de los objetos más significativos [49].

- Finalmente, las áreas de posibles objetos se calculan después de la aplicación de la operación de gradiente morfológico. Es muy probable que las áreas más importantes contengan el objeto objetivo (parásito). Por lo tanto, estas áreas fueron recortadas y utilizadas para alimentar el clasificador de redes neuronales, como se muestra en la Figura 2.3 h y Figura 2.3 i.

2.2.2. Aumento de datos

El objetivo de las técnicas de aumento de datos es generar nuevas muestras de imágenes transformando la original. Por lo general, estas transformaciones son afines, es decir, transformaciones proyectivas que no mueven los objetos de las imágenes [50]. Por tanto, conservan las características de colinealidad de los objetos en el espacio analizado. En la actualidad, el aumento de datos es una solución práctica para entrenar modelos de redes profundas que exigen una gran cantidad de muestras de imágenes, evitando el sobreajuste del modelo. Usamos las siguientes estrategias de aumento:

- Rango de rotación: es el rango de grados para rotaciones aleatorias. Usamos un rango entre -180 y +180 grados.

- Rango de cambio de ancho: desplaza aleatoriamente una imagen hacia la izquierda o hacia la derecha en un porcentaje proporcional del ancho de la imagen. Este valor se estableció en 0,2, es decir, el 20 % del ancho de la imagen.

- Rango de cambio de altura: similar a la transformación anterior, pero el cambio es hacia arriba o hacia abajo. Este valor se estableció en 0,2.

- Rango de zoom: permite variar el zoom de una imagen de forma aleatoria. Este valor se estableció en 0,2, es decir, el rango de zoom se encuentra entre 80 % y 120 %.

- Volteo horizontal: permite que una imagen se voltee horizontalmente de forma aleatoria.

- Volteo vertical: permite voltear verticalmente una imagen aleatoriamente.

A través de este proceso, aumentamos las instancias del conjunto de datos para que cada clase tenga aproximadamente 1500 imágenes, incluidas imágenes originales, imágenes aumentadas y fotogramas de video para aumentar la oportunidad de un mejor rendimiento del modelo. Dado que los videos contienen parásitos en movimiento, que son visualmente similares a las imágenes resultantes de los procedimientos de aumento de datos, evitamos el uso de estrategias de aumento de datos en muestras de video.
Finalmente, para la clase rhyti, no se utilizaron algoritmos de aumento de datos porque las instancias de esta clase junto con los fotogramas de vídeo ya alcanzaron más de 2000 imágenes. Luego de realizar estos procedimientos, se obtuvieron 10099 imágenes con la distribución que se muestra en Tabla 2.2.

Tabla 2.2: Número total de imágenes en la base de datos de parásitos después del aumento de datos

Clases de Parásitos	Imágenes Originales	Imágenes Aumentadas	Fotogramas de Video	Imágenes Totales
ophi	245	118	1236	1599
blas	950	550	0	1500
oxi	345	402	888	1635
rhyti	1072	0	1048	2120
strong	648	262	764	1674
tae	356	302	913	1571
Total	3616	1634	4849	**10099**

2.2.3. Clasificación de la región mediante la transferencia de aprendizaje con MobileNet

Dado que Mobilenet fue diseñado originalmente para clasificar 1000 clases, hemos adaptado su arquitectura a nuestro problema de seis clases. Por lo tanto, reemplazamos la última capa de clasificación después de la capa de agrupación promedio con una capa densa de seis salidas (una para cada clase de parásito) con una función de activación softmax. La arquitectura Mobilnet adaptada a nuestro problema de clasificación se muestra en la Figura 2.4. A partir de esta figura, debe tenerse en cuenta que incluimos una capa de dropout con una tasa de abandono de 25 % para mejorar el error de generalización y evitar el sobreajuste [51].

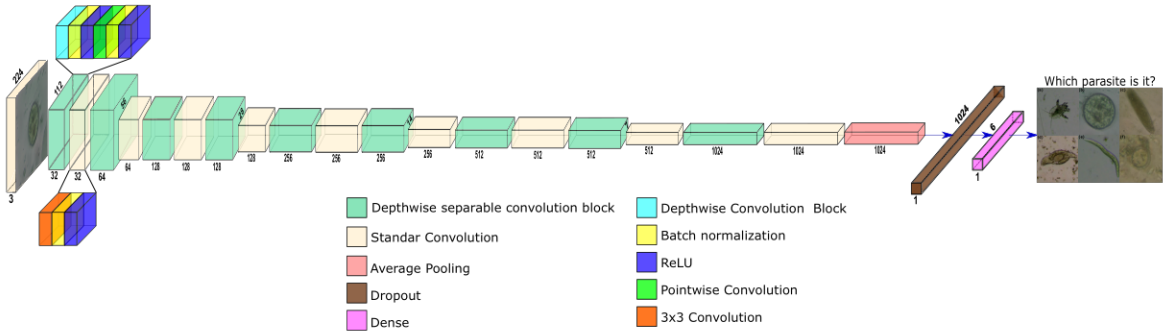


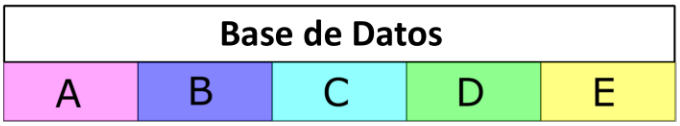
Figura 2.4: Arquitectura de MobileNet.

2.3. Configuración experimental

Esta sección describe la metodología de experimentación empleada para evaluar el enfoque propuesto, como la partición de datos, el aumento de datos, la configuración de la red neuronal, la comparación con un modelo RNC personalizado y las métricas de evaluación.

2.3.1. Partición de datos

Usamos el conjunto de datos experimental descrito en Tabla 2.2, que se compone de 10099 instancias que incluyen imágenes originales, imágenes aumentadas y fotogramas de vídeo. Aplicamos un método estratificado de validación cruzada de cinco veces [52] ($k = 5$) en el conjunto de datos experimentales para asegurar conjuntos disjuntos (entrenamiento y prueba) y la representación proporcional de las seis clases de parásitos en cada pliegue. Observe la Figura 2.5 donde se representa dicha partición de los datos.



Entrenamiento: 85% ABCD + frames	**Validación** 15% ABCD	**Prueba** (E)
Entrenamiento: 85% ABCE + frames	**Validación** 15% ABCE	**Prueba** (D)
Entrenamiento: 85% ABDE + frames	**Validación** 15% ABDE	**Prueba** (C)
Entrenamiento: 85% ACDE + frames	**Validación** 15% ACDE	**Prueba** (B)
Entrenamiento: 85% BCDE + frames	**Validación** 15% BCDE	**Prueba** (A)

Figura 2.5: Diagrama de validación cruzada k-fold.

