FACULTAD DE INGENIERIA, ARQUITECTURA Y URBANISMO

ESCUELA PROFESIONAL DE INGENIERIA DE SISTEMAS

TESIS

Clasificación de la viruela del mono mediante aprendizaje profundo y técnicas de procesamiento de imágenes

PARA OPTAR EL TÍTULO PROFESIONAL DE INGENIERO DE SISTEMAS

Autor(es):

Bach. Alcantara Calderon Gianmarco

ORCID: https://orcid.org/0000-0001-8007-3768

Bach. Arica Guerrero Lauren David

ORCID: https://orcid.org/0000-0003-4393-7170

Asesor:

Dr. Tuesta Monteza Víctor Alexci

ORCID: https://orcid.org/0000-0002-5913-990X

Línea de investigación:

Ciencias de la información como herramientas multidisciplinares y estratégicas en el contexto industrial y de organizaciones

Sublínea de Investigación:

Informática y transformación digital en el contexto industrial y organizacional

Pimentel – Perú

2024

CLASIFICACION DE LA VIRUELA DEL MONO MEDIANTE APRENDIZAJE PROFUNDO Y TECNICAS DE PROCESAMIENTO DE IMAGENES

Aprobación del jurado

DR. TUESTA MONTEZA VICTOR ALEXCI

Presidente del Jurado de Tesis

MG. GUEVARA ALBURQUEQUE LAURITA BELEN

Secretario del Jurado de Tesis

MG. ARCILA DIAZ JUAN CARLOS

Vocal del Jurado de Tesis

NOMBRE DEL TRABAJO

TESIS_Arica-Alcantara nueva turnitin.doc x

RECUENTO DE PALABRAS

6318 Words

RECUENTO DE CARACTERES

33771 Characters

RECUENTO DE PÁGINAS

30 Pages

TAMAÑO DEL ARCHIVO

1.8MB

FECHA DE ENTREGA

Nov 11, 2024 10:45 AM GMT-5

FECHA DEL INFORME

Nov 11, 2024 10:46 AM GMT-5

15% de similitud general

El total combinado de todas las coincidencias, incluidas las fuentes superpuestas, para cada base de datos.

- 9% Base de datos de Internet
- 4% Base de datos de publicaciones
- Base de datos de Crossref
- Base de datos de contenido publicado de Crossref
- 10% Base de datos de trabajos entregados

Excluir del Reporte de Similitud

- Material bibliográfico
- Material citado
- Coincidencia baja (menos de 8 palabras)

Universidad Señor de Sipán

DECLARACIÓN JURADA DE ORIGINALIDAD

Quien(es) suscribe(n) la DECLARACIÓN JURADA, somos **Alcantara Calderon Gianmarco y Arica Guerrero Lauren David,** del Programa de Estudios de **Ingeniería de Sistemas** de la Universidad Señor de Sipán S.A.C, declaramos bajo juramento que somos autores del trabajo titulado:

CLASIFICACIÓN DE LA VIRUELA DEL MONO MEDIANTE APRENDIZAJE PROFUNDO Y TECNICAS DE PROCESAMIENTO DE IMAGENES

El texto de mi trabajo de investigación responde y respeta lo indicado en el Código de Ética del Comité Institucional de Ética en Investigación de la Universidad Señor de Sipán, conforme a los principios y lineamientos detallados en dicho documento, en relación con las citas y referencias bibliográficas, respetando el derecho de propiedad intelectual, por lo cual informo que la investigación cumple con ser inédito, originaly autentico.

En virtud de lo antes mencionado, firman:

Alcantara Calderon Gianmarco	74727547	
Arica Guerrero Lauren David	73581496	

Pimentel, 15 de Diciembre de 2023.

DEDICATORIA

Dedico esta tesis a mis padres, José y Lorena, quienes siempre han estado apoyándome a seguir adelante durante todo este camino. A mi familia quienes a lo largo de mi vida siempre han sido mi fuente de fortaleza y motivación para lograr todos mis objetivos. Por último, mis amigos y seres queridos a quienes les expreso mi más sincero agradecimiento por haber confiado en mí. Este logro no hubiera sido posible sin ustedes, gracias de corazón.

Gianmarco Alcantara Calderon

Dedico esta tesis, primeramente, a Dios por darme la fortaleza y sabiduría para seguir adelante, incluso en los momentos difíciles. A mis padres, José de la Rosa Arica Cunias y Eva María Guerrero Ñañez, por estar siempre a mi lado, apoyándome en cada paso de este camino. Su amor incondicional, su constante apoyo y sacrificio han sido mi mayor fuente de inspiración. A mi hermano Harold y a mi hermana Camila, por su aliento y motivación constante. A mi familia y amigos, quienes siempre han estado presentes con su cariño y compresión. Este logro refleja el respaldo que me han brindado, y por ello les expreso mi más sincero agradecimiento y gratitud.

Lauren David Arica Guerrero

AGRADECIMIENTOS

Expresamos nuestros más sinceros agradecimientos a la Universidad Señor de Sipán por el respaldo incondicional y los recursos brindados durante todo nuestro proceso de formación académica. También un reconocimiento a los docentes de la universidad, cuyas enseñanzas y orientaciones fueron fundamentales para nuestro crecimiento. Finalmente, un agradecimiento especial a nuestro asesor Víctor Alexci Tuesta Monteza por su invaluable orientación y apoyo a lo largo de este proceso. Este logro es, en gran medida, un reflejo del compromiso y dedicación de todos aquellos que nos ofrecieron su apoyo.

Los autores

Índice de Contenidos

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

El brote de la viruela del mono es una enfermedad causada por el virus Orthopox, es el más cercano al virus de la viruela, cuyos contagios siguen aumentando a nivel mundial. La presente investigación busca comparar el desempeño de tres modelos de aprendizaje profundo para clasificar dicha enfermedad. Para ello se empleó un conjunto de datos de un total de 1577 imágenes. Mediante una revisión de la literatura se seleccionaron los modelos de aprendizaje profundo empleados en el estudio, siendo estos una CNN personalizada, VGG16 y ResNet50. Posteriormente se dividió el dataset en 70% para el entrenamiento, 15% para la validación y 15% para la prueba. Los resultados indicaron que el modelo ResNet50 tuvo mejor desempeño con 98.00% en exactitud, 98.50% en precisión, 98.25% en recall y 98.50% en f1-score. Los hallazgos destacan la importancia del aprendizaje profundo para el desarrollo de herramientas diagnosticas, evidenciando la eficacia de ResNet50 para la clasificación de la enfermedad de la virtual del mono.

Palabras Clave: viruela del mono, aprendizaje profundo, modelos, clasificadores.

ABSTRACT

The outbreak of monkeypox is a disease caused by the Orthopox virus, which is the closest to the smallpox virus, whose infections continue to increase worldwide. The present research seeks to compare the performance of three deep learning models for classifying this disease. For this purpose, a dataset of a total of 1577 images was used. Through a literature review, the deep learning models used in the study were selected, being a customized CNN, VGG16 and ResNet50. The dataset was then divided into 70% for training, 15% for validation and 15% for testing. The results indicated that the ResNet50 model performed better with 98.00% in accuracy, 98.50% in precision, 98.25% in recall and 98.50% in f1-score. The findings highlight the importance of deep learning for the development of diagnostic tools, evidencing the efficacy of ResNet50 for the classification of virtual monkey disease.

Keywords: monkeypox, deep learning, models, classifiers.

I. INTRODUCCIÓN

El brote de la viruela del mono es una enfermedad causada por el virus Orthopox, es el más cercano al virus de la viruela [1]. A pesar de que este virus ya es conocido y ha sido tratado en el pasado, los contagios siguen aumentando a nivel mundial y los gobiernos están teniendo dificultades para controlarlo [2]. El primer caso notificado fue de un niño en la República Democrática del Congo en 1970 [3]. Desde entonces los contagios solo se dieron en el continente africano debido a que el virus es endémico, sin embargo, esta enfermedad ha sido reportado en lugares no endémicos como Europa y América [1]. La OMS (Organización Mundial de la Salud) ha tomado muy enserio la situación, ya que ha publicado orientaciones de salud pública, ha colaborado con comunidades y ha convocado a científicos e investigadores para acelerar la investigación sobre la viruela del mono para el desarrollo de tratamientos, vacunas y pruebas [4]. La viruela del mono fue declarado como una emergencia de salud pública de importancia internacional (ESPII) por la Organización Mundial de la Salud el 23 de julio de 2022, siendo la alerta más alta a nivel mundial [4]. Según el informe presentado por Centros para el Control y la Prevención de Enfermedades, la cantidad de casos confirmados por esta enfermedad a nivel mundial, hasta el 27 de septiembre de 2023 era de 90.630 casos y 161 decesos confirmados en 115 países, siendo Estados Unidos, Brasil y España los más afectados por este virus, cifras que actualmente siguen aumentando [5]. En el caso de Perú, las cifras de contagio confirmadas llegan a 3829 con un total de 20 fallecidos a nivel nacional [6]. Los estudios clínicos de la viruela del mono suelen ser difíciles en su análisis y hay cierta confusión con otras enfermedades como la varicela y el sarampión. La prueba de diagnóstico y análisis de la viruela del mono es realizada por un especialista y en laboratorios altamente especializados [7].

La prueba implica tomar muestras de la herida con un hisopo, realizar un hisopado orofaríngeo parecido al del covid-19 y llevar a cabo una prueba de sangre, con ello se realiza el análisis molecular para determinar si la enfermedad que tiene el paciente se trata de viruela del mono u otra enfermedad similar, dicho resultado del descarte se obtiene en un plazo de

24 horas [8].

Sin embargo, la pruebas de diagnóstico son de poca accesibilidad, ya que no están en los centros de salud y hospitales sino en laboratorios y clínicas [2]. Al ser una enfermedad cuyos números de casos siguen aumentando en el país, es necesario implementar nuevas herramientas tecnológicas que permitan detectar la enfermedad de forma rápida, accesible y precisa, ya que esta presenta síntomas muy similares a otras enfermedades, lo que puede conllevar a un diagnóstico erróneo. En dicho escenario la detección asistida por ordenador de lesiones provocadas por la viruela del mono podría ser de mucha utilidad para una detección rápida de casos sospechosos en zonas donde las pruebas de polimerasa (PCR) no están disponibles o son de difícil acceso. Además de ello [9] menciona que los modelos de aprendizaje profundo han demostrado su utilidad en casos anteriores para el diagnóstico automatizado de enfermedades cutáneas. Por otro lado, se puede evidenciar que existe poca o escasa cantidad de estudios de investigación que indiquen el potencial de los enfoques de aprendizaje profundo en el diagnóstico y clasificación del virus de la viruela del mono [10]. Considerando este contexto, el objetivo de la investigación es clasificar la viruela del mono mediante aprendizaje profundo y técnicas de procesamiento de imágenes, a raíz de ello surgió la necesidad de formular la siguiente pregunta de investigación ¿De qué manera se podrá clasificar la viruela del mono?, esto conllevó a plantear la siguiente hipótesis, la cual sostiene que mediante el modelo de aprendizaje profundo ResNet50 se clasifica la enfermedad de la viruela del mono. La investigación conlleva determinar el dataset de la viruela del mono, asimismo seleccionar los algoritmos de aprendizaje profundo, luego implementar los algoritmos de aprendizaje profundo para clasificar la viruela del mono, después evaluar los algoritmos de aprendizaje profundo implementados y finalmente desarrollar una aplicación web funcional para la clasificación de la viruela del mono. Este trabajo de investigación se justifica por la necesidad de mejorar y facilitar el diagnóstico de la viruela del mono especialmente en zonas con accesos limitados a pruebas y recursos médicos, además que tiene el potencial de brindar un valioso apoyo a los expertos del área médica.

