Journal of Applied Sciences, Information and Computing

Volume 5, Issue 1, June 2024

ISSN: 1813-3509 https://doi.org/10.59568/JASIC-2024-5-1-03

An enhanced image based mobile deep learning model for identification of Newcastle poultry disease

[1]Damilare Andrew Omideyi, [2]Rafiu Mope Isiaka, [3]Ronke Seyi Babatunde

[1]Department of Computer Science, Faculty of Natural Science, Ajayi Crowther University Oyo, Nigeria.
[1]da.omideyi@acu.edu.ng
[2, 3] Department of Computer Science, Faculty of Information and Communication Technology, Kwara State University, Molete, Nigeria
[2]abdulrafiu.isiaka@kwasu.edu.ng
[3] ronke.babatunde@kwasu.edu.ng

Abstract

An enhanced mobile deep learning model based on images is presented in this paper to identify Newcastle poultry disease. A dataset of manually annotated and labeled images of the disease was utilized to pre-train an image-based Convolutional Neural Network (CNN). An Android smartphone app was developed to communicate with the model. A local server was integrated with the generated model to do image classification. A mobile application was developed and made available, enabling users to upload a fecal photograph to a website housed on the streamlet server and obtain the model's processed findings. The user regains control over their health status. The model achieved an accuracy of 95% on the test set and was able to correctly identify specific instances of Newcastle poultry disease. The paper discusses the advantages of a mobile-based approach in comparison to traditional methods of identification and proposes the model as an effective low-cost solution for farmers and researchers.

Keywords. Convolutional Neural Network, Newcastle disease, deep learning model, fecal image, image data.

1. Introduction

Recent advancements in deep learning and mobile computing offer promising opportunities to revolutionize Disease diagnosis, including Newcastle Disease (ND) detection in poultry. Deep learning techniques, particularly

convolutional neural networks (CNNs), have demonstrated remarkable capabilities in image recognition and classification tasks. Moreover, the integration of deep learning models with mobile devices allows for real-time analysis and decision-making at the point of care, facilitating timely intervention and control measures.

This study develop an enhanced image-based mobile deep learning model specifically tailored for the identification of Newcastle poultry disease. By leveraging the power of deep learning algorithms optimized for mobile platforms, the model seeks to provide accurate and rapid diagnosis of Newcastle Disease (ND) directly on-site, empowering poultry farmers and veterinarians with actionable insights for disease management.

The mobile concept in this study refers to the integration of deep learning algorithms with mobile devices, enabling on-device processing of images for disease identification. Mobile devices such as smartphones and tablets are widely available and accessible, even in remote or resource-limited settings. By deploying deep learning models on mobile platforms, Newcastle Disease (ND) diagnosis can be performed directly at the point of care without the need for specialized equipment or infrastructure.

Mobile deep learning models enable real-time analysis of images captured on-site, allowing for immediate feedback and decision-making. This capability is particularly valuable in situations where timely intervention is critical, such as during disease outbreaks in poultry farms. Mobile deep learning models eliminate the need for expensive hardware or cloud-based services for image processing and analysis. By leveraging the computational capabilities of mobile devices, this model offers a cost-effective solution for Newcastle Disease (ND) diagnosis in poultry farming operations.

Mobile technology are changing dramatically as a result of the widespread use of smartphones and tablets, with a heavy emphasis on computer vision. The majority of mobile applications—including those for social media, e-commerce, healthcare, and other industries—are image-based.

2. Review of Literature:

These applications leverage deep learning to enable devices to interpret, process, and extract valuable insights from visual data in real-time. Image-based mobile deep learning models have been extensively studied in various domains. A convolutional neural network (CNN) was proposed by research by Karthikeyan (2022) to separate garbage into biodegradable and non-biodegradable wastes which achieved an accuracy of 97%. Research by Alem, A., & Kumar, S. (2022) demonstrates the utilization of distant sensed hyperspectral pictures for land survey and classification utilizing deep learning algorithms such as EfficientNetB7, InceptionV3, and Mobile Net. Complex deep learning models can now be installed directly onto mobile devices through recent advancements in neural network architecture and hardware improvements for mobile devices. With the aid of this innovation, mobile applications can now analyze images intelligently without the need of distant servers, ushering in a new era.

Grayscale images were used in a method that Mercaldo, F., & Santone, A. (2020) presented to distinguish between malicious and valid samples in the mobile environment, with encouraging results. Gupta, K., and Bajaj, V. (2023) established a robust framework for the automated detection of COVID-19 using chest CT scan images. Transfer-learned with 98.91%, DarkNet19 had the best categorization accuracy. These studies show how mobile deep learning models based on images perform well in a range of applications. Applications for mobile devices have grown in importance across a range of industries, including productivity, healthcare, and education. (Ahmadzadegan, M. H., et al., 2020; Józsa, T. 2020). They provide a number of functions and features that help enhance overall wellbeing, attention span, symptom control, time

 https://doi.org/10.59568/JASIC-2024-5-1-03

management, and cognitive function. (Vega-Ramírez, Let al., 2020).