2.3.2. Configuración de la red neuronal

Dado que utilizamos la transferencia de aprendizaje con MobileNet que se entrenó previamente con diferentes imágenes (es decir, del conjunto de datos de ImageNet), se realizó un proceso de ajuste de parámetros para todas las capas de la red. Otros hiperparámetros y configuraciones se establecieron como:

- Algoritmo de optimización: se utilizó el método Adam ya que funciona mejor que el descenso degradante estocástico común. Adapta la tasa de aprendizaje mientras se entrena para diferentes parámetros de las estimaciones de primer y segundo momento de los gradientes [53].
- Tasa de aprendizaje: este hiperparámetro permite actualizar las ponderaciones para cada época durante el entrenamiento de una red neuronal. Usamos una tasa de aprendizaje pequeña como sugiere [54] con un valor inicial de 0.001. Este hiperparámetro se redujo en un factor de 0,5 si la red no mejoraba su precisión en 2 épocas.
- Función de pérdida: dado que este problema de clasificación es multiclase, se eligió la entropía cruzada categórica como función de pérdida, ya que conduce a un entrenamiento más rápido, así como a una generalización mejorada para las tareas de clasificación [55].

2.3.3. Modelo RNC de referencia

Entrenamos una RNC personalizada, construida desde cero para compararla con el esquema de aprendizaje por transferencia con MobileNet. Esta red recibe imágenes de 224x224 como entrada y tiene seis clases de salidas, similar a la red MobileNet. La red personalizada tiene bloques convolucionales C como se muestra en la Figura 2.6. Cada bloque se compone de las siguientes capas:

- Capa convolucional con filtros F de tamaño 3×3. Establecemos los valores de zancadas en 1x1 y relleno de ceros de modo que la salida tenga las mismas dimensiones que la entrada.
- Capa de normalización por lotes (BN) que actúa como un regularizador para evitar el sobreajuste y principalmente permite el entrenamiento con tasas de aprendizaje más altas, que es la causa de una convergencia más rápida y una mejor generalización [56].
- La capa Leaky ReLU con una pendiente de 0.3 tiene una velocidad de cálculo y una tasa de convergencia más rápidas, a diferencia de otras funciones de activación como

la sigmoidea, gracias a su linealidad. Además, esta capa se introdujo para evitar la desaparición del problema del gradiente, ya que esta capa no causa saturación para las entradas negativas y positivas [36].

- La capa Max Pooling permite reducir la dimensionalidad de los mapas de características al resumir la presencia más activa de una característica [57].
- Dropout Layer con una tasa de 0.25, ayuda a evitar el sobreajuste de [51].

Después de los bloques convolucionales, se agregaron una capa plana y una capa densa de 512 nodos con activación de Leaky ReLU. Finalmente, la capa de salida es una capa densa de seis nodos con activación softmax para discriminar cada uno de los seis tipos de parásitos.

Variamos C de 3 a 6 bloques convolucionales manteniendo constantes los filtros $F = 128$ para explorar cómo la profundidad afecta el rendimiento de la red. Luego, mantuvimos constantes $C = 6$ bloques convolucionales y variamos F para 32, 64 y 128 filtros para analizar el impacto del tamaño del filtro en la red.

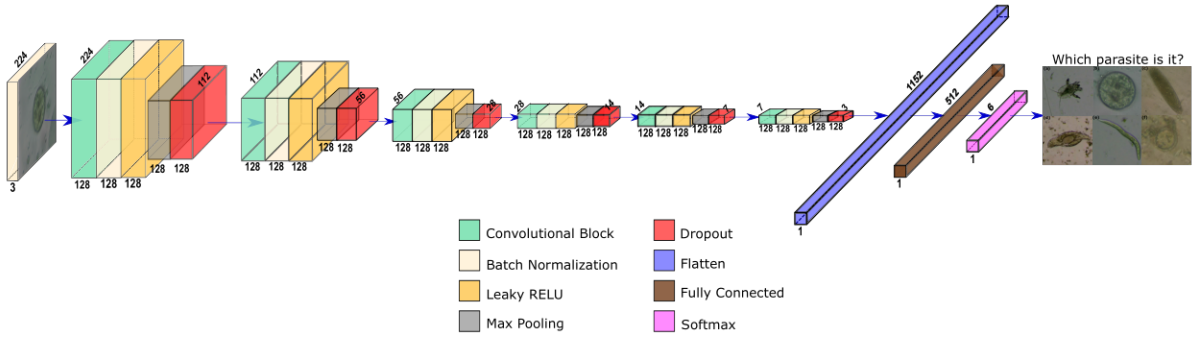


Figura 2.6: RNC personalizada con $C = 6$ bloques y $F = 128$ filtros.

2.3.4. Métricas de evaluación

Usamos la métrica Area Under the Curve (AUC) obtenida de la curva ROC (Receiver Operating Characteristic) en las mismas particiones de validación cruzada tanto para la RNC personalizada como para las arquitecturas de transferencia de aprendizaje.

Además, para garantizar una comparación justa y estadísticamente confiable, repetimos cuatro veces (con diferentes semillas aleatorias) el esquema de partición de validación cruzada, dando un total de 20 ejecuciones para cada arquitectura de red neuronal. Dado que nos ocupamos de un problema multiclase, utilizamos el AUC micropromedio para comparar el esquema de transferencia de aprendizaje con el RNC personalizado. Finalmente, calcu-

lamos la matriz de confusión a partir de una ejecución de validación cruzada de cinco veces y la precisión general de todas las ejecuciones.

3. RESULTADOS, CONCLUSIONES Y RECOMENDACIONES

3.1. Resultados

Antes de presentar los resultados con nuestro esquema de aprendizaje por transferencia, presentaremos los resultados de la optimización de hiperparámetros del número de bloques convolucionales C y el número de filtros F de la RNC personalizada. En Tabla 3.1, mostramos la precisión promedio para diferentes valores de C y F en las 20 ejecuciones con diferentes particiones de datos. También mostramos el número de parámetros entrenables para cada experimento.

Tabla 3.1: Precisión y entre paréntesis la desviación estándar con su número respectivo de parámetros entrenables y valores p. La precisión es el promedio de las 20 ejecuciones con diferentes particiones de datos aleatorios. Los valores p se calculan utilizando 128 filtros y 6 bloques de convolución como pivote

Filtros F	**Bloques Conv** C	**Presición (sd)**	**Parámetros Entrenables**	**Valor-P**
128	3	95.91 (2.379)	51,683,334	0.06858
128	4	95.88 (2.636)	13,296,006	0.12889
128	5	94.56 (3.696)	3,810,054	0.07352
128	6	95.30 (3.234)	1,336,454	–
64	6	93.68 (4.256)	485,702	1.377e-08
32	6	88.69 (5.459)	198,566	0.012

Observe que a medida que aumenta el número de filtros, también aumenta la precisión promedio. Para seleccionar el mejor modelo de RNC personalizado, seleccionamos el que tiene la mejor precisión promedio. En caso de que haya resultados estadísticamente similares según la prueba t con una precisión del 95 %, elegimos el que tiene el menor número de parámetros entrenables. Según esta estrategia, el mejor modelo se logra con 128 filtros y 6 bloques convolucionales. A continuación, los resultados que comparen MobileNet con la RNC personalizada se basarán en este mejor modelo.