Existe diferentes investigaciones que han tratado de dar solución a esta problemática, [9] utilizó tres características de CNN profundas (AlexNet, GoogleNet y Vgg16Net) junto con 5 algoritmos de Machine Learning para la clasificación: SVM, KNN, árbol de decisión, Naïve Bayes y Random Forest. Empleó un dataset con 228 imágenes RGB originales (102 de monkeypox y 126 de otras enfermedades cutáneas) y 3192 imágenes aumentadas (1428 de monkeypox y 1764 de otras enfermedades cutáneas). Los resultados indicaron que con las características de AlexNet, el clasificador Naïve Bayes ofrece una precisión máxima del 86,66%. Con GoogleNet, el clasificador Naive Bayes ofrece una precisión máxima del 62,22%. Y, con Vgg16Net, el clasificador Naive Bayes ofrece una precisión del 91,11%. Así mismo, [11] propuso un sistema basado en redes neuronales convolucionales (CNN) asistidas por inteligencia artificial explicable (xAI). Utilizó 2 clases de datasets: Monkeypox y Piel sana con un total de 572 imágenes obtenidas de Kaggle, para la clasificación emplearon ResNet-18, ResNet-50, VGG-16, Densenet-161, EfficientNet B7, EfficientNet V2, GoogLeNet MobileNet V2, MobileNet V3, ResNeXt-50, ShuffleNet V2 y ConvNeXt. De acuerdo a los resultados el modelo MobileNet V2 tiene el mejor rendimiento con una precisión del 98,25%, una sensibilidad del 96,55%, una especificidad del 100,00% y F1-score del 98,25%. Por otra parte, [12] comparó 13 modelos diferentes de aprendizaje profundo preentrenados (VGG-16, VGG-19, ResNet-50, ResNet-101, IncepResNetv2, MobileNetV2, InceptionV3, Xception, EfficientNet-B0, EfficientNet-B1, EfficientNet-B2, DenseNet-121 y DenseNet-169) para la detección del virus Monkeypox, los resultados del análisis mostraron que el enfoque de ensamblaje proporciona el mayor rendimiento (Precisión: 85,44%; Recall: 85,47%; F1-score: 85,40%; y Exactitud: 87,13%) durante la clasificación de la viruela del mono, mientras que el modelo Xception DL ofrece el segundo mejor rendimiento (Precisión: 85,01%; Recall: 85,14%; puntuación F1: 85,02%; y Precisión: 86,51%). De otra forma [13] utilizó un conjunto de datos personalizado con imágenes de 224x244 píxeles en formato RGB. Las imágenes se obtuvieron de medios en línea contrastados (como los CDC o la OMS) y de conjuntos de datos públicos. El conjunto de imágenes se utilizó para entrenar a los algoritmos, el 60% de las imágenes se utilizaron para el entrenamiento, el 20% para la validación y el último 20%

para las pruebas, emplearon los modelos de clasificadores VGG-16, VGG-19, ResNet50, MobileNet-V2 y EfficientNet-B0, los resultados señalaron que el mejor modelo individual fue ResNet50 con el conjunto formado por ResNet50 + EfficientNet-B0 y MobileNet-V2. De todos estos casos, el modelo individual obtuvo una precisión del 95% y el conjunto una precisión del 98,33%, mejorando los resultados de los modelos individuales en un 3,33%. Igualmente [14] recopiló 228 imágenes, 102 de viruela del mono y 126 de otros (varicela y sarampión), aplicó métodos de aumento de datos obteniendo un total de 1428 y 1764 imágenes respectivamente, utilizó 3 arquitecturas de redes neuronales, VGG16, ResNet50 e InceptionV3. Los resultados mostraron que ResNet50 tiene la mejor precisión 82.96%, VGG muestra un rendimiento competitivo con un 81.48% a comparación de InceptionV3 con un 74.07%. Por el contrario, al no existir un conjunto de datos [15] recopiló imágenes de la viruela del mono y enfermedades cutáneas, logrando obtener 164 muestras, que aplicando técnicas de aumento de datos llegó a 1915 muestras. Se utilizó el modelo VGG16 en dos estudios distintos, que consta de 16 capas CNN, se asignó el 80% de datos para el entrenamiento y el 20% para la prueba, finalmente los resultados mostraron que el primer modelo tiene una exactitud del 97% (AUC=97.2) mientras que el segundo modelo tiene 78% de exactitud (AUC=0.867). Finalmente, [16] utilizó Deep learning para diagnosticar la viruela del mono a partir de imágenes cutáneas, para los datos usaron el “Monkeypox Skin Lesion Dataset” disponible en Kaggle, Las pruebas se hicieron en 5 redes neuronales profundas (GoogLeNet, Places365-GoogLeNet, SqueezeNet, AlexNet y ResNet-18), se tuvieron en cuenta métricas de rendimiento como exactitud, precisión, recall, f1-score y AUC. Entre los modelos anteriores, ResNet18 fue capaz de obtener la mayor precisión con un 99,49%.

II. MATERIALES Y MÉTODOS

2.1. Materiales

Tabla 1. Materiales de la investigación

Ítem	Nombre	Descripción
1	Conjunto de Datos de Imágenes de la viruela del mono	Conjunto de datos que contiene imágenes de enfermedades de la viruela del mono para entrenar y evaluar los modelos [17] [18].
2	Herramientas de Software	Entorno de desarrollo de Python, bibliotecas de aprendizaje profundo (TensorFlow y Keras), ImageJ y Flask.
3	Hardware	Laptop con procesador Intel(R) Core(TM) i5-10300H CPU @ 2.50GHz con 6 núcleos, memoria RAM de 16 Gigabytes (GB) y sistema operativo Windows 10 Pro de 64 bits. Laptop con procesador Intel(R) Core(TM) i5-10300H CPU @ 2.50GHz 2.50 GHz con 4 núcleos, memoria RAM de 8 Gigabytes (GB) y sistema operativo Windows 10 Home Single Language de 64 bits.
4	Servicio	Plataforma de Google Colab para ejecutar código Python.

Nota: Listado de materiales utilizados para la investigación. Elaboración propia

2.2. Método

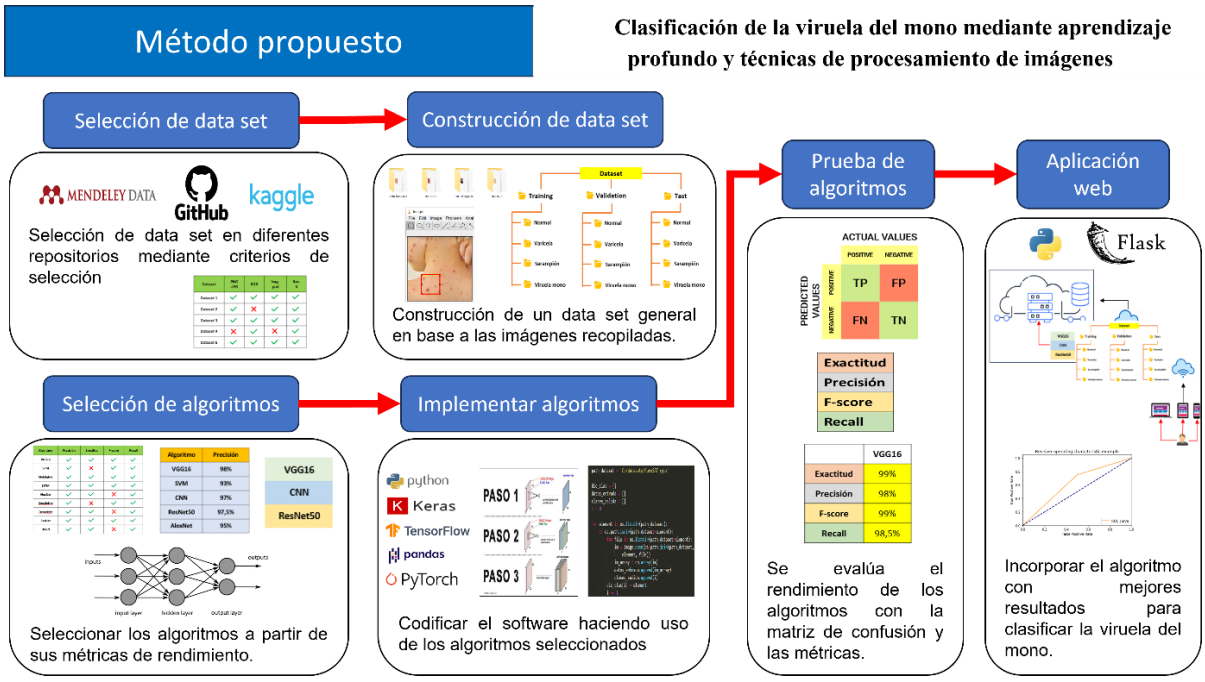


Fig. 1. Diagrama del método utilizado para la investigación. Elaboración propia

En la figura 1 se presenta el método empleado en la investigación, el cual consta de cinco etapas, en la primera etapa se seleccionaron los dataset de imágenes de la viruela del mono en distintos repositorios, para luego ser unificados en un solo dataset. En la segunda etapa se realizó la selección de los modelos a partir de una revisión sistemática de la literatura. En la tercera etapa se entrenaron los modelos utilizando el lenguaje de programación Python y a través del entorno de Google Colaboratory. En la cuarta etapa se evaluaron los resultados según las métricas. Finalmente, en la quinta etapa se desarrolló una aplicación web que tiene incorporado el algoritmo con mejor el resultado.

En primera instancia fue necesario realizar la selección del dataset para el entrenamiento de los modelos de aprendizaje profundo, para ello se realizó una búsqueda exhaustiva en diferentes fuentes como Kaggle, Mendeley Data y GitHub donde se logró identificar 13 datasets. Para la selección de los datasets se tuvieron en cuenta criterios como el formato de la imagen, debido a la necesidad de mantener la calidad de las imágenes; color RGB, para identificar características críticas de cada imagen; resolución, debido a que los modelos trabajan con tamaño de resoluciones específicas; imágenes de enfermedades

cutáneas, debido a que se necesita imágenes con información de la viruela del mono y otras enfermedades; licencia gratuita, que solo sean dataset de acceso público; y finalmente el criterio repetida, para no utilizar los mismos dataset y evitar la duplicidad de las imágenes.

A continuación, en la tabla 2 se muestra la lista de dataset con su respectivo repositorios así como también los criterios utilizados, los dataset que superaron los criterios fueron Monkeypox Skin Images Dataset (MSID) [17] y 750 Monkeypox Dataset [18].

Tabla 2. Lista de selección de datasets

#	Nombre	Criterios						
		JPG/PNG	**RGB**	**Resolución**	**Enfermedad cutánea**	**Licencia gratuita**	**Repetida**	**Seleccionado**
1	Monkeypox Skin Lesion Dataset [19]	X	X	X	X	X		No
2	Monkeypox Skin Images Dataset (MSID) [17]	X	X	X	X	X	X	Si
3	Monkeypox dataset [20]		X		X	X	X	No
4	MonkeyPox Dataset [21]	X	X	X	X	X		No
5	Monkeypox-skinimage-dataset [22]	X	X		X	X		No
6	Monkeypox Skin Image Dataset [23]	X	X	X	X	X		No
7	monkeypox 2022 remastered [24]	X	X		X	X		No
8	Monkeypox [25]	X		X	X	X	X	No

9	Mpox Skin Lesion Dataset Version 2.0 (MSLD v2.0) [26]	X	X		X	X		No
10	750 Monkeypox Dataset [18]	X	X	X	X	X	X	Si
11	monkeypox-vs-smallpox [27]	X	X		X	X	X	No
12	MonkeyPox_diplom [28]	X	X	X	X	X		No
13	monkey pox [29]	X	X	X	X	X		No

Nota: Listado de los dataset con sus respectivos criterios de selección.

Luego de que se seleccionaron y unificaron los dataset, se procedió a recortar el área de interés de las imágenes de enfermedades cutáneas a un tamaño de 64*64 pixeles (px) utilizando la herramienta ImageJ, obteniendo 399 imágenes de la enfermedad Chickenpox, 426 imágenes de Measles, 374 imágenes de Monkeypox y 378 imágenes de piel sana, tal como se puede apreciar en la figura 2.