Mobile application will assist people with chronic conditions keep an ongoing eye on their health, which may reduce the number of doctor visits that are necessary. Smartphones and mobile application have the potential to improve student motivation and learning experiences in the realm of education. The efficiency and efficacy of mobile applications are enhanced by their technical capabilities, which include social media networking, search functions, and flexible designs. (Ahmadzadegan, M. H., et al., 2020). Overall, mobile applications have the potential to transform many different kinds of industries and increase the productivity, accuracy, and speed of diverse jobs and activities.

Aim and Objectives

The aim of this research is to construct a model capable of precisely detecting and classifying Newcastle disease in poultry through the utilization of mobile devices.

The objectives of the study to acquire fecal image dataset; develop model using machine learning and deep learning approach: evaluate the developed system based on some metrics.

The model encompasses the utilization of artificial intelligence and machine learning techniques, specifically deep Convolutional Neural Networks (CNN), to analyze images of poultry fecal samples and identify the presence of Newcastle disease. The scope of the study includes using fecal images of chickens as input for the deep learning model. The study aims to leverage the capabilities of deep Convolutional Neural Networks (CNN) for accurate prediction and diagnosis of Newcastle disease in chickens. The model will be developed using pre-trained models such as Convolutional Neural Networks (CNN) and MobileNet, and their efficacy will be assessed through comparative analysis to ascertain the optimal model for poultry disease detection. This study evaluate the performance of the model in terms of accuracy and prediction capabilities. The model is designed to be deployed on mobile devices, making it accessible and user-friendly for poultry farmers, extension officers, and other stakeholders involved in poultry disease detection.

Statement Problem

The existing approaches for poultry disease diagnostics have limitations in terms of early detection and accuracy. Limited access to agricultural support services and experts in poultry farming hinders early detection and diagnosis of diseases. Traditional farming practices and lack of systematic methods contribute to the spread of diseases in poultry. There is a need for reliable and efficient methods to detect and diagnose poultry diseases, especially in areas with limited access to experts. The existing approaches rely on manual counting of oocytes in stool or intestinal tracts, virus detection, and polymerase chain reaction (PCR) procedures, which can be time-consuming and require multiple diagnoses. Late diagnoses or a lack of credible specialists result in significant losses for poultry farmers. The accuracy and performance of existing models for poultry disease diagnostics vary, with some models achieving higher accuracies than others. There is a need to bridge the gap between existing approaches and develop more accurate efficient models for early identification and classification of poultry ailments.

3. Methodology:

The methodology adopted for this study are acquiring of dataset (fecal images) including both healthy and diseased samples. The dataset are diverse and representative of different variations of Newcastle poultry disease.

Preprocessing of the acquired images to include reducing them to a standard resolution, pixel value normalization, and dataset augmentation (including rotation, flipping, and zooming) to boost size and variability. Choose a relevant

 https://doi.org/10.59568/JASIC-2024-5-1-03

pre-trained deep learning model for the task of image classification. Models CNN and Mobile Net can be considered based on their performance and efficiency. The chosen pre-trained model initializes transfer learning using weights acquired from a large set of images (such as ImageNet). Fine-tune the model is done by training it on the collected fecal image dataset specific to Newcastle poultry disease. The dataset is divided into training and validation sets for the purpose of training this model. Gradient descent and back-propagation are used in the deep learning model's training set to optimize the model's parameters. In order to avoid overfitting, monitoring of the model's performance on the validation set is essential.

Accuracy, precision, recall, and F1 score are relevant assessment metrics that are used to assess the performance of the model. Use the validation set to assess the model's ability to correctly classify healthy and diseased fecal images. The deployment is done by converting the trained model into a mobile-friendly format and integrate it into a mobile application. The application has a user-friendly interface for capturing and uploading fecal images for disease identification. The Testing of the application on mobile devices is done to ensure its functionality and performance

3.1 Feature Extraction and Feature Selection

It is nowadays becoming quite common to be working with datasets of hundreds (or even thousands) of features. If the amount of features becomes similar than the amount of observations stored in a dataset then this may presumably cause a Machine Learning model affected by overfitting. So as to avoid this sort of problem, it's necessary to use either regularization or dimensionality reduction techniques (Extraction of Features). Feature extraction is a dimensionality reduction technique that partitions an initial collection of data into more manageable processing groups. Numerous variables in these enormous data sets necessitate a high computing capability to handle. Techniques that combine or select variables to produce features are referred to as "feature extraction" because they reduce the quantity of data that needs to be analyzed while maintaining accurate and comprehensive descriptions of the original data set. (www.deepai.org, 2019).