La Figura 3.1 muestra la incrustación bidimensional aprendida por MobileNet antes de la capa softmax, donde cada punto representa las características de una imagen de forma bidimensional y un color por tipo de parásito. Observe que las características aprendidas forman grupos bastante distintivos, lo que demuestra que Mobilenet es capaz de aprender

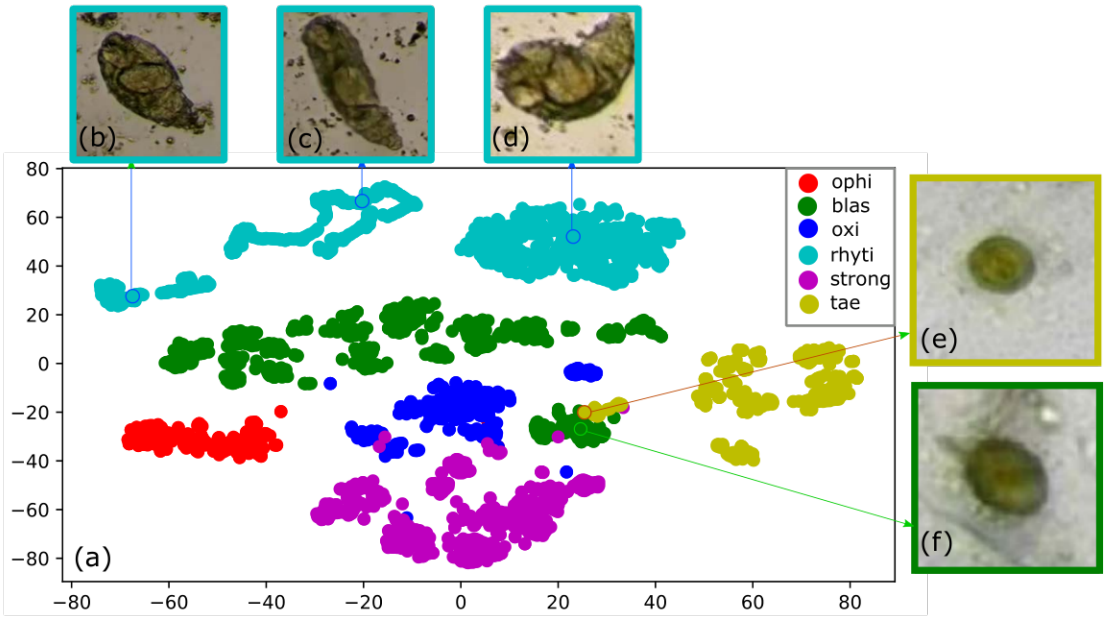


Figura 3.1: a) La gráfica T-SNE muestra que MobileNet está aprendiendo una representación significativa de las clases de imágenes. Las Figuras b), c) y d) muestran parásitos representativos de tres grupos diferentes de la clase rhyti. Las figuras e) y f) representan algunos ejemplos de los parásitos tae y blas, respectivamente, de regiones superpuestas para mostrar su similitud.

una representación útil de las imágenes.

Vale la pena señalar que existe cierta superposición entre las clases debido a las similitudes en la morfología y el color. Por ejemplo, observe en la gráfica de T-SNE cómo algunas imágenes de Blas se superponen a imágenes de Tae debido a sus similitudes visuales, como se muestra en la Figura 3.1 e) y la Figura 3.1 f) para Clases de Tae y Blas respectivamente. También tenga en cuenta que la incrustación de T-SNE tiende a formar tres grupos diferentes para la clase Rhyti. Estos tres grupos de Rhyti pueden explicarse por el movimiento natural del parásito. Para ejemplificar esto, observe los tres ejemplos representativos de Rhyti de la Figura 3.1 b), c) y d) tomados de estos tres grupos de Rhyti. Cuando el parásito fue observado en vídeo, podía estar en una posición retraída como se muestra en la Figura 3.1 b. A medida que evoluciona el movimiento del parásito, el parásito puede tomar una forma más alargada como en la Figura 3.1 c). Finalmente, este parásito también puede encontrarse a menudo en forma curva (Figura 3.1 d), que es su posición de alimentación [18]. Aquí, es importante señalar que la red neuronal fue capaz de aprender las etapas de movimiento de este parásito durante el entrenamiento.

En la Figura 3.2, observe la representacion de T-SNE para la RNC personalizada. Aunque el gráfico T-SNE tiende a agrupar parásitos similares, observe cómo las clases se superponen mucho más en comparación con el T-SNE de MobileNet. Concretamente, las clases Strong,

Oxi y Tae se confunden entre sí con mayor frecuencia. También hay confusión entre Tae y Blas en menor grado.

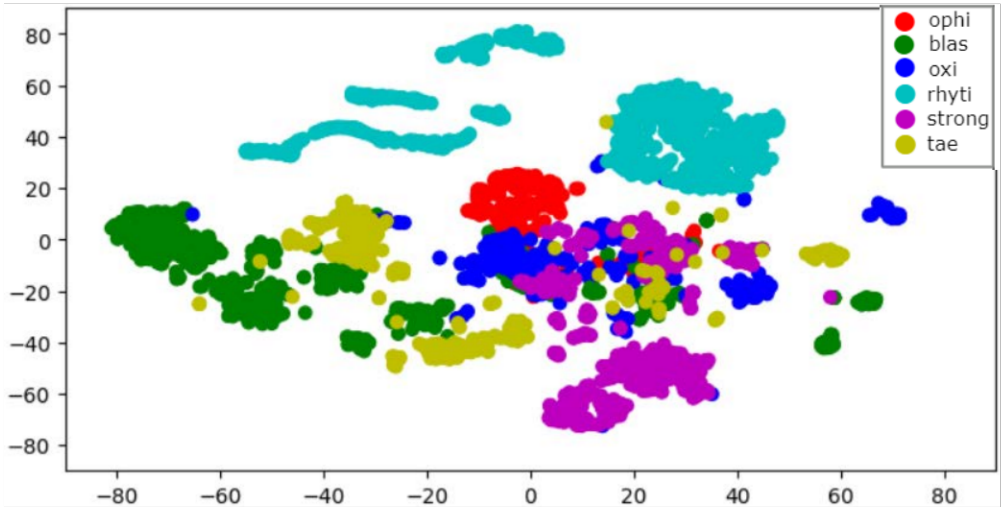


Figura 3.2: Gráfico T-SNE del modelo multiclase de la RNC personalizada

La Figura 3.3 muestra una representación de la tasa de verdaderos positivos (TPR) frente a la tasa de falsos positivos (FPR) mediante las curvas ROC de cada clase para MobileNet, donde se alcanza el AUC más bajo de 99,919 % para el parásito Oxi. y el AUC más alto del 100 % para el parásito Rhyti. Asimismo, muestra el micropromedio que suma los aportes de todas las clases antes de calcular la precisión promedio, obteniendo un valor de 99,972 %.