Fig. 2. Imágenes de enfermedades cutáneas

Como resultado se obtuvo la siguiente estructura, una carpeta principal denominada “dataset64x642” y dentro de la cual se crearon las subcarpetas Chickenpox (Varicela), Measles (Sarampión), Monkeypox (Viruela del mono) y Normal (Piel sana), como se puede observar en la figura 3.

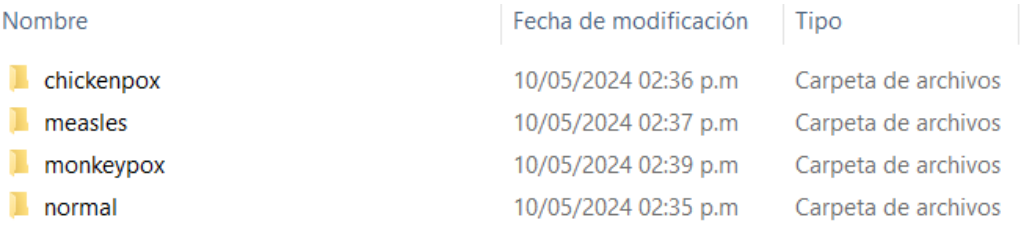


Fig. 3. Estructura del dataset

De igual forma, se realizó una selección para los modelos mediante una revisión sistemática de la literatura sobre el aprendizaje profundo aplicado a enfermedades cutáneas a través de bases de datos como ScienceDirect, Scopus y IEEExplore. Se obtuvieron un total de 80 artículos los cuales pasaron por criterios de inclusión y exclusión, quedando un total de 35 artículos para el desarrollo de la investigación (anexo 5). Luego se realizó una selección de artículos teniendo en cuenta los siguientes criterios (Accuracy, precisión, f-score, recall), obteniendo así el top 10 de los modelos con mejores resultados tal como se puede observar en la tabla 3.

Tabla 3. Lista top de los algoritmos

Modelos	Accuracy	Precision	F-score	Recall
ResNet101 [30]	99,00%	99,00%	-	99,00%
VGG19 [31] [32]	97,97%	96,94%	96,94%	96,94%
Inception-Resnet [33]	98,00%	95,00%	95,00%	95,00%
VGG16 [10] [31] [32] [34] [35]	90,27%	96,59%	94,07%	91,91%
ResNet [36][10] [37]	86,58%	96,00%	94,80%	93,63%
DenseNet [33] [37]	90,88%	92,21%	93,05%	93,92%
InceptionV3 [35] [38]	94,01%	91,29%	93,00%	88,69%
ResNet50 [11] [13] [34] [39] [40] [32]	89,90%	91,45%	93,44%	90,54%
MobileNet [31] [34] [41]	90,03%	88,96%	89,72%	96,14%
CNN [42]	84,23%	86,07%	-	81,75%

Nota: Rendimiento de los modelos de aprendizaje profundo.

En base a los criterios utilizados para la selección de los modelos, se determinó emplear VGG16, ResNet50 y una CNN personalizada para la investigación.

Para entrenar los modelos, se empleó en entorno virtual de Google Colaboratory debido a que brinda acceso gratuito a recurso de hardware como almacenamiento, memoria, GPU (Unidades de procesamiento gráfico) y CPU (Unidad Central de procesamiento) e incluso utiliza el lenguaje de programación Python.

Para el entrenamiento, validación y prueba de VGG16, ResNet50 y la CNN personalizada, cada uno inicia con la carga de la carpeta en formato “zip” que contiene las imágenes de las enfermedades cutáneas Chickenpox (Varicela), Measles (Sarampión), Monkeypox (Viruela del mono) y Normal (Piel sana) y se extrae en Google Drive.

```
# Definir la ruta de los archivos comprimidos
images_zip_path = "/content/drive/MyDrive/PAPER MONKEYPOX/dataset64x642.zip"

# Directorio donde se extraerán los archivos
extracted_path = "/content/extracted"

# Crear el directorio para extraer los archivos
os.makedirs(extracted_path, exist_ok=True)

# Extraer los archivos de imágenes
with zipfile.ZipFile(images_zip_path, 'r') as zip_ref:
    zip_ref.extractall(extracted_path)

print("Datos divididos y movidos exitosamente.")
```

```
Datos divididos y movidos exitosamente.
```

Fig. 4. Carga y extracción del dataset

Luego se realizó la división del conjunto de imágenes en grupos, de entrenamiento (train), de validación (valid) y de prueba (test) utilizando la función train_test_split, se asignó el 70% para el entrenamiento, el 15% para la validación y el 15% restante para la prueba, dichos valores se encuentran almacenadas en las variables train_percent, valid_percent y test_percent respectivamente, además se utiliza random_state = 42 con el objetivo de garantizar la consistencia al momento de ser ejecutado el código, posteriormente la división resultante de las imágenes son almacenadas en los directorios de train, valid y test.

```python
# Porcentaje para train, validación y test
train_percent = 0.7
valid_percent = 0.15
test_percent = 0.15

# Función para copiar imágenes a la nueva estructura
def copy_images(images, src_dir, dst_dir):
    os.makedirs(dst_dir, exist_ok=True)
    for img in images:
        shutil.copy(os.path.join(src_dir, img), dst_dir)

# Recorrer cada subdirectorio en el directorio de entrada
for folder in os.listdir(input_dir):
    if not os.path.isdir(os.path.join(input_dir, folder)):
        continue

    images = os.listdir(os.path.join(input_dir, folder))

    # Dividir imágenes en train, validación y test
    train_images, test_images = train_test_split(images, test_size=test_percent, random_state=42)
    train_images, valid_images = train_test_split(train_images, test_size=valid_percent/(1-test_percent), random_state=42)

    # Crear estructura de directorios para train, validación y test
    train_dir = os.path.join(output_dir, "train", folder)
    valid_dir = os.path.join(output_dir, "valid", folder)
    test_dir = os.path.join(output_dir, "test", folder)

    # Copiar imágenes a los directorios correspondientes
    copy_images(train_images, os.path.join(input_dir, folder), train_dir)
    copy_images(valid_images, os.path.join(input_dir, folder), valid_dir)
    copy_images(test_images, os.path.join(input_dir, folder), test_dir)
```

Fig. 5. División del dataset

Después de dividir el dataset, se realizó el aumento de datos para el directorio de “train” a través de la instancia de ImageDataGenerator, la cual se encarga de realizar diferentes transformaciones al conjunto de imágenes tales como reescalado, rotación, desplazamiento, entre otros con el fin de generar nuevas versiones de las mismas.

```python
# Aumento de datos
train_datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=40,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    fill_mode='nearest'
)
```

Fig. 6. Aumento de imágenes

Las Redes Neuronales son modelos que están diseñados para trabajar con datos que tengan una relación temporal y están presentando un mejor funcionamiento a problemas relacionados con imágenes. Asimismo, son utilizados usualmente para el procesamiento de imágenes, ya que es fácil detectar características de las imágenes como los borde,

tonalidades, entre otros [43]. Las CNN son utilizadas mayormente para clasificar imágenes por lo cual los datos de entrada suelen consistir en matrices de imágenes de dimensiones que contemplan el alto, ancho y la profundidad en píxeles.

El funcionamiento de una neurona artificial sigue una serie de pautas que simulan el funcionamiento de un modelo natural, a continuación se presenta su esquema [44].

u_i : representa la neurona i-ésima

y_i: representa la salida que genera la neurona i, en el caso de una neurona biológica es similar a la salida que genera el axón.

W_{ij}: representa las conexiones entre las neuronas u_j y u_i. Cuando $W_{ij} > 0$ se genera una sinapsis excitadora, $W_{ij} < 0$ se genera una sinapsis inhibidora y cuando $W_{ij} = 0$ no hay conexión.

Net_i Hace referencia a la suma de las señales que llegan a la neurona u_i. Y $f(Net)$ representa la trasferencia o la función de salida, es decir la acumulación de señales que van llegando a Net_i [44].

$$Net_i = \sum_j y_i w_{ij}$$

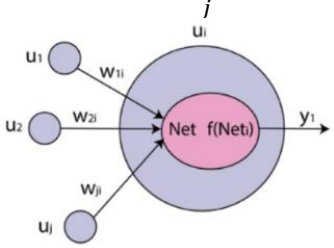


Fig. 7. Suma de señales de una neurona [44]

La convolución realizada en la CNN es una operación matemática donde se utiliza dos funciones (f y g), y produce una tercera función como resultado (s), la primera función f hace referencia a la imagen de entrada que puede ser una imagen de las enfermedades de la piel ya mencionadas, con una dimensión de 64*64 pixeles, la función g hace referencia al kernel de 3*3 y la función (s) es el mapa de características en inglés “feature map” [45], para ello se aplica la siguiente formula:

$$s(t) = (f * g)(t) \int f(T)g(t - T)dT$$

La CNN utiliza múltiples capas debido a que tiene una gran eficacia para aprender características jerárquicas a partir de datos de imágenes complejas, especialmente en el ámbito del análisis de imágenes médicas. Por ello se presenta una arquitectura de Red Neuronal Convolucional (CNN) utilizada para clasificar imágenes de Chickenpox (Varicela), Measles (Sarampión), Monkeypox (Viruela del mono) y Normal (Piel sana).

a. CNN personalizada

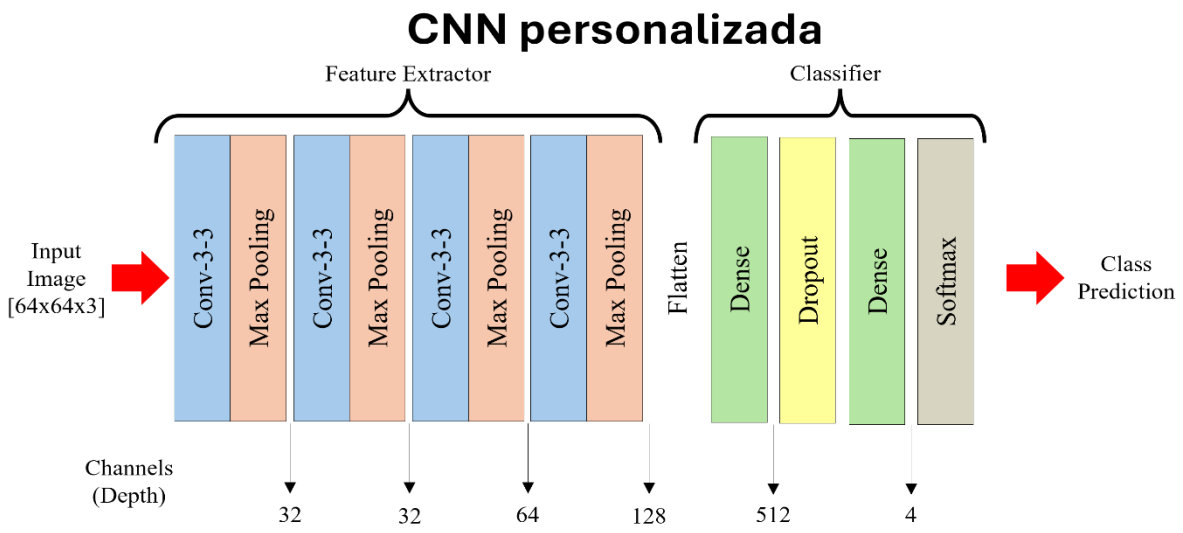


Fig. 8. Arquitectura de la CNN personalizada

Luego de realizar los pasos de carga y extracción del dataset, la división y aumento de las imágenes, se crea una CNN personalizada que consta de cuatro capas convolucionales, cada una equipada con filtros de tamaño 3x3 y activadas mediante la función ReLU para introducir no linealidad y mejorar la capacidad de detección de patrones complejos en los datos.