The purpose of feature extraction is to condense the information needed to interpret extensive data sets. A primary challenge in intricate image analysis lies in managing the multitude of variables involved. In addition to requiring an excessive amount of memory and processing capacity, analysis with an excessive number of variables sometimes results in a classification formula that over fits to training samples and performs poorly on replacement samples. A broad name for methods of creating variable combinations to get around these problems while still providing an accurate enough description of the data is feature extraction.

Unlike feature extraction methods, which transform data into a new representation, feature selection techniques select a subset of features from the original set. This selected subset aims to achieve optimal performance based on specific criteria and objective functions. Feature selection serves as a dimensionality reduction method, aiming to eliminate irrelevant and redundant features. Moreover, it enhances classification accuracy (Maksoud & Elmogy, 2019).

A goal shared by feature extraction and selection techniques is to prevent overfitting of the data. It provides an opportunity to generate more analysis. Algorithms for feature selection are divided into two categories: Filters extract features from data without necessitating learning, whereas wrappers identify useful features using learning-based approaches. Embedded methods that integrate feature selection with the establishment of the classifier.

 https://doi.org/10.59568/JASIC-2024-5-1-03

Feature selection aims to seek out the most important information to save lots of computational efforts and data storage.

3.2 Preprocessing Stage

At the stage, the datasets from the Machine Learning Dataset 1 Machine Learning Dataset 2. Using Polymerase Chain Reaction (PCR) and farm-labeled fecal images, this series of data about poultry disease diagnostics was annotated. The fecal image excrement were taken in Tanzania in the Arusha and Kilimanjaro regions between September 2020 and February 2021. The normal fecal material, or the 'healthy' class, and the Newcastle disease, or the 'ncd' class, were collected from poultry farms; the chickens were also inoculated against Newcastle disease, and fecal images for the 'ncd' class were obtained within three days. For more technical details, it is necessary to divide the data into the respective directories in order to support images for Artificial Neural Network learning using Keras and Tensorflow tools:

3.3 Training and Learning of Dataset

Learning is the process by which a system is taught and made adaptive so that it can provide results accurately. The learning stage holds paramount importance as the algorithms applied to the data dictate the system's performance with the supplied data. The entire dataset is segregated into two categories: the training set, utilized for model training, and the testing set, employed to evaluate the model post-training.

Training Set: A model is developed utilizing the training dataset. It is made up of the collection of images that the system is trained on. Relevant information on how to link input data with output decision is provided by the training rules and algorithms that are used. By using these algorithms on the dataset, the system is trained; every relevant detail is taken out of the data, and outcomes are produced.

Testing Set: To verify the system, testing data is used. The collected data is employed to validate the accuracy of the system's output post-training. The accuracy of the system is confirmed by data testing.

Train subdirectory is data to use for neural network learning to find optimal weight and make architecture. It is has to done with training of the image dataset using neural network algorithm, 80% of the dataset is set-side for the purpose of training.

Validation subdirectory: In this path, the training network is being validated to ensure that there is no overfitting during training, 10% of the dataset is set-side for the validation process.

Testing subdirectory is data to make final capability with our model. Due to ncd class in the train subdirectory finding an imbalance problem, we decide to use image augmentation to increase image in this class such as flip, shift and rotate and add noise and contrast. In conclusion, we preprocess the data to make it compatible with training deep learning models using TensorFlow and Keras. We utilize a CNN model and implement transfer learning and fine-tuning techniques to enhance the performance of the image classification task. So, we selected CNN and mobilenetv2 model because all both is easy to us in Tensorflow api and high accuracy for this problem.

In this subdirectory, the models were tested individually using 80% of the dataset. The test was done to determine the level of accuracy of the model. The best model (MobileNetV2) in term of accuracy was implemented and deployed into a mobile poultry disease detection mobile application.

 https://doi.org/10.59568/JASIC-2024-5-1-03

4. Implementation of the New System

4.1 Data importation

The data downloaded has to be imported in the system for implementation to take place. The code for data is stated below:

```
# Importing necessary libraries
import os
from os import makedirs, listdir
from shutil import copyfile
import cv2
from PIL import Image
from random import seed, random
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import albumentations as A
```