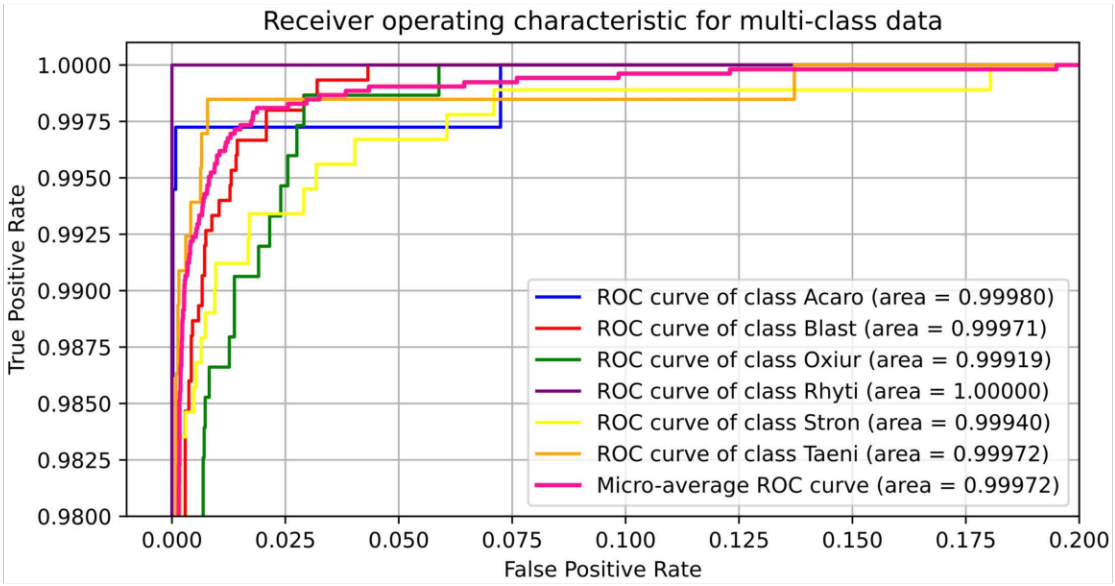


Figura 3.3: Gráfica ROC para el modelo MobileNet

La Figura 3.4 muestra las curvas ROC similares para la RNC personalizada, obteniendo el AUC más bajo de 99,617 % para el parásito Oxi y el AUC más alto de 99,999 % para el Rhyti, y un valor micro promedio de 99,844 %.Al comparar los dos gráficos, la precisión es

menor en la RNC personalizada que en el aprendizaje por transferencia Además, la Figura 3.5 muestra la matriz de confusión obtenida con MobileNet. La Figura 3.6 muestra la matriz de confusión obtenida con MobileNet normalizada.

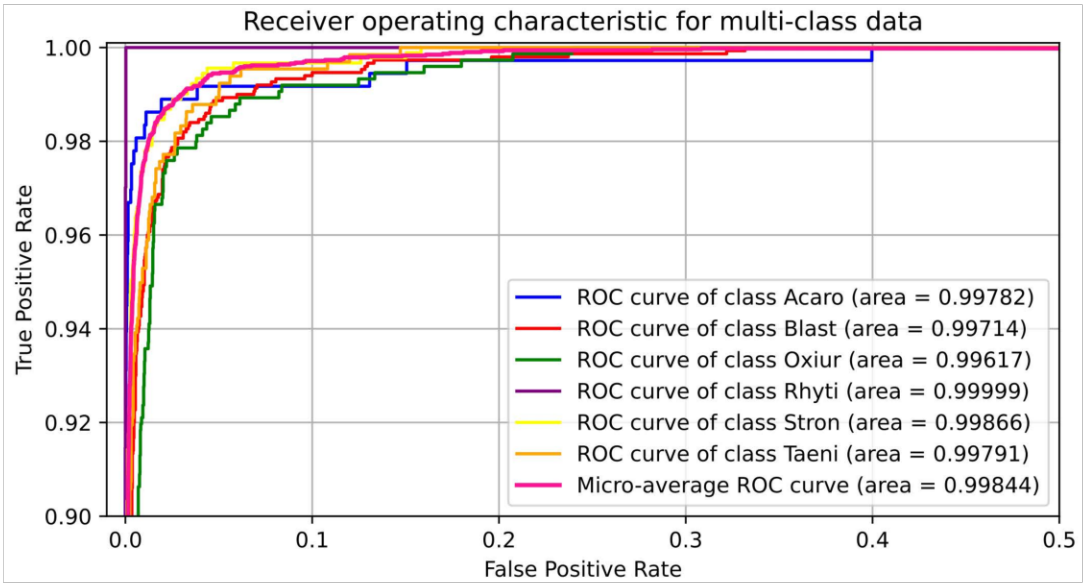


Figura 3.4: Gráfico ROC para la RNC personalizada

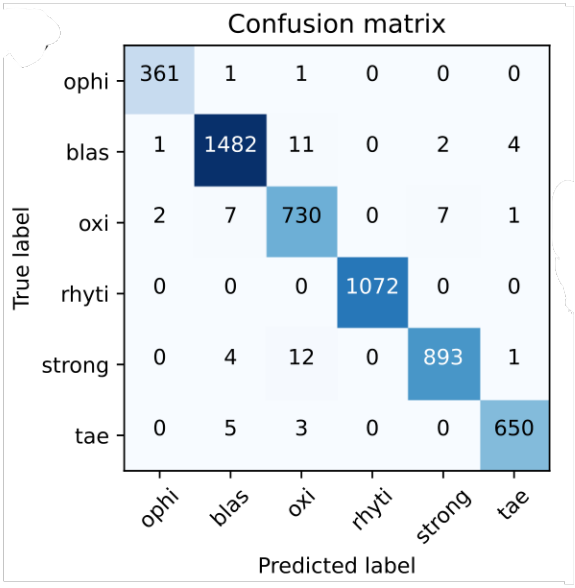


Figura 3.5: Matriz de confusión no normalizada del modelo multiclase de MobileNet.