La activación ReLU aplica la función $f(x) = \max(0, x)$, devuelve x si x es positivo, y devuelve 0 si x en negativo, siendo x la entrada de la función [46]. Cada capa convolucional es seguida por una capa de max-pooling con un tamaño de 2x2, lo que permite reducir las dimensiones espaciales de la imagen mientras se retienen las características esenciales, generalmente es agregada para reducir el número de parámetros, es decir reduce la altura y el ancho pero mantiene la profundidad intacta a fin de que la red aprenda características.

```python
model = Sequential([
    Conv2D(32, (3, 3), activation='relu', input_shape=(64, 64, 3)),
    MaxPooling2D((2, 2)),
    Conv2D(32, (3, 3), activation='relu'),
    MaxPooling2D((2, 2)),
    Conv2D(64, (3, 3), activation='relu'),
    MaxPooling2D((2, 2)),
    Conv2D(128, (3, 3), activation='relu'),
    MaxPooling2D((2, 2)),
    Flatten(),
    Dense(512, activation='relu'),
    Dropout(0.5),
    Dense(4, activation='softmax')  # 4 clases de salida
])

model.compile(optimizer='adam',
              loss='categorical_crossentropy',
              metrics=['accuracy'])

history = model.fit(
    train_generator,
    epochs=100,
    validation_data=valid_generator
)
```

Fig. 9. Entrenamiento del modelo CNN personalizado

Posteriormente, se incluyen dos capas completamente conectadas. La primera de estas capas utiliza la función de activación ReLU para mantener la no linealidad, mientras que la segunda capa emplea la función de activación softmax, con un parámetro ajustable que en este caso contiene cuatro neuronas, la cual representan las predicciones de salida, como: Monkeypox (viruela del mono), Normal (piel normal), Chickenpox (varicela) y Measles (sarampión), el modelo CNN personalizado fue entrenado durante 100 épocas utilizando el conjunto de datos de entrenamiento y validación.

Esta arquitectura está diseñada para extraer características profundas y relevantes de las imágenes, facilitando una clasificación precisa, para ello se calculan la métricas de evaluación, como exactitud, precesión, recall y f1-score para evaluar el modelo.

```python
# Evaluar el modelo
test_steps = test_generator.samples // 8
test_loss, test_acc = model.evaluate(test_generator, steps=test_steps)
print(f"Test Accuracy: {test_acc}")

# Predicciones y evaluación detallada
test_generator.reset()
Y_pred = model.predict(test_generator, steps=test_steps)
y_pred = np.argmax(Y_pred, axis=1)
y_true = test_generator.classes[:len(y_pred)]

print('Classification Report')
target_names = list(test_generator.class_indices.keys())
print(classification_report(y_true, y_pred, target_names=target_names))
```

Fig. 10. Métricas de evaluación para la CNN personalizada

b. VGG16

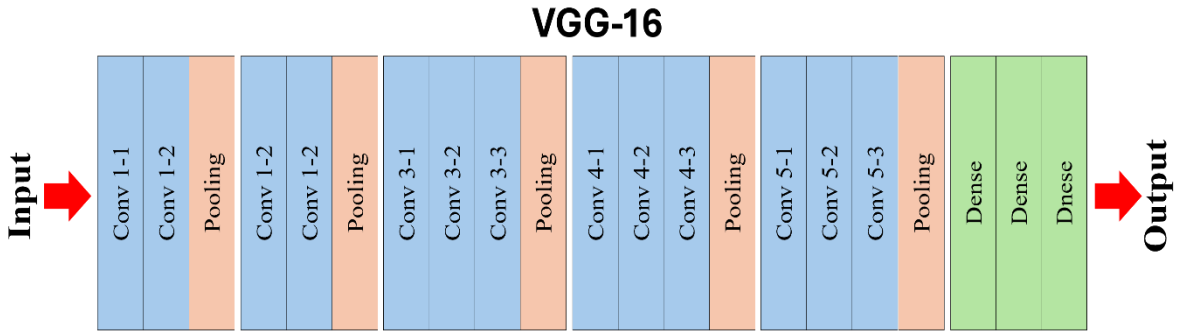


Fig. 11. Arquitectura de la VGG16

VGG16 es una red neuronal convolucional propuesta por la Universidad de Oxford. Su nombre VGG16 hace referencia a que tiene 16 capas con peso es decir con parámetros aprendibles, de los cuales 13 capas son convolucionales y 3 son capas densas o totalmente conectadas [47].

Luego de realizar los pasos de carga y extracción del dataset, la división y aumento de las imágenes, se carga el modelo VGG16 utilizando los pesos de Imagenet pero sin incluir las capas totalmente conectada debido a que include_top está en False. Seguido se ajustan las capas para que sus pesos no cambien durante el entrenamiento.

Posteriormente, al igual que el modelo CNN se incluyeron dos capas completamente conectadas, la primera fue una función ReLu y la segunda una función softmax con salida de cuatro neuronas haciendo referencia al número de las clases Monkeypox (viruela del mono),

Normal (piel normal), Chickenpox (varicela) y Measles (sarampión). Paso siguiente se creó el modelo y se definió tanto el input (imágenes) como el output (capa de salida). Finalmente se compiló y entrenó el modelo durante 100 épocas utilizando el conjunto de datos de entrenamiento y validación.

```
base_model = VGG16(weights='imagenet', include_top=False, input_shape=(64, 64, 3))

for layer in base_model.layers:
    layer.trainable = False

x = base_model.output
x = Flatten()(x)
x = Dense(512, activation='relu')(x)
x = Dropout(0.5)(x)
predictions = Dense(4, activation='softmax')(x)  # 4 clases

model = Model(inputs=base_model.input, outputs=predictions)

# Compilar el modelo
model.compile(optimizer=Adam(learning_rate=0.0001), loss='categorical_crossentropy', metrics=['accuracy'])

# Entrenar el modelo
epochs = 100

history = model.fit(
    train_generator,
    steps_per_epoch=train_generator.samples // batch_size,
    validation_data=valid_generator,
    validation_steps=valid_generator.samples // batch_size,
    epochs=epochs
)
```

Fig. 12. Entrenamiento del modelo VGG16

La evaluación del modelo se realiza haciendo uso del conjunto de datos de prueba, donde se calculan las métricas de evaluación como exactitud, precesión, recall y f1-score para cada una de las clases existentes.

```
# Evaluar el modelo
test_steps = test_generator.samples // 8
test_loss, test_acc = model.evaluate(test_generator, steps=test_steps)
print(f"Test Accuracy: {test_acc}")

# Predicciones y evaluación detallada
test_generator.reset()
Y_pred = model.predict(test_generator, steps=test_steps)
y_pred = np.argmax(Y_pred, axis=1)
y_true = test_generator.classes[:len(y_pred)]

print('Classification Report')
target_names = list(test_generator.class_indices.keys())
print(classification_report(y_true, y_pred, target_names=target_names))
```

Fig. 13. Métricas de evaluación para la VGG16

c. RESNET50

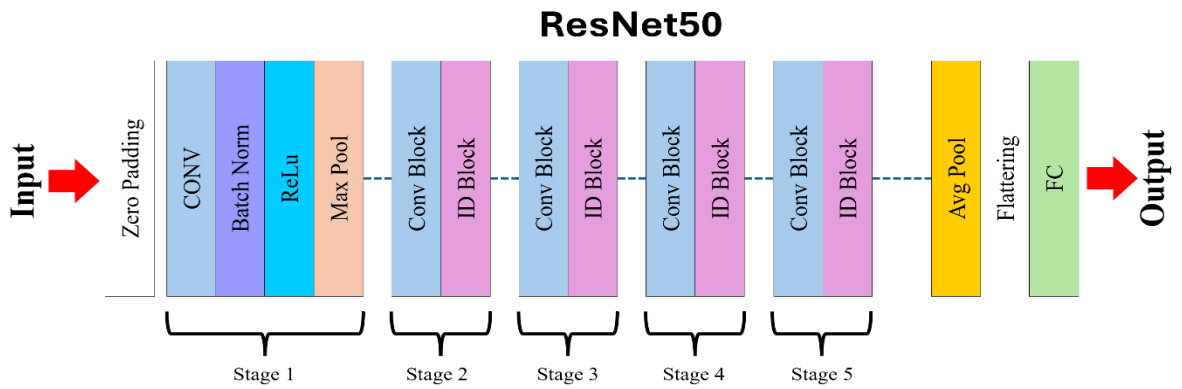


Fig. 14. Arquitectura de la ResNet50

ResNet50 es una red neuronal convolucional desarrollador por Microsoft Research. Su nombre ResNet50 hace referencia a que tiene 50 capas de profundidad [n], de los cuales 48 son capas convolucionales, tiene un max-pooling y una capa de Average Pooling [48].

Luego de realizar los pasos de carga y extracción del dataset, la división y aumento de las imágenes, se carga el modelo ResNet50 utilizando los pesos de Imagenet pero sin incluir las capas totalmente conectada debido a que include_top está en False, el modelo a diferencia de la VGG16 utiliza las 10 ultimas capas como entrenables para que se puedan ajustar durante el entrenamiento.

Posteriormente, al crear el modelo se incluyeron dos capas completamente conectadas con las mismas funciones al igual que el modelo CNN y VGG16. También se añadió la capa GlobalAveragePooling2D para disminuir el tamaño de las dimensiones. Paso siguiente se compiló y entrenó el modelo durante 100 épocas utilizando el conjunto de datos de entrenamiento y validación.

```python
# Construir el modelo
base_model = ResNet50(weights='imagenet', include_top=False, input_shape=(64, 64, 3))

for layer in base_model.layers[-10:]:
    layer.trainable = True

model = Sequential([
    base_model,
    GlobalAveragePooling2D(),
    Dropout(0.5),
    Dense(1024, activation='relu'),
    Dropout(0.5),
    Dense(4, activation='softmax')
])

# Compilar el modelo
model.compile(optimizer=Adam(learning_rate=0.0001),
              loss='categorical_crossentropy',
              metrics=['accuracy'])

# Entrenar el modelo
history = model.fit(
    train_generator,
    steps_per_epoch=train_generator.samples // 8,
    validation_data=valid_generator,
    validation_steps=valid_generator.samples // 8,
    epochs=100
)
```

Fig. 15. Entrenamiento del modelo ResNet50

Por último, se establecieron las métricas como exactitud, precisión, recall y f1-score que permiten indicar el desempeño del modelo haciendo uso del conjunto de datos de prueba. Es relevante mencionar que el test_generator se reinicia después de haber calculado la exactitud con el fin de realizar las predicciones y obtener la clase predicha por el modelo.

```python
# Evaluar el modelo
test_steps = test_generator.samples // 8
test_loss, test_acc = model.evaluate(test_generator, steps=test_steps)
print(f"Test Accuracy: {test_acc}")

# Predicciones y evaluación detallada
test_generator.reset()
Y_pred = model.predict(test_generator, steps=test_steps)
y_pred = np.argmax(Y_pred, axis=1)
y_true = test_generator.classes[:len(y_pred)]

print('Classification Report')
target_names = list(test_generator.class_indices.keys())
print(classification_report(y_true, y_pred, target_names=target_names))
```

Fig. 16. Métricas de evaluación para la ResNet50

De acuerdo al trabajo de investigación, donde se aplica el aprendizaje profundo como son los modelos para clasificar la viruela del mono. En la tabla 4, se presenta las siglas que son utilizadas en la matriz de confusión, donde se utilizan fórmulas de cada indicador para medir el rendimiento de los modelos.

Tabla 4. Término de la matriz de confusión

Término de la matriz de confusión	
Término	Sigla
Verdaderos Positivos	VP
Verdaderos Negativos	VN
Falsos Positivos	FP
Falsos Negativos	FN

Exactitud

Evalúa el resultado global de la exactitud y la clasificación exacta.

$$E = \frac{VP + VN}{VP + FP + FN + VN}$$

Precisión

Evalúa la calidad de respuestas positivas que generan los modelos.

$$P = \frac{VP}{VP + FP}$$

Recall

Evalúa la eficiencia de clasificación de todos los elementos que pertenecen a una misma clase.

$$R = \frac{VP}{VP + FN}$$

F1-score

Es una métrica que evalúa el rendimiento, mediante la combinación de recall y precisión, para ello proporciona una medida que varía entre 0 y 1.

$$F = 2 * \frac{(P * R)}{(P + R)}$$

Finalmente como último paso de la investigación, se implementó una aplicación web que permita que cualquier usuario pueda cargar una imagen a través de la aplicación y esta le informará si es la viruela del mono, la varicela, el sarampión o piel sana.

En la figura 17 se muestra la arquitectura de la aplicación web desarrollada. Se ha trabajado en una interfaz de usuario para cargar la imagen y seleccionar el área de interés, luego esta área recortada se envía a la API web donde se encuentra el modelo ResNet50.

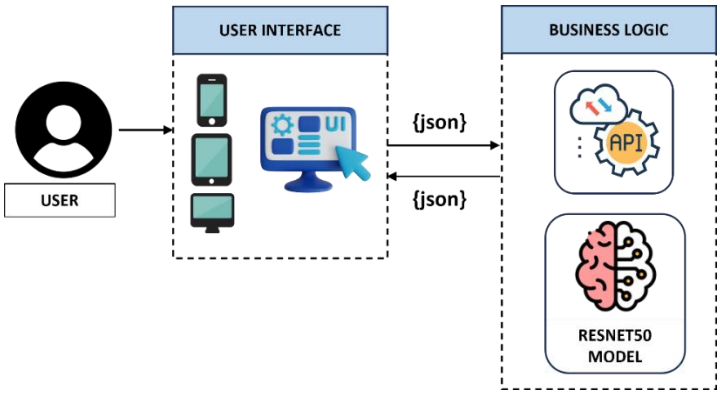


Fig. 17. Arquitectura de la aplicación web. Elaboración propia

Posteriormente se programó la aplicación web utilizando el entorno de Visual Studio Code con en el lenguaje de programación Python y el framework Flask para la creación de las interfaces web, facilitando la interacción fluida entre los usuarios y el modelo de aprendizaje profundo.

En la figura 18 se muestra la interfaz de inicio de la aplicación web donde el usuario puede seleccionar una de las opciones para clasificar una imagen.

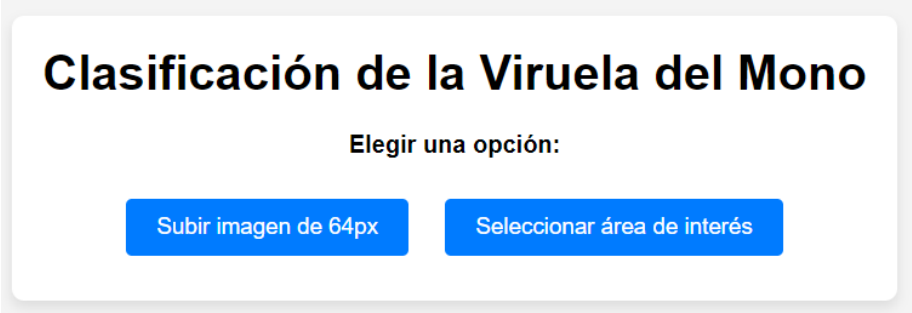


Fig. 18. Interfaz de inicio de la aplicación web

De acuerdo a la opción seleccionada los usuarios pueden cargar imágenes desde sus dispositivos ya sea de un tamaño de 64*64 pixeles o seleccionar áreas específicas de interés en la lesión cutánea que desean analizar.

Fig. 19. Secciones que contiene la aplicación web

Posteriormente, la aplicación envía esta región seleccionada al modelo de aprendizaje profundo previamente entrenado. Este modelo realiza un análisis detallado de la imagen para determinar la presencia y el tipo de enfermedad dermatológica. Una vez procesada la imagen, el modelo devuelve un resultado de clasificación que identifica la enfermedad detectada junto con un porcentaje de confianza asociado. Este resultado se presenta de manera visual en la interfaz de usuario, permitiendo a los usuarios interpretar fácilmente la información proporcionada y tomar decisiones informadas sobre su salud cutánea.

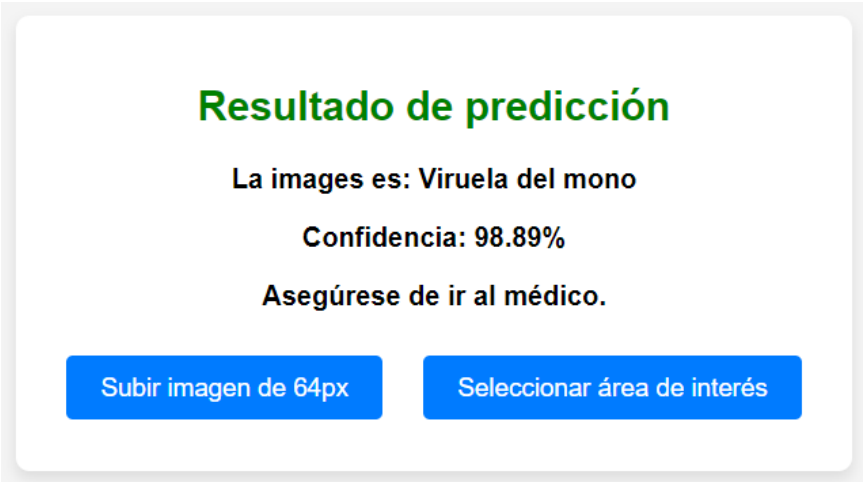


Fig. 20. Interfaz de resultado de la aplicación web

Esta integración tecnológica no solo mejora la accesibilidad a diagnósticos dermatológicos precisos, sino que también ofrece un recurso educativo valioso para pacientes y profesionales de la salud interesados en la dermatología digital y la inteligencia artificial aplicada a la medicina.

III. RESULTADOS Y DISCUSIÓN

3.1. Resultados

Tabla 5. Medidas de consumo de CPU, memoria RAM y tiempo de respuesta empleado para los modelos de aprendizaje profundo

Modelo empleado	Consumo de CPU	Consumo de RAM	Tiempo de Respuesta
CNN personalizada	57.20%	1821.24 MB	413.93 s
VGG16	68.70%	2181.82 MB	442.63 s
ResNet50	67.20%	3589.43 MB	913.25 s

Nota: Medidas de consumo de recursos de CPU, RAM y tiempo de respuesta utilizando los ordenadores antes mencionados. Fuente: Elaboración propia.

La tabla 5 muestra el consumo de CPU, el consumo de RAM y el tiempo de respuesta que fue utilizado para cada uno de los clasificadores de aprendizaje profundo, Es importante señalar que las diferencias en el uso de CPU, memoria RAM y tiempos de respuesta se deben a la complejidad de las arquitecturas.

El primer modelo es la CNN personalizada, utilizó el 57.20% del CPU, consumió 1821.24 MB de RAM y tuvo el menor tiempo de respuesta con 413 segundos.

El segundo modelo es VGG16 el cual utilizó el 68.70% del CPU, consumió 2181.82 MB de RAM y un tiempo de respuesta de 442 segundos.

El tercer modelo ResNet50 empleó el 67.20% del CPU, consumió 3589.43 MB de RAM y tuvo el mayor tiempo de respuesta de 913 segundos.

A continuación, se presentan los resultados que se obtuvieron, teniendo en cuenta las métricas de rendimiento para cada modelo al clasificar las imágenes en clases específicas.

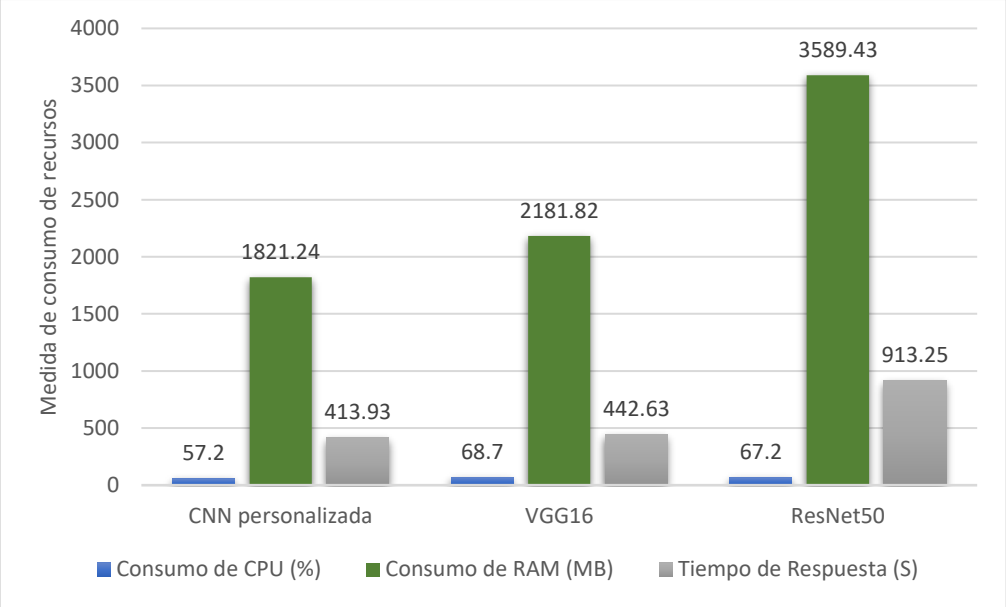


Fig. 21. Medidas de consumo de CPU, memoria RAM y tiempo de respuesta empleado para los modelos de aprendizaje profundo

CNN personalizada

El primer modelo evaluado es la CNN personalizada. A continuación, se detallan en la tabla 6 los resultados obtenidos por este modelo.

Tabla 6. Medidas de rendimiento del modelo CNN personalizado

Clase	Exactitud	Precisión	Recall	F1-score
Chickenpox	93.00%	90.00%	95.00%	93.00%
Measles	93.00%	86.00%	94.00%	90.00%
Monkeypox	93.00%	98.00%	88.00%	93.00%
Normal	93.00%	100.00%	94.00%	97.00%

Nota: Medidas de rendimiento del modelo CNN personalizado, los cálculos se obtienen utilizando el mismo conjunto de datos tanto para el entrenamiento, prueba y validación.

El modelo CNN personalizado tuvo un mejor rendimiento en comparación del VGG16 sin embargo no supera los resultados obtenidos por el modelo ResNet50 en la clasificación de las cuatro categorías: Chickenpox, Measles, Monkeypox y Normal, obtuvo una exactitud general del 93.00%.

Para la identificación de casos de chickenpox (varicela) el modelo tuvo una precisión del 90.00%, el recall fue del 95.00% y el F1-score del 93.00% lo que refleja un buen desempeño en la clasificación del chickenpox.

Para los casos de measles (sarampión) el modelo mostró una precisión del 86.00%, el recall fue del 94.00% y el F1-score del 90.00%, aunque muestra un buen rendimiento en la clasificación de measles existe un margen de mejora en la precisión.

Para los casos de monkeypox (viruela del mono) el modelo obtuvo una precisión alta del 98.00% al igual que el modelo ResNet50, sin embargo, el recall fue menor ya que obtuvo un 88.00% y un F1-score del 93.00%, lo que sugiere que, aunque el modelo sea muy preciso en las predicciones aún puede mejorarse en la identificación de casos reales de monkeypox.

Para los casos de imágenes normales el modelo obtuvo una precisión del 100.00%, el recall del 94.00% y F1-score del 97.00%, lo que refleja que el modelo tiene una buena capacidad para clasificar imágenes normales.

VGG16

El segundo modelo evaluado es el VGG16. Los resultados obtenidos por este modelo están detallados en la tabla 7.

Tabla 7. Medidas de rendimiento del modelo VGG16

Clase	Exactitud	Precisión	Recall	F1-score
Chickenpox	88.36%	81.82%	90.00%	85.74%
Measles	88.36%	84.51%	93.75%	88.91%
Monkeypox	88.36%	91.67%	77.19%	83.78%
Normal	88.36%	100.00%	92.16%	95.89%

Nota: Medidas de rendimiento del modelo VGG16, los cálculos se obtienen utilizando el mismo conjunto de datos tanto para el entrenamiento, prueba y validación.

El modelo VGG16 demostró un rendimiento notable en la clasificación de las cuatro categorías de enfermedades: Chickenpox, Measles, Monkeypox y Normal, con una exactitud general del 88.36%.

Para la identificación de casos de chickenpox (varicela) el modelo alcanzó una precisión del 81.82% lo que significa que ese mismo porcentaje de imágenes de Chickenpox se clasificaron correctamente, el recall fue del 90.00% indicando una alta capacidad para identificar casos reales de chickenpox y un F1-score del 85.74%.