Se observa que el clasificador distingue las clases de manera efectiva. Además, se observa un alto grado de confusión entre la clase Oxi y las clases Blas y Strong. También se observa cierta confusión entre las clases de Tae y Blas. Se espera este resultado ya que ambas clases se superponen en la incrustación de T-SNE como se muestra en. Figura 3.1

Con respecto a la RNC personalizada, la Figura 3.7 muestra su matriz de confusión norma-

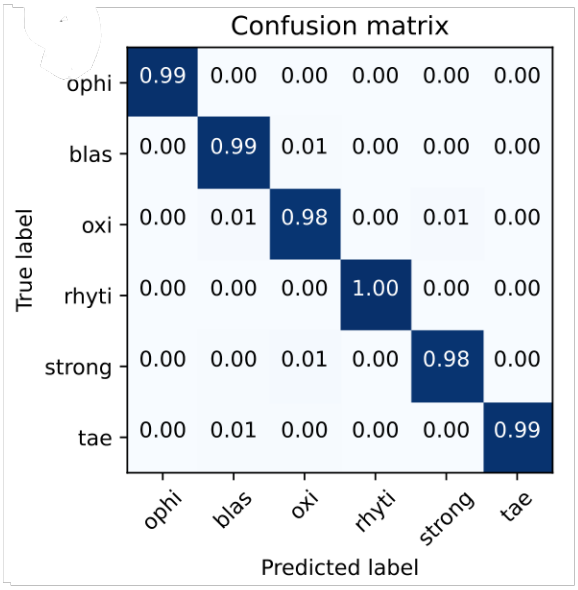


Figura 3.6: Matriz de confusión normalizada del modelo multiclase de MobileNet.

lizada y la Figura 3.8 su matriz de confusión normalizada. En comparación con MobileNet, la RNC personalizada tiene menor precisión en todas las clases. Específicamente, la clase Oxi tiene una alta confusión con las clases Strong y Blas, y moderadamente con la clase Ophi. Asimismo, Blas se confunde a menudo con Strong y Tae. En general, existe confusión entre todas las clases excepto la clase Rhyti. Este resultado se espera de lo que se ha visto anteriormente en la incrustación de T-SNE (ver Figura 3.2).

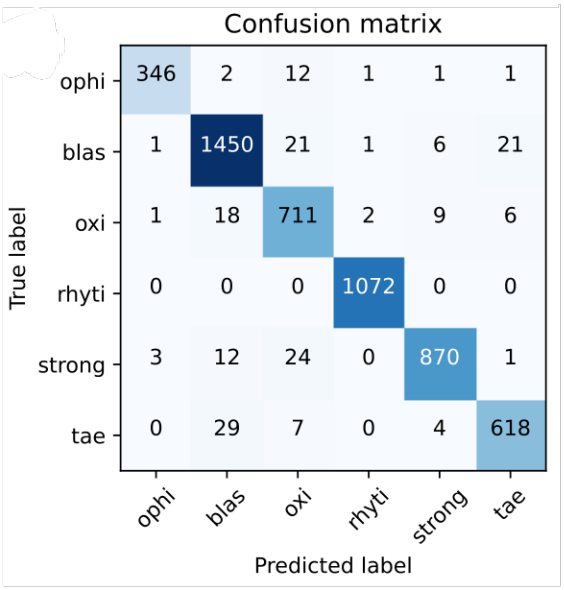


Figura 3.7: Matriz de confusión normalizada del modelo multiclase de la RNC personalizada.

Esta tendencia es confirmada por la Figura 3.2 que presenta la incrustación bidimensional de T-SNE a partir de las características aprendidas por la RNC personalizada. Se puede

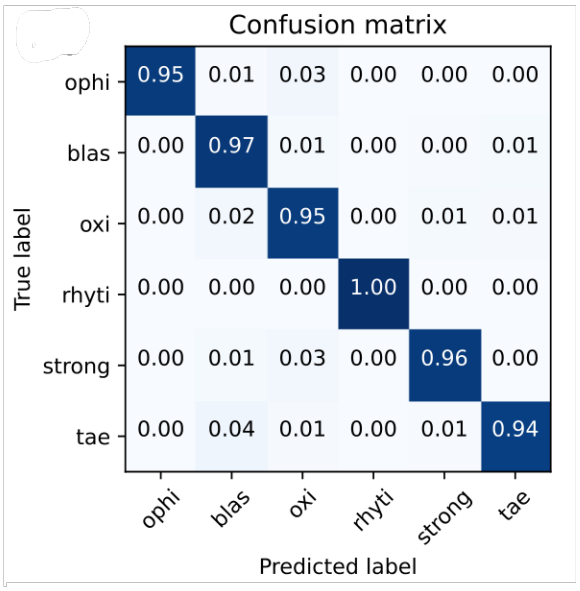


Figura 3.8: Matriz de confusión normalizada del modelo multiclase de la RNC personalizada.

observar de forma general que la red personalizada confunde a los parásitos en mayor medida que la MobileNet. Por lo tanto, la precisión es menor que el esquema de aprendizaje por transferencia.

Realizamos una prueba t para confirmar si el móvil supera estadísticamente al RNC personalizado. Se llevaron a cabo tres experimentos con MobileNet y los resultados se muestran en la Tabla 3.2. Se obtuvieron de precisión con el valor de precisión máximo de 98,66 % de la prueba MobileNet2. Por lo tanto, la precisión es mayor que el esquema RNC personalizado. Esas pruebas se tomaron para comparar con la prueba 128F6C de RNC personalizado, siendo 128F6C el pivote para obtener los valores P correspondientes a la tabla. Se observa que las pruebas de MobileNet son estadísticamente similares a la prueba 128F6C. Tomando estas consideraciones, se ha decidido tomar la prueba MobileNet2 como punto de referencia para comparar con el mejor resultado de RNC personalizado.

La Tabla 3.2 muestra la precisión tanto del esquema de aprendizaje de transferencia con Mobilnet como del RNC personalizado optimizado (es decir, 128 filtros y 6 bloques convolucionales). Observe que el valor de precisión obtenido por MobileNet (98,66 %) es mejor que el RNC personalizado optimizado. Realizamos una prueba t para confirmar si esta mejora es estadísticamente significativa. La prueba t confirma que la red móvil supera estadísticamente a la RNC personalizada optimizada (es decir, 128 filtros y 6 bloques convolucionales) a un nivel de significancia de 0.05, de acuerdo con el valor p que se muestra en la Tabla 3.2. Las Figuras 3.9 y 3.10 muestran las curvas de aprendizaje para el esquema de aprendizaje por transferencia y el RNC personalizado, respectivamente. Con MobiletNet, durante el

Tabla 3.2: Precisión, desviación estándar y valores p. La precisión es el promedio de las 20 ejecuciones con diferentes particiones de datos aleatorios. Los valores p se calculan utilizando RNC personalizado optimizado como pivote

Red Neuronal	Precisión	Desviación Estándar	P-value
MobileNet	98.66	0.404	0.0001304
RNC personalizado optimizado	95.30	3.234	-

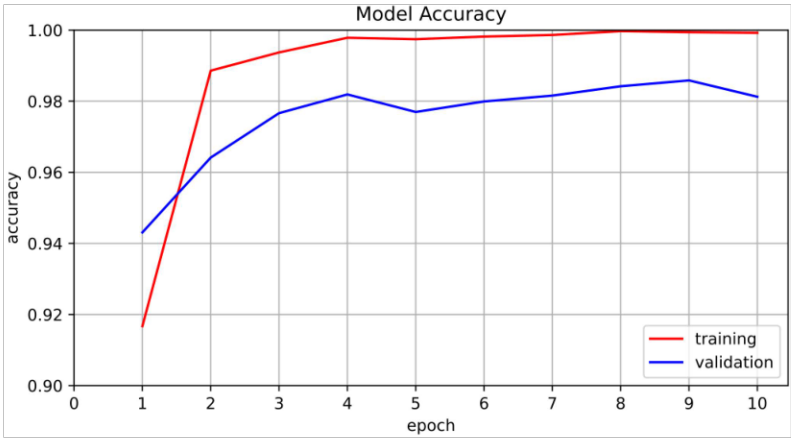


Figura 3.9: Curva de aprendizaje para MobileNet.

entrenamiento, la red neuronal logra un mejor rendimiento antes de 10 épocas, mientras que el RNC personalizado optimizado tarda más de 30 épocas. Esto se debe a que MobileNet, al ser una red previamente capacitada, no aprende desde cero como lo hace la RNC personalizada.