Para los casos de measles (sarampión) el modelo tuvo una precisión alta del 84.51%, el recall obtenido fue del 93.75% y el F1-score tuvo 88.91% reflejando la efectividad del modelo en la clasificación de measles.

Para los casos de monkeypox (viruela del mono) el modelo logró una precisión del 91.67%, un recall del 77.19% y un F1-score del 83.78% lo que indica una buena capacidad del modelo para clasificar los casos de monkeypox, sin embargo, el F1-score indica que hay un margen de mejora en la identificación de los casos reales.

Para la detección de imágenes normales el modelo logró clasificar todas las imágenes clasificadas como “normal” de forma correcta con una precisión del 100.00%, el recall fue del 92.16% y el F1-score tuvo 95.89%, mostrando una alta capacidad de clasificación para las imágenes normales.

ResNet50

El tercer y último modelo evaluado es el ResNet50. A continuación se detallan en la tabla 8 los resultados obtenidos por este modelo.

Tabla 8. Medidas de rendimiento del modelo ResNet50

Clase	Exactitud	Precisión	Recall	F1-score
Chickenpox	98.00%	100.00%	93.00%	97.00%
Measles	98.00%	96.00%	100.00%	98.00%
Monkeypox	98.00%	98.00%	100.00%	99.00%
Normal	98.00%	100.00%	100.00%	100.00%

Nota: Medidas de rendimiento del modelo ResNet50, los cálculos se obtienen utilizando el mismo conjunto de datos tanto para el entrenamiento, prueba y validación.

El modelo ResNet50 tuvo un rendimiento excelente en la clasificación de las cuatro

categorías: Chickenpox, Measles, Monkeypox y Normal, con una exactitud general del 98.00%.

Para la identificación de casos de chickenpox (varicela) el modelo logró una precisión del 100.00% lo que indica que todas las imágenes clasificadas como Chickenpox fueron correctas, el recall fue del 93.00% y el F1-score alcanzó un 97.00% lo que significa que hay un balance excelente entre precisión y recall.

Para los casos de measles (sarampión) el modelo mostró una precisión del 96.00%, el recall fue del 100.00% lo que indica que el modelo logró identificar todos los casos reales de measles y el F1-score del 98.00%.

Para los casos de monkeypox (viruela del mono) el modelo obtuvo una precisión del 98.00%, es decir que 98.00% de las imágenes fueron correctamente clasificadas como monkeypox, el recall fue del 100.00% y el F1-score del 99.00%, lo que demuestra que el modelo tiene una gran capacidad de clasificar las imágenes de monkeypox de forma efectiva.

Para los casos de imágenes normales el modelo obtuvo una precisión, recall y F1-score del 100%, es decir que todas las imágenes clasificadas como “normal” fueron correctas al igual que los casos de imágenes reales.

3.2. Discusión

Los hallazgos derivados del presente estudio enfocado en la evaluación de modelos de aprendizaje profundo para la clasificación de la viruela del mono proporcionan información fundamental para el avance de herramientas diagnósticas en el ámbito clínico. La implementación de modelos de aprendizaje profundo demostró un desempeño alentador en lo que respecta a la exactitud, precisión, recall y f1-score en la detección de la viruela del mono.

El análisis reveló que el modelo clasificador ResNet50 superó a otros enfoques, evidenciado por una exactitud del 98.00%, la precisión del 98.50%, recall del 98.25% y f1-score del 98.50%. Estos resultados demuestran la gran capacidad de ResNet50 para poder distinguir con precisión los casos de enfermedades de la viruela del mono, lo cual significaría una ayuda importante en el diagnóstico de esta enfermedad.

Sin embargo, es necesario abordar las situaciones encontradas a lo largo del estudio, el conjunto de datos presenta limitaciones de imágenes médicas para la viruela del mono y otras enfermedades cutáneas, lo que podría dificultar la creación de un conjunto de datos robusto para el entrenamiento de los modelos. La variabilidad de las imágenes y diversidad de las lesiones son factores a considerar para mejorar la confiabilidad del dataset.

Al realizar la comparación de los resultados con investigaciones previas, como la de [11] empleó un dataset con un conjunto de imágenes para viruela del mono y otro para piel sana con un total de 572 imágenes, las cuales fueron redimensionadas a 224x224 píxeles, entre los modelos que entrenó se encontraron los de ResNet-18, ResNet-50, VGG-16, Densenet-161, EfficientNet B7, EfficientNet V2, GoogLeNet, MobileNet V2, MobileNet V3, ResNeXt-50, ShuffleNet V2, y ConvNeXt. Durante la fase de entrenamiento de clasificación de imágenes emplearon la entropía cruzada, un batch size de 32 y 50 épocas. Los resultados que obtuvieron con el modelo ResNet50 fueron 96.49% de exactitud, 93.10% de recall y 96.43% de f1-score; siendo estos resultados menores a los que se obtuvieron en la presente investigación. Por otro lado, [13] empleó un conjunto de datos personalizados con imágenes de la viruela del mono, piel sana y otras enfermedades de 224x224 píxeles, dicho conjunto de datos fue dividido en 60% para entrenamiento, 20% para validación y 20% para las pruebas; los clasificadores empleados fueron VGG-16, VGG-19, ResNet50, MobileNet-V2 y EfficientNet-B0, su modelo con mejores resultados fue el ResNet50 con una exactitud, precisión, recall y f1-score del 95.00%, no pudiendo superar los resultados obtenidos en el presente estudio. Por otro lado, el modelo CNN personalizado no superó a ResNet50, ya que al ser una red más profunda y con una arquitectura de 50 capas que incluye bloques residuales y al utilizar los pesos preentrenados de ImageNet permite aprender y generalizar patrones complejos.

Los resultados prácticos del estudio sugieren que los modelos de aprendizaje profundo tienen un gran potencial para mejorar la clasificación de la viruela del mono. No obstante, aún existen desafíos para llevar los modelos a la aplicación clínica, especialmente en la adaptación a entornos reales debido a las variaciones en la calidad de las imágenes y

las condiciones de captura.

El análisis presentado respalda la capacidad significativa de la utilización de modelos de aprendizaje profundo en la clasificación de la viruela del mono, resaltando la necesidad de que futuras investigaciones puedan abordar las limitaciones y desafíos actuales, con el fin de facilitar la implementación de estas herramientas en la práctica clínica.

IV. CONCLUSIONES Y RECOMENDACIONES

4.1. Conclusiones

- La selección de los dataset de licencia pública identificados en distintas fuentes como Kaggle, Mendeley Data y GitHub, permitió la construcción de un dataset que incluye imágenes de la viruela del mono y enfermedades cutáneas la cual ha sido esencial para lograr el entrenamiento, validación y prueba de los modelos de aprendizaje profundo.
- Al realizar una revisión sistemática de la literatura se logró identificar los diez modelos de aprendizaje profundo con mejor desempeño para la clasificación de enfermedades cutáneas, de los cuales se seleccionaron CNN, VGG16 y ResNet50 para la elaboración de la investigación teniendo en cuenta los criterios de exactitud, precisión, recall, F1-score.
- La implementación de los modelos seleccionados se llevó a cabo utilizando el lenguaje de programación Python en el entorno virtual de Google Colaboratory. Para el entrenamiento de los modelos, el dataset de dividió en tres partes, el 70% fue destinado para el entrenamiento al cual se le realizó aumento de datos, 15% para la validación y el 15% restante para las pruebas, lo que contribuyó a obtener resultados destacables.
- Se evaluó el desempeño de los modelos de aprendizaje profundo para la clasificación de la viruela del mono y otras enfermedades, donde ResNet50 obtuvo los resultados más sobresalientes con una exactitud del 98.00%, precisión del 98.50%, recall del 98.25% y f1-score del 98.50%.
- Se desarrolló una aplicación web funcional que tiene incorporado el modelo

ResNet50, dicha aplicación permite subir imágenes de 64*64px o seleccionar el área de interés de una imagen para que el modelo pueda clasificarlo y proporcionar información sobre la condición cutánea identificada.

4.2. Recomendaciones

- Se recomienda utilizar una técnica de aumento de datos cuando existen problemas con los conjuntos de datos limitados como es el caso de la enfermedad de la viruela del mono. De esta forma, se crearán variaciones adicionales en los datos de entrenamiento, lo que mejorará el rendimiento y generalización de los modelos de aprendizaje profundo.

- Las bibliotecas que ofrece Python facilitan el desarrollo e implementación de los modelos de aprendizaje profundo, se sugiere utilizar bibliotecas especializadas en la creación de modelos o procesamiento de imágenes como TensorFlow o Keras; debido a que facilitan el desarrollo de los modelos y mejoran el rendimiento general.

REFERENCIAS

[1] L. Ochoa, “Cómo evitar confundir la Viruela del mono con la Sífilis, el Herpes o la Varicela.” Accessed: Jul. 23, 2023. [Online]. Available: https://medicinaysaludpublica.com/noticias/infectologia/como-evitar-confundir-la-viruela-del-mono-con-la-sifilis-el-herpes-o-la-varicela/15267

[2] BBC, “Viruela del mono: por qué no se ha podido frenar la expansión de la enfermedad (y qué riesgos tiene que no se haya declarado una emergencia).” Accessed: Jul. 23, 2023. [Online]. Available: https://www.bbc.com/mundo/noticias-62172415

[3] Sociedad Española de Infectología Pediátrica, “Viruela del mono (Monkeypox).” Accessed: Jul. 23, 2023. [Online]. Available: https://www.seipweb.es/wp-content/uploads/2022/06/Documento-SEIP-Viruela-mono-v.1.0.-docx.pdf

[4] Organización Mundial de la Salud, “Viruela símica.” Accessed: Jul. 23, 2023. [Online]. Available: https://www.who.int/es/news-room/questions-and-answers/item/monkeypox

[5] CDC, “Mapa mundial del brote de viruela símica o del mono del 2022. Obtenido de Centros para el Control y la Prevención de Enfermedades.” Accessed: Jul. 23, 2023. [Online]. Available: https://espanol.cdc.gov/poxvirus/monkeypox/response/2022/world-map.html

[6] M. de Salud, “SALA SITUACIONAL DE LA MPOX (VIRUELA SÍMICA).” [Online]. Available: https://www.dge.gob.pe/sala-monkeypox/

[7] Agencia de Seguridad Sanitaria de Reino Unido, “Monkeypox: background information.” Accessed: Jul. 23, 2023. [Online]. Available: https://www.gov.uk/guidance/monkeypox

[8] Andina, “Viruela del mono: ¿qué pruebas se hacen para detectar esta enfermedad?” [Online]. Available: https://andina.pe/agencia/noticia-viruela-del-mono-pruebas-se-hacen-para-detectar-esta-enfermedad-897410.aspx

[9] Kumar. Vidit, “Analysis of CNN features with multiple machine learning classifiers in diagnosis of monkepox from digital skin images,” 2022.

[10] Md Shahin A., Md Sipon M., Jahurul Haque, Md Mahbubur R., and Md Khairul I., “An

enhanced technique of skin cancer classification using deep convolutional neural network with transfer learning models," *Machine Learning with Applications*, vol. 5, 2021.

[11] K. D. Akın, C. Gurkan, A. Budak, and H. Karataş, "Classification of Monkeypox Skin Lesion using the Explainable Artificial Intelligence Assisted Convolutional Neural Networks," *European Journal of Science and Technology*, 2022.

[12] Chiranjibi Sitaula and Tej Bahadur Shahi, "Monkeypox virus detection using pre-trained Deep learning-based approaches," *J Med Syst*, 2022.

[13] L. Muñoz Saavedra, E. Escobar Linero, J. Civit Masot, F. Luna Perejón, Civit Antón, and M. Domínguez Morales, "Monkeypox Diagnostic-Aid System with Skin Images Using Convolutional Neural Networks," 2022.

[14] N. A. Shams *et al.*, "Monkeypox skin lesion detection using deep learning models: A feasibility study," *Ithaca: Cornell University Library*, 2022.

[15] M. M. Ahsan, M. R. Uddin, M. Farjana, A. N. Sakib, K. A. Momin, and A. L. Shahana, "Image data collection and implementation of deep learning-based model in detecting monkeypox disease using modified VGG16," Cornell University Library. Accessed: Jul. 23, 2023. [Online]. Available: https://www.proquest.com/working-papers/image-data-collection-implementation-deep/docview/2673709946/se-2

[16] T. Nayak *et al.*, "Deep learning based detection of monkeypox virus using skin lesion images," *Med Nov Technol Devices*, vol. 18, no. May, p. 100243, 2023, doi: 10.1016/j.medntd.2023.100243.

[17] D. Bala, "Monkeypox Skin Images Dataset (MSID)." [Online]. Available: https://www.kaggle.com/datasets/dipuiucse/monkeypoxskinimagedataset

[18] B. PRATOMO, "750 Monkeypox Dataset." [Online]. Available: https://www.kaggle.com/datasets/bintangpratomo/750-monkeypox-dataset

[19] J. Paul, M. T. Ahmed, and T. J. Peana, "Monkeypox Skin Lesion Dataset." [Online]. Available: https://www.kaggle.com/datasets/nafin59/monkeypox-skin-lesion-dataset

[20] J. DABASS, "Monkeypox dataset." [Online]. Available:

https://www.kaggle.com/datasets/jyotidabas/monkeypox-dataset

[21] P. GUPTA, “MonkeyPox Dataset.” [Online]. Available: https://www.kaggle.com/datasets/piyush19/monkeypox-datasetDataset

[22] J. DABASS, “Monkeypox-skinimage-dataset,” 2022.

[23] A. M. Bakhiet, “Monkeypox Skin Image Dataset.” [Online]. Available: https://www.kaggle.com/datasets/alimohammedbakhiet/monkeypox-skin-image-dataset

[24] M. Melichov, “monkeypox 2022 remastered.” [Online]. Available: https://www.kaggle.com/datasets/maxmelichov/monkeypox-2022-remastered

[25] S. Parvez, “monkeypox.” [Online]. Available: https://www.kaggle.com/datasets/shadanparvez/monkeypox

[26] T. J. P. Joydip Paul, Md Tazuddin Ahmed, “Mpox Skin Lesion Dataset Version 2.0 (MSLD v2.0).” [Online]. Available: https://www.kaggle.com/datasets/joydippaul/mpox-skin-lesion-dataset-version-20-msld-v20

[27] S. Thamke, “monkeypox-vs-smallpox.” [Online]. Available: https://www.kaggle.com/datasets/soumyathamke/monkeypoxvssmallpox/data

[28] M. Protsenko, “MonkeyPox_diplom.” [Online]. Available: https://www.kaggle.com/datasets/marynaprotsenko/monkeypox-diplom

[29] S. Mallik, “monkey fox.” [Online]. Available: https://www.kaggle.com/datasets/sheshabhusanmallik/monkey-fox

[30] M. M. Ahsan *et al.*, “Deep transfer learning approaches for Monkeypox disease diagnosis,” *Expert Syst Appl*, vol. 216, p. 119483, 2023, doi: https://doi.org/10.1016/j.eswa.2022.119483.

[31] H. M. Balaha and A. E.-S. Hassan, “Skin cancer diagnosis based on deep transfer learning and sparrow search algorithm,” *Neural Comput Appl*, vol. 35, no. 1, pp. 815 – 853, 2023, doi: 10.1007/s00521-022-07762-9.

[32] S. Hosseinzadeh Kassani and P. Hosseinzadeh Kassani, “A comparative study of deep learning architectures on melanoma detection,” *Tissue Cell*, vol. 58, pp. 76–83, 2019,

doi: https://doi.org/10.1016/j.tice.2019.04.009.

[33] R. Sadik, A. Majumder, A. A. Biswas, B. Ahammad, and Md. M. Rahman, “An in-depth analysis of Convolutional Neural Network architectures with transfer learning for skin disease diagnosis,” *Healthcare Analytics*, vol. 3, p. 100143, 2023, doi: https://doi.org/10.1016/j.health.2023.100143.

[34] M. Michalska-Ciekańska, “NEURAL NETWORKS FROM KERAS IN SKIN LESION DIAGNOSTIC; [SIECI NEURONOWE Z KERAS W DIAGNOSTYCE ZMIAN SKÓRNYCH],” *Informatyka, Automatyka, Pomiary w Gospodarce i Ochronie Srodowiska*, vol. 12, no. 1, pp. 40 – 43, 2022, doi: 10.35784/iapgos.2876.

[35] D. Bala *et al.*, “MonkeyNet: A robust deep convolutional neural network for monkeypox disease detection and classification,” *Neural Networks*, vol. 161, pp. 757–775, 2023, doi: https://doi.org/10.1016/j.neunet.2023.02.022.

[36] P. Priyanti and K. Ahasan, “An artificial neural network based detection and classification of melanoma skin cancer using hybrid texture features,” *Sensors International*, 2021.

[37] F. Bagheri, M. J. Tarokh, and M. Ziaratban, “Skin lesion segmentation from dermoscopic images by using Mask R-CNN, Retina-Deeplab, and graph-based methods,” *Biomed Signal Process Control*, vol. 67, p. 102533, 2021, doi: https://doi.org/10.1016/j.bspc.2021.102533.

[38] M. S. Junayed, A. N. M. Sakib, N. Anjum, M. B. Islam, and A. A. Jeny, “EczemaNet: A Deep CNN-based Eczema Diseases Classification,” in *2020 IEEE 4th International Conference on Image Processing, Applications and Systems (IPAS)*, 2020, pp. 174–179. doi: 10.1109/IPAS50080.2020.9334929.

[39] D. Popescu, M. El-Khatib, and L. Ichim, “Skin Lesion Classification Using Collective Intelligence of Multiple Neural Networks,” *Sensors*, vol. 22, no. 12, 2022, doi: 10.3390/s22124399.

[40] F. W. Alsaade, T. H. H. Aldhyani, and M. H. Al-Adhaileh, “Developing a Recognition System for Diagnosing Melanoma Skin Lesions Using Artificial Intelligence Algorithms,”

Comput Math Methods Med, vol. 2021, 2021, doi: 10.1155/2021/9998379.

[41] E. Goceri, “Diagnosis of skin diseases in the era of deep learning and mobile technology,” *Comput Biol Med*, vol. 134, Jul. 2021, doi: 10.1016/j.compbiomed.2021.104458.

[42] P. R. Kshirsagar, H. Manoharan, S. Shitharth, A. M. Alshareef, N. Albishry, and P. K. Balachandran, “Deep Learning Approaches for Prognosis of Automated Skin Disease,” *Life*, vol. 12, no. 3, Mar. 2022, doi: 10.3390/life12030426.

[43] E. Soria Olivas, P. Rodriguez Belenguer, and Q. Garcia Vidal, *Inteligencia Artificial: casos practicos con aprendizaje profundo*. RA-MA Editorial, 2022.

[44] J. Bobadilla, *Machine Learning y Deep Learning: Usando Python, Scikit y Keras*. RA-MA Editorial, 2020.

[45] A. Bosch Rue, J. Casas Roma, and T. Lozano Bagen, *Deep learning: principios y fundamentos*. Editorial UOC, 2019.

[46] L. F. Torres, “Convolutional Neural Network From Scratch.” [Online]. Available: https://medium.com/latinxinai/convolutional-neural-network-from-scratch-6b1c856e1c07

[47] M. A. Arif, “Understanding VGG16: A Powerful Deep Learning Model for Image Recognition.” [Online]. Available: https://smuhabdullah.medium.com/understanding-vgg16-a-powerful-deep-learning-model-for-image-recognition-d40b074fd01c

[48] Nutan, “Image Classification With ResNet50 Model,” 2021.

ANEXOS

Anexo 1: Acta de revisión de similitud de la investigación

Universidad Señor de Sipán

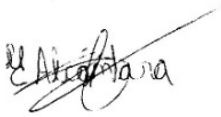
DECLARACIÓN JURADA DE ORIGINALIDAD

Quien(es) suscribe(n) la DECLARACIÓN JURADA, somos **Alcantara Calderon Gianmarco y Arica Guerrero Lauren David,** del Programa de Estudios de **Ingeniería de Sistemas** de la Universidad Señor de Sipán S.A.C, declaramos bajo juramento que somos autores del trabajo titulado:

CLASIFICACIÓN DE LA VIRUELA DEL MONO MEDIANTE APRENDIZAJE PROFUNDO Y TECNICAS DE PROCESAMIENTO DE IMAGENES

El texto de mi trabajo de investigación responde y respeta lo indicado en el Código de Ética del Comité Institucional de Ética en Investigación de la Universidad Señor de Sipán, conforme a los principios y lineamientos detallados en dicho documento, en relación con las citas y referencias bibliográficas, respetando el derecho de propiedad intelectual, por lo cual informo que la investigación cumple con ser inédito, originaly autentico.

En virtud de lo antes mencionado, firman:

Alcantara Calderon Gianmarco	74727547	
Arica Guerrero Lauren David	73581496	

Pimentel, 15 de Diciembre de 2023.

Anexo 2. Acta de aprobación del asesor

Universidad Señor de Sipán

ANEXO 03: ACTA DE APROBACIÓN DEL ASESOR

Yo **Víctor Alexci Tuesta Monteza**. quien suscribe como asesor designado mediante Resolución de Facultad N° **0426-2023**, del proyecto de investigación titulado **Clasificación de la viruela del mono mediante aprendizaje profundo y técnicas de procesamiento de imágenes**., desarrollado por el(los) estudiante(s): **Alcantara Calderon Gianmarco, Arica Guerrero Lauren David**., del programa de estudios de **Ingeniería de Sistemas**, acredito haber revisado, realizado observaciones y recomendaciones pertinentes, encontrándose expedito para su revisión por parte del docente del curso.

En virtud de lo antes mencionado, firman:

Tuesta Monteza Víctor Alexci	DNI: 42722929	
Alcantara Calderon Gianmarco	**DNI: 74727547**	
Arica Guerrero Lauren David	DNI: 73581496	

Pimentel, 20 de diciembre de 2023

Anexo 3. Tabla de matriz de operacionalización

Variable de estudio	Definición conceptual	Definición operacional	Dimensión	Indicadores	Instrumento	Valores finales	Tipo de variable	Escala de medición
Algoritmos de aprendizaje profundo	Los algoritmos de aprendizaje profundo son redes neuronales que se modelan en base al cerebro humano con el fin de resolver problemas complejos empleando cálculos matemáticos para el procesamiento de datos.	Obtener un dataset que incluya los datos necesarios para realizar el procesamiento con el algoritmo de aprendizaje profundo. Seleccionar los algoritmos de aprendizaje profundo más adecuados para realizar la clasificación. Realizar el entrenamiento a cada algoritmo con los datos obtenidos y ajustando los parámetros del algoritmo con el fin de obtener el mejor desempeño. Analizar y comparar los resultados obtenidos de los algoritmos de aprendizaje profundo.	Consumo de recursos	Consumo de GPU $cp = \sum_{i}^{n} \frac{ce_i}{n}$	Técnica: Observación Ficha digital de observación	Gigabytes (Mb)	Variable independiente	Razón (0GB – 15GB)
				Consumo de memoria RAM $cm = \sum_{i}^{n} \frac{cm_i}{n}$				
				Tiempo de ejecución $t = \sum_{j}^{n} \frac{tf_j - tf_i}{n}$		Minutos (Min)		Proporcional (0% – 100%)

Clasificación de la viruela del mono	Es una enfermedad causada por un virus que genera ampollas o granos en la piel, esta definición se basa en la revisión de artículos y está relacionada con el diagnóstico de las enfermedades cutáneas.	La clasificación de la viruela del mono se puede dar mediante la evaluación clínica realizada por médicos donde analizan los síntomas comunes, análisis de muestras donde se toman muestras corporales de las lesiones, análisis histopatológico se realiza biopsia de las lesiones para examinar los tejidos y el estudio de imágenes radiográficas.	Rendimiento	Precisión $P = \frac{VP}{VP + FP}$		Los valores finales serán el porcentaje (%)	Variable dependiente	Proporcional (0% – 100%)
				Exactitud $E = \frac{VP + VN}{VP + FP + FN + VN}$				
				Recall $R = \frac{VP}{VP + FN}$				
				F-score $F = 2 * \frac{P * R}{P + \mathrm{R}}$				

Anexo 4. Reporte de similitud de turnitin

Reporte de similitud

NOMBRE DEL TRABAJO

TESIS_Arica-Alcantara nueva turnitin.doc x

RECUENTO DE PALABRAS

6318 Words

RECUENTO DE CARACTERES

33771 Characters

RECUENTO DE PÁGINAS

30 Pages

TAMAÑO DEL ARCHIVO

1.8MB

FECHA DE ENTREGA

Nov 11, 2024 10:45 AM GMT-5

FECHA DEL INFORME

Nov 11, 2024 10:46 AM GMT-5

15% de similitud general

El total combinado de todas las coincidencias, incluidas las fuentes superpuestas, para cada base de datos.