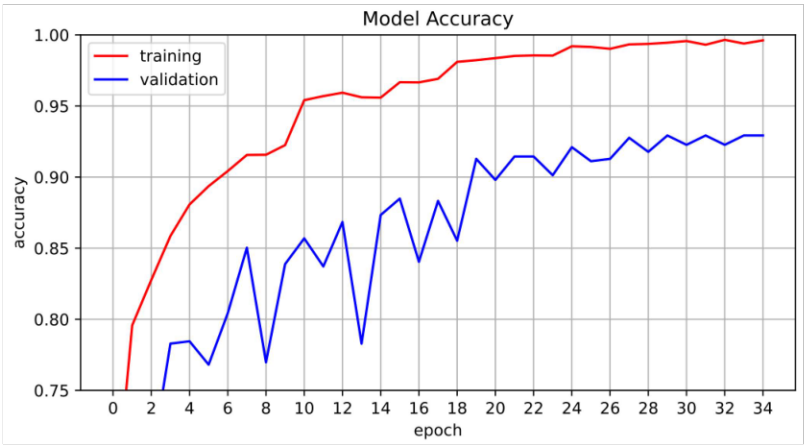


Figura 3.10: Curva de aprendizaje para la RNC personalizada.

3.2. Conclusiones

- El sistema propuesto para la clasificación de agentes parasitarios fue diseñado para obtener un desempeño adecuado utilizando un modelo con recursos limitados como MobileNet.

- Los mecanismos de segmentación de imágenes eran una tarea de vital importancia, ya que la mayoría de las imágenes, debido a la naturaleza de su adquisición, tenían escombros o basura que les introducían un ruido excesivo.

- El uso de aumento de datos a través de fotogramas de vídeo para la etapa de entrenamiento fue crucial para mejorar el rendimiento de la red, debido al tamaño limitado de la base de datos. Además, los algoritmos de aumento de datos abordaron el problema de clases desequilibradas que se encuentra en este conjunto de datos.

- Según los resultados, la implementación de un sistema que permita la clasificación de parásitos es factible ya que los porcentajes de precisión están por encima de lo aceptable para el modelo MobileNet y el modelo CNN personalizado, y el porcentaje de confusión entre las clases del modelo es bajo teniendo en cuenta el tamaño limitado de la base de datos.

- El desarrollo de varias pruebas con la CNN personalizado permite encontrar el modelo adecuado que pueda competir con MobileNet, buscando un bajo costo computacional y una gran precisión. Se determinó que la prueba de 128 filtros y 6 bloques computacionales es la CNN personalizada optimizada ya que es la CNN personalizada con la menor cantidad de parámetros entrenables y la mejor precisión.

- Nuestros resultados muestran que MobileNet supera a una CNN personalizada entrenada desde cero, lo que demuestra que los esquemas de aprendizaje por transferencia son adecuados para aprender características relevantes en este dominio.

3.3. Recomendaciones

- En trabajos futuros es posible implementar un sistema de detección y clasificación para un grupo mayor de clases, ya que como se mencionó en la etapa de recolección de información, se identificaron un total de 42 parásitos, que por su limitado número de imágenes, solo 6 tipos de parásitos se clasificaron.

- Dado que la obtención de datos etiquetados de reptiles parásitos es una tarea intensiva y principalmente manual realizada por veterinarios expertos, debería ser interesante explorar técnicas de aumento de datos más avanzadas, como las que utilizan la red generativa adversarial (GANS) [58].

- Además, un estudio futuro podría explorar nuevas formas de segmentación (p. Ej., Segmentación U-net [59]) que podrían mejorar el rendimiento general del sistema.

4. REFERENCIAS BIBLIOGRÁFICAS

[1] F. Jorge, M. A. Carretero, V. Roca, R. Poulin, and A. Perera, “What you get is what they have? detectability of intestinal parasites in reptiles using faeces,” *Parasitology Research*, vol. 112, no. 12, pp. 4001–4007, 2013.

[2] M. Khatun, N. Begum, M. Mamun, M. Mondal, and M. Azam, “Coprological study of gastrointestinal parasites of captive animals at rangpur recreational garden and zoo in bangladesh,” *Journal of Threatened Taxa*, vol. 6, no. 8, pp. 6142–6147, 2014.

[3] D. S. Bower, L. A. Brannelly, C. A. McDonald, R. J. Webb, S. E. Greenspan, M. Vickers, M. G. Gardner, and M. J. Greenlees, “A review of the role of parasites in the ecology of reptiles and amphibians,” *Austral Ecology*, vol. 44, no. 3, pp. 433–448, 2019.

[4] E. Jacobson and M. Garner, *Diseases and Pathology of Reptiles: Color Atlas and Text, Two Volume Set*. CRC Press, 2021.

[5] D. R. Mader and S. J. Divers, *Current Therapy in Reptile Medicine and Surgery-E-Book*. Elsevier Health Sciences, 2013.

[6] J. A. Mendoza-Roldan, D. Modry, and D. Otranto, “Zoonotic parasites of reptiles: a crawling threat,” *Trends in parasitology*, vol. 36, no. 8, pp. 677–687, 2020.

[7] Q. Li, W. Cai, X. Wang, Y. Zhou, D. D. Feng, and M. Chen, “Medical image classification with convolutional neural network,” in *2014 13th international conference on control automation robotics & vision (ICARCV)*. IEEE, 2014, pp. 844–848.

[8] K. Weiss, T. M. Khoshgoftaar, and D. Wang, “A survey of transfer learning,” *Journal of Big data*, vol. 3, no. 1, pp. 1–40, 2016.

[9] E. Valle, M. Fornaciali, A. Menegola, J. Tavares, F. V. Bittencourt, L. T. Li, and S. Avila, “Data, depth, and design: Learning reliable models for melanoma screening,” *arXiv preprint arXiv:1711.00441*, vol. 1, pp. 770–778, 2017.

[10] J. Y. Kim, H. E. Lee, Y. H. Choi, S. J. Lee, and J. S. Jeon, “Cnn-based diagnosis models for canine ulcerative keratitis,” *Scientific reports*, vol. 9, no. 1, pp. 1–7, 2019.

[11] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, “A survey on deep transfer learning,” in *International conference on artificial neural networks*. Springer, 2018, pp. 270–279.

[12] K. Espinoza, D. L. Valera, J. A. Torres, A. Lopez, and F. D. Molina-Aiz, “Combination of image processing and artificial neural networks as a novel approach for the identification of bemisia tabaci and frankliniella occidentalis on sticky traps in greenhouse agriculture,” *Computers and Electronics in Agriculture*, vol. 127, pp. 495–505, 2016.

[13] J. Cho, J. Choi, M. Qiao, C. Ji, H. Kim, K. Uhm, and T. Chon, “Automatic identification of whiteflies, aphids and thrips in greenhouse based on image analysis,” *Red*, vol. 346, no. 246, p. 244, 2007.

[14] M. I. Razzak, “Automatic detection and classification of malarial parasite,” *International Journal of Biometrics and Bioinformatics (IJBB)*, vol. 9, no. 1, pp. 1–12, 2015.

[15] A. Hanbury, “The morphological top-hat operator generalised to multi-channel images,” in *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, vol. 1. IEEE, 2004, pp. 672–675.

[16] B. Ramalingam, R. E. Mohan, S. Pookkuttath, B. F. Gómez, C. S. C. Sairam Borusu, T. Wee Teng, and Y. K. Tamilselvam, “Remote insects trap monitoring system using deep learning framework and iot,” *Sensors*, vol. 20, no. 18, p. 5280, 2020.

[17] Y. Ginoris, A. Amaral, A. Nicolau, M. Coelho, and E. Ferreira, “Recognition of protozoa and metazoa using image analysis tools, discriminant analysis, neural networks and decision trees,” *Analytica Chimica Acta*, vol. 595, no. 1-2, pp. 160–169, 2007.

[18] K. A. Núñez Alverca, “Identificación de parásitos con diferentes métodos coprológicos en muestras de reptiles en el vivarium de quito,” B.S. dissertation, Amato: UTA, 2021.

[19] H. Inoue, “Data augmentation by pairing samples for images classification,” *arXiv preprint arXiv:1801.02929*, 2018.

[20] T. B. Arnold, “kerasr: R interface to the keras deep learning library,” *Journal of Open Source Software*, vol. 2, no. 14, p. 296, 2017.

[21] S. M. Pizer, E. P. Amburn, J. D. Austin, R. Cromartie, A. Geselowitz, T. Greer, B. ter Haar Romeny, J. B. Zimmerman, and K. Zuiderveld, “Adaptive histogram equalization and its variations,” *Computer vision, graphics, and image processing*, vol. 39, no. 3, pp. 355–368, 1987.

[22] C. Kanan and G. W. Cottrell, “Color-to-grayscale: does the method matter in image recognition?” *PloS one*, vol. 7, no. 1, p. e29740, 2012.

[23] N. Otsu, “A threshold selection method from gray-level histograms,” *IEEE transactions on systems, man, and cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[24] I. Ragnemalm, “Fast erosion and dilation by contour processing and thresholding of distance maps,” *Pattern recognition letters*, vol. 13, no. 3, pp. 161–166, 1992.

[25] X.-D. Zhang, *A matrix algebra approach to artificial intelligence*. Springer, 2020.

[26] P. Cunningham, M. Cord, and S. J. Delany, “Supervised learning,” in *Machine learning techniques for multimedia*. Springer, 2008, pp. 21–49.

[27] S. García, A. Fernández, J. Luengo, and F. Herrera, “A study of statistical techniques and performance measures for genetics-based machine learning: accuracy and interpretability,” *Soft Computing*, vol. 13, no. 10, p. 959, 2009.

[28] P. Martínez-Camblor, “Comparación de pruebas diagnósticas desde la curva roc,” *Revista Colombiana de Estadística*, vol. 30, no. 2, pp. 163–176, 2007.

[29] S. Visa, B. Ramsay, A. L. Ralescu, and E. Van Der Knaap, “Confusion matrix-based feature selection.” *MAICS*, vol. 710, pp. 120–127, 2011.

[30] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An introduction to statistical learning*. Springer, 2013, vol. 112.

[31] L. Van der Maaten and G. Hinton, “Visualizing data using t-sne.” *Journal of machine learning research*, vol. 9, no. 11, 2008.

[32] B. Zhao, H. Lu, S. Chen, J. Liu, and D. Wu, “Convolutional neural networks for time series classification,” *Journal of Systems Engineering and Electronics*, vol. 28, no. 1, pp. 162–169, 2017.

[33] J. Wu, “Introduction to convolutional neural networks,” *National Key Lab for Novel Software Technology. Nanjing University. China*, vol. 5, no. 23, p. 495, 2017.

[34] K. O’Shea and R. Nash, “An introduction to convolutional neural networks,” *arXiv preprint arXiv:1511.08458*, 2015.

[35] J. Bjorck, C. Gomes, B. Selman, and K. Q. Weinberger, “Understanding batch normalization,” *arXiv preprint arXiv:1806.02375*, 2018.

[36] J. Xu, Z. Li, B. Du, M. Zhang, and J. Liu, “Reluplex made more practical: Leaky relu,” in *2020 IEEE Symposium on Computers and Communications (ISCC)*. IEEE, 2020, pp. 1–7.

[37] P. Baldi and P. J. Sadowski, “Understanding dropout,” *Advances in neural information processing systems*, vol. 26, pp. 2814–2822, 2013.

[38] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, “Mobilenets: Efficient convolutional neural networks for mobile vision applications,” *arXiv preprint arXiv:1704.04861*, 2017.

[39] S. Targ, D. Almeida, and K. Lyman, “Resnet in resnet: Generalizing residual architectures,” *arXiv preprint arXiv:1603.08029*, 2016.

[40] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.

[41] E. Díaz García and E. Cabrera Granado, “Manual de uso de jupyter notebook para aplicaciones docentes,” *E-Prints Complutense*, 2018.

[42] I. Challenger-Pérez, Y. Díaz-Ricardo, and R. A. Becerra-García, “El lenguaje de programación python,” *Ciencias Holguín*, vol. 20, no. 2, pp. 1–13, 2014.

[43] G. Zaccone, M. R. Karim, and A. Menshawy, *Deep learning with TensorFlow*. Packt Publishing Ltd, 2017.

[44] N. K. Manaswi, “Understanding and working with keras,” in *Deep Learning with Applications Using Python*. Springer, 2018, pp. 31–43.

[45] O. Kramer, “Scikit-learn,” in *Machine learning for evolution strategies*. Springer, 2016, pp. 45–53.

[46] E. Bressert, *SciPy and NumPy: an overview for developers*. .ºReilly Media, Inc.”, 2012.