- 9% Base de datos de Internet
- 4% Base de datos de publicaciones
- Base de datos de Crossref
- Base de datos de contenido publicado de Crossref
- 10% Base de datos de trabajos entregados

Excluir del Reporte de Similitud

- Material bibliográfico
- Material citado
- Coincidencia baja (menos de 8 palabras)

Anexo 5. Listado de artículos científicos

N°	Título	Algoritmos	Accuracy	Precision	F-score	Recall
1	Skin lesion detection based on deep neural networks	Proposed	85,45%	82,40%	-	86,47%
		SVM	78,24%	72,95%	-	82,87%
		DT	77,57%	72,60%	-	78,15%
		KNN	76,14%	74,50%	-	76,86%
		Naive Bayes	73,03%	72,50%	-	74,36%
2	Multi-Class Skin Lesions Classification Using Deep Features	Naive Bayes	81,34%	-	-	-
		ELM	84,92%	-	-	-
		KELM	90,67%	-	-	-
		MSVM	85,50%	-	-	-
		Fine KNN	82,08%	-	-	-
3	Skin Lesion Area Segmentation Using Attention Squeeze U-Net for Embedded Devices	U-Net	89,65%	-	-	-
		Attention U-Net	87,66%	-	-	-
		Squeeze U-Net	89,87%	-	-	-
		Attention Squeeze U-Net	90,35%	-	-	-
4	Skin Lesion Classification Using Collective Intelligence of Multiple Neural Networks	Xception	78,45%	-	-	-
		DenseNet-201	78,59%	-	-	-
		IncpetionResNet-V2	79,39%	-	-	-
		GoogleLeNet-Place365	79,45%	-	-	-
		GoogLeNet	79,45%	-	-	-
		AlexNet	79,65%	-	-	-
		MobileNet-V2	80,59%	-	-	-
		ResNet50	81,12%	-	-	-
		ResNet-101	83,99%	-	-	-
		Proposed CIS	86,71%	-	-	-
5	Deep Learning Approaches for Prognosis of Automated Skin Disease	FTNN	80,23%	85,07%	-	80,65%
		CNN	81,67%	86,07%	-	81,75%
		Depth-based CNN	82,93%	80,49%	-	80,23%
		Channel boost CNN	83,45%	82,39%	-	81,24%
		MobileNet V1	83,34%	90,92%	-	85,40%
		MobileNet V2	85,23%	91,69%	-	87,51%
		MobileNet V2–LSTM	86,57%	93,34%	-	89,34%
6	Skin cancer diagnosis based on deep transfer learning and sparrow search algorithm	MobileNet	98,27%	98,27%	98,27%	98,27%
		MobileNetV2	97,42%	97,42%	97,42%	97,42%
		MobileNetV3Small	93,00%	93,00%	93,00%	93,00%
		MobileNetV3Large	96,10%	96,10%	96,10%	96,10%
		VGG16	95,29%	95,29%	95,29%	95,29%
		VGG19	96,94%	96,94%	96,94%	96,94%

		NASNetMobile	96,83%	96,83%	96,83%	96,83%
		NASNetLarge	93,20%	93,20%	93,20%	93,20%
7	NEURAL NETWORKS FROM KERAS IN SKIN LESION DIAGNOSTIC	VGG16	90,10%	-	-	-
		ResNet50	92,10%	-	-	-
		MobileNet	89,50%	-	-	-
8	Automatic skin disease diagnosis using deep learning from clinical image and patient information	Mobilent-v2	98,30%	98,50%	98,50%	98,50%
9	A Convolutional Neural Network Framework for Accurate Skin Cancer Detection	DenseNet201	95,09%	93,46%	-	91,57%
		GoogLeNet	89,80%	84,13%	-	85,20%
		Inception-ResNetV2	89,02%	78,90%	-	91,15%
		InceptionV3	93,02%	93,88%	-	84,38%
		MobileNetV2	91,48%	92,41%	-	80,88%
10	Diagnosis of skin diseases in the era of deep learning and mobile technology	SqueezeNet	68,26%	61,66%	63,69%	-
		ShuffleNet	71,45%	61,75%	63,79%	-
		MobileNet	76,08%	69,48%	71,52%	-
		RMNv2	76,17%	69,59%	71,62%	-
		MobileNetV2	81,19%	74,61%	76,65%	-
		LWEN	83,32%	76,71%	78,76%	-
		LB-FCN light	85,68%	79,01%	80,97%	-
		CustomNet2	88,01%	81,35%	83,40%	-
		Modified-MobileNet	94,76%	90,60%	91,31%	-
11	Automated multi-class classification of skin lesions through deep convolutional neural network with dermoscopic images	DCNN	-	94,00%	-	-
12	Developing a Recognition System for Diagnosing Melanoma Skin Lesions Using Artificial Intelligence Algorithms	AlexNet	91,67%	75,00%	85,71%	-
		ResNet50	94,05%	81,25%	88,64%	-
13	The effects of skin lesion segmentation on the performance of dermatoscopic image classification	SkinLinkNet	89,60%	-	-	-
		SkinUNet	89,80%	-	-	-
		SkinFPN+	89,80%	-	-	-
14	Recognition of Skin Diseases using Deep Neural Network Optimized by Group Teaching Algorithm	DNN-GTO classifier	98,75%	97,10%	-	98,12%
15	Deep Learning–Based Methods for Automatic Diagnosis of Skin Lesions	Global Classifier	95,00%	-	-	-
16		RF	82,22%	42,37%	45,12%	48,26%

	Deep Learning Based Skin Lesion Segmentation and Classification of Melanoma Using Support Vector Machine (SVM)	SVM	85,19%	42,59%	46,00%	50,00%
		KNN	79,26%	45,04%	45,92%	47,55%
		NB	65,93%	45,62%	44,40%	43,86%
17	Multiple skin lesions diagnostics via integrated deep convolutional networks for segmentation and classification	Inception-v3	77,04%	-	78,39%	-
		ResNet50	79,95%	-	80,85%	-
		Inception-ResNet-v2	81,79%	-	82,59%	-
		DenseNet-201	81,27%	-	81,73%	-
18	An enhanced technique of skin cancer classification using deep convolutional neural network with transfer learning models	AlexNet	93,82%	97,88%	94,85%	92,01%
		ResNet	91,25%	96,00%	94,80%	93,63%
		VGG16	86,17%	100,00%	92,48%	86,02%
		DenseNet	91,64%	91,42%	93,09%	94,83%
		MobileNet	90,31%	92,08%	93,10%	94,15%
		Proposed DCNN model	93,16%	96,57%	95,09%	93,66%
19	Automatic diagnosis of skin diseases using convolution neural network	AlexNet	91,03%	-	-	-
20	A comparative study of deep learning architectures on melanoma detection	AlexNet	80,45%	84,21%	82,31%	81,25%
		ResNet50	92,08%	93,73%	92,74%	92,53%
		VGGNet16	88,36%	90,70%	90,61%	90,32%
		VGGNet19	88,70%	88,55%	88,38%	88,82%
		Xception	90,30%	90,19%	90,41%	90,57%
21	Studies on Different CNN Algorithms for Face Skin Disease Classification Based on Clinical Images	Resnet50	-	62,90%	-	63,40%
		Incpetion V3	-	64,00%	-	66,60%
		Densenet-121	-	68,80%	-	68,20%
		Xception	-	68,10%	-	70,60%
		Incpetion-Resnet V2	-	70,80%	-	77,00%
22	Skin lesion image retrieval using transfer learning-based approach for query-driven distance recommendation	ResNet50	-	97,00%	97,00%	97,00%
23	Skin Disease Detection for Kids at School Using Deep Learning Techniques	CNN	99,00%	-	-	-
		VGG19	99,00%	-	-	-
24	Skin disease detection and segmentation using dynamic graph cut algorithm and classification through Naive Bayes classifier	Naive Bayes	-	-	-	-
25	MonkeyNet: A robust deep convolutional neural network for monkeypox disease detection and classification	ResNet50	95,86%	95,89%	95,87%	95,86%
		MobileNetV1	96,44%	96,48%	96,44%	96,44%
		Inception V3	97,70%	97,71%	97,70%	97,70%
		VGG16	94,43%	94,48%	94,44%	94,43%

		Xception	96,49%	96,53%	96,50%	96,49%
26	EczemaNet: A Deep CNN-based Eczema Diseases Classification	InceptionV3	91,00%	87,00%	-	-
		MobileNetV1	92,00%	88,00%	-	-
27	A Comparison of Neural Network Approaches for Melanoma Classification	2D CNN	74,10%	-	-	-
		ResNet	81,50%	-	-	-
		SOM	69,00%	-	-	-
28	An Al-Biruni Earth Radius Optimization-Based Deep Convolutional Neural Network for Classifying Monkeypox Disease	BERSFS-CNN	98,00%	-	80,00%	-
		AlexNet	94,00%	-	41,00%	-
		GoogLeNet	93,00%	-	37,00%	-
		VGG19Net	92,00%	-	34,00%	-
		ResNet50	92,00%	-	27,00%	-
29	Deep transfer learning approaches for Monkeypox disease diagnosisDeep transfer learning approaches for Monkeypox disease diagnosis Deep transfer learning approaches for Monkeypox disease diagnosis	Xception	80,00%	80,00%	-	80,00%
		ResNet101	99,00%	99,00%	-	99,00%
30	An in-depth analysis of Convolutional Neural Network architectures with transfer learning for skin disease diagnosis	ResNet50	96,00%	87,00%	87,00%	87,00%
		InceptionV3	98,00%	93,00%	93,00%	93,00%
		Inception-Resnet	98,00%	95,00%	95,00%	95,00%
		DenseNet	94,00%	93,00%	93,00%	93,00%
		MobileNet	96,00%	96,00%	96,00%	96,00%
		Xception	97,00%	97,00%	97,00%	97,00%
31	An improved transformer network for skin cancer classification	MobileNetV2	89,00%	89,00%	89,00%	-
		ResNet50	92,00%	91,00%	92,00%	-
		InceptionV2	91,00%	91,00%	91,00%	-
		VIT	93,00%	93,00%	93,00%	-
32	A CNN-Based Model for Early Melanoma Detection	VGG16	81,00%	-	-	-
		SVM	86,00%	-	-	-
		CNN	90,00%	-	-	-
		AlexNet, GoogLeNet	90,00%	-	-	-
33	DETECTION OF SKIN CANCER USING DEEP NEURAL NETWORKS	ResNet50	-	91,00%	89,00%	89,00%
34	AS-Net: Attention Synergy Network for skin lesion segmentation	ResNet50	94,00%	-	-	-
		DenseNet121	93,00%	-	-	-
		EfficientNetB0	94,10%	-	-	-
		EfficientNetB5	94,50%	-	-	-
		VGG16	94,60%	-	-	-
35	Skin lesion segmentation from dermoscopic images by using Mask R-CNN, Retina-Deeplab, and graph-based methods	ResNet	87,00%	-	-	-
		DenseNet	87,00%	-	-	-