[47] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright *et al.*, “Scipy 1.0: fundamental algorithms for scientific computing in python,” *Nature methods*, vol. 17, no. 3, pp. 261–272, 2020.

[48] K. Bhargavi and S. Jyothi, “A survey on threshold based segmentation technique in image processing,” *International Journal of Innovative Research and Development*, vol. 3, no. 12, pp. 234–239, 2014.

[49] A. N. Evans and X. U. Liu, “A morphological gradient approach to color edge detection,” *IEEE Transactions on Image Processing*, vol. 15, no. 6, pp. 1454–1463, 2006.

[50] H. T. Croft, K. Falconer, and R. K. Guy, *Unsolved problems in geometry: unsolved problems in intuitive mathematics*. Springer Science & Business Media, 2012, vol. 2.

[51] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: a simple way to prevent neural networks from overfitting,” *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[52] Y. Bengio and Y. Grandvalet, “No unbiased estimator of the variance of k-fold cross-validation,” *Journal of machine learning research*, vol. 5, no. Sep, pp. 1089–1105, 2004.

[53] S. Bock and M. Weiß, “A proof of local convergence for the adam optimizer,” in *2019 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2019, pp. 1–8.

[54] D. R. Wilson and T. R. Martinez, “The need for small learning rates on large problems,” in *IJCNN’01. International Joint Conference on Neural Networks. Proceedings (Cat. No. 01CH37222)*, vol. 1. IEEE, 2001, pp. 115–119.

[55] M. Svensén and C. M. Bishop, “Pattern recognition and machine learning,” 2007.

[56] T. Tran, O.-H. Kwon, K.-R. Kwon, S.-H. Lee, and K.-W. Kang, “Blood cell images segmentation using deep learning semantic segmentation,” in *2018 IEEE International Conference on Electronics and Communication Engineering (ICECE)*. IEEE, 2018, pp. 13–16.

[57] E. Grefenstette, P. Blunsom *et al.*, “A convolutional neural network for modelling sentences,” in *The 52nd Annual Meeting of the Association for Computational Linguistics, Baltimore, Maryland*, 2014.

[58] S. Kazeminia, C. Baur, A. Kuijper, B. van Ginneken, N. Navab, S. Albarqouni, and A. Mukhopadhyay, “Gans for medical image analysis,” *Artificial Intelligence in Medicine*, p. 101938, 2020.

[59] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

ANEXOS

Dado que los algoritmos de aprendizaje automático fueron entrenados en la plataforma Jupyter, se adjunta como anexo digital los scripts en extensión ”.ipynb” para cada etapa del trabajo y además la base de datos utilizada.

ANEXO A. Instrucciones para ejecutar el código fuente.

ANEXO I

Instrucciones para ejecutar el código fuente

URL del repositorio de datos

Por favor, descargue este repositorio a su computadora. El repositorio incluye el conjunto de datos y scripts de python.
Dataset y código fuente

¿Cómo empezar?

Este documento describe el uso de scripts y conjuntos de datos para reproducir los resultados del manuscrito titulado "Identificación automática de parásitos intestinales en reptiles usando imágenes microscópicas de heces y redes neuronales convolucionales".
Las imágenes y videos tomados por el microscopio se encuentran en el directorio `./dataSET1500/BASE1` (en adelante dataset `BASE1`). Las mismas imágenes tomadas por el microscopio y los fotogramas extraídos de los videos usando el editor de video Sony Vegas Pro 11 se pueden encontrar en el directorio `./dataSET1500/BASE2` (en adelante dataset `BASE2`). Este último conjunto de datos es el necesario para ejecutar los scripts.

1. Para la correcta ejecución de los scripts, siga las instrucciones a continuación:
   Se incluyen los siguientes cuadernos jupyter, que deben estar ubicados en su directorio de trabajo (es decir, directorio ./):

   - `SEGMENTATION + DATA AUGMENTATION.ipynb`
   - `Reptile parasites MOBILENET notebook.ipynb`
   - `Reptile parasites CUSTOM CNN notebook.ipynb`

2. Asegúrese de que el conjunto de datos `BASE2` se encuentra en el siguiente directorio:

   `./dataSET1500/BASE2`

3. Asegúrese de que dentro de la carpeta `BASE2` haya una carpeta llamada `Images` con 3616 imágenes de parásitos y otra carpeta llamada `Frames` con 4849 imágenes de parásitos extraídas de los videos.

4. Se recomienda instalar las siguientes versiones de bibliotecas de python en un entorno virtual de Anaconda:
   - `Numpy 1.20.1`
   - `Tensorflow 2.3.0`

- Matplotlib 3.3.4
- OpenCV 4.0.1
- Pandas 1.2.4
- Keras 2.4.3
- Scikit-learn 0.24.1
- More-itertools 8.7.0
- Scipy 1.6.2

5. El primer cuaderno que se ejecuta es `SEGMENTACIÓN + AUMENTO DE DATOS.ipynb`. Este cuaderno implementa la etapa de segmentación y aumento de datos. Después de ejecutar este cuaderno, las carpetas `BASE3` y `BASE4` y los archivos `baseFrames.csv` y `baseImages.csv` se generan en el directorio `./dataSET1500`. Los archivos `baseFrames.csv` y `baseImages.csv` contienen las etiquetas y los nombres de los frames y las imágenes, respectivamente. El conjunto de datos `BASE3` es el resultado de la etapa de segmentación del conjunto de datos a partir de las imágenes `BASE2`. El conjunto de datos `BASE4` es el resultado de la etapa de aumento de datos a partir de las imágenes `BASE3`. El último conjunto de datos es la entrada a las redes neuronales.

6. Hay dos cuadernos jupyter que implementaron el esquema de transferencia de aprendizaje con MobileNet y RNC personalizada, respectivamente.:
   - `Reptile parasites MOBILENET notebook.ipynb`
   - `Reptile parasites CUSTOM CNN notebook.ipynb`

   Tenga en cuenta que estos cuadernos se pueden ejecutar de forma independiente. Hay varias variables que se pueden cambiar según la prueba que desee realizar:

   - `RSEED`: variable que define la semilla aleatoria para la validación cruzada de k fold (ambos scripts).
   - `batch_size`: variable que define el tamaño del lote (ambos scripts).
   - `N_Epochs`: número de épocas para la etapa de entrenamiento (ambos scripts).
   - `nFilters`: número de filtros (solo RNC personalizada).
   - `nBlocks`: número de bloques convolucionales (solo RNC personalizada).

   Nota: Para obtener los mejores resultados que se muestran en el documento, se recomienda definir estas variables de la siguiente manera: RSEED = 7, batch_size = 32, N_Epochs = 50, nFilters = 128, nBlocks = 6.

   Después de ejecutar los scripts mencionados anteriormente, las variables y modelos se almacenan en el directorio `./dataSET1500/Test/`
   Las matrices de confusión y las curvas ROC se muestran al final de cada script.