4.2 Dataset Training

The imported data has to be trained, validated and tested under what we describe as training validation and testing subdirectory. The code to achieve this is as stated below:

```
# create diretories
# train test validate
dataset_home = 'dataset_train_valid_test/' # name of diretories
subdirs = ['train/', 'valid/','test/'] # sub diretories
for subdir in subdirs:
# create label subdirectories
labeldirs = ['healthy/','ncd/']
for labldir in labeldirs:
newdir = dataset_home + subdir + labldir
makedirs(newdir, exist_ok=True)
In [3]: # seed random number generator
seed(42)
# define ratio of pictures to divide
# train:80/test:10/valid:10
val_ratio_1 = 0.2 # divide 80/20 of all data
val_ratio_2 = 0.5 # divide 50/50 of divided 20% all data
In [4]: # you can `download dataset form https://doi.org/10.5281/zenodo.5801834, https://doi.org/10.5281/zenodo.4628934`
# and bring all together in a folder
# you must rename image data for pcr dataset such as pcrcocci.1.jpg to cocci.x.jpg
# (x follow by after order from farmmer-labeled dataset)
src_directory = 'all' # path diretories have all data
# random data form all diretories to of each subdirectories
for file in listdir(src_directory):
src = src_directory + '/' + file
dst_dir = 'train/'
# train valid test split
if random() < val_ratio_1: # 80/20
if random() < val_ratio_2: # 50/50
dst_dir = 'test/'
else :
dst_dir = 'valid/'
# copyimages into subdirectories
if file.startswith('cocci'):
dst = dataset_home + dst_dir + 'cocci/'+ file
copyfile(src, dst)
elif file.startswith('healthy'):
dst = dataset_home + dst_dir + 'healthy/' + file
copyfile(src, dst)
elif file.startswith('ncd'):
dst = dataset_home + dst_dir + 'ncd/' + file
copyfile(src, dst)
elif file.startswith('salmo'):
dst = dataset_home + dst_dir + 'salmo/' + file
copyfile(src, dst)
In [5]: # Recheck
# cocci in train
train_cocci = len(listdir('dataset_train_valid_test/train/cocci/'))
# cocci in test
test_cocci = len(listdir('dataset_train_valid_test/test/cocci/'))
# cocci in valid
valid_cocci = len(listdir('dataset_train_valid_test/valid/cocci/'))

# Show percent of each subdirectories in cocci class
print(f"train : {round(train cocci/(train cocci+test cocci+valid cocci),3)}")

print(f train : {round(train_cocci/(train_cocci+test_cocci+valid_cocci), 3)} ) print(f"test : {round(test_cocci/(train_cocci+test_cocci+valid_cocci),3) }") print(f"validate : {round(valid_cocci/(train_cocci+test_cocci+valid_cocci), 3)}")
```

 https://doi.org/10.59568/JASIC-2024-5-1-03

Train: 0.807

Test: 0.094

Validate: 0.1

4.3 Image Augmentation

Once fecal images are collected, we clean the dataset by removing irrelevant features to facilitate algorithm training. This process also enables us to keep the network from overfitting and give it the ability to generalize.

In the case of Convolutional Neural Networks, a larger dataset is necessary, and image augmentation aids in artificially expanding the training dataset by generating variations of images.

Because the augmented images offer distinct variants in addition to the original training dataset, it improves the performance of the models. Through the use of zoom, shift, flip, rotation, mirror-imaging, cropping, and other techniques, we change the images, or transform the data.

In summary, image augmentation improves the model's performance and allows for a larger training dataset. To make more copies of the first training images, we can move, flip, crop, and enlarge the images.

The code to achieve this is as stated below:

```
# check number of each group in train subdirectories

train_ncd = len(listdir('dataset_train_valid_test/train/ncd/'))

train_healthy = len(listdir('dataset_train_valid_test/train/healthy/'))

# imbalance class in ncd just have 450 image
print(f"Train dataset ncd class : {train_ncd}")
print(f"Train dataset healthy class : {train_healthy}")
Train dataset ncd class : 450
Train dataset healthy class : 1903
# use technique image augmentation for increase image data in class ncd #

for help imbalance class problem

# image augmentation flip horizontal(mirror) image

transform_flip = A.Compose([A.HorizontalFlip(always_apply=True)])

# image augmentation rotation image 45 degree angle, rescale 10% and shift 0.062%

transform_shift = A.Compose([A.ShiftScaleRotate(shift_limit=(0.0625, 0.0625), scale_limit=(0.1, 0.1), rotate_limit=(45, 45), p=1.0)])

# image augmentation add gaussian noise and decrease brightness transform_noise = A.Compose([A.GaussNoise(var_limit=(100,200), mean=-30, p=1.0)])

# source code and example

#https://github.com/albumentations-team/albumentations#pixel-level-transforms

#https://tugot17.github.io/data-science-blog/albumentations/data-augmentation/tutorial/2020/09/20/Pixel-level-transforms-using-albumentations-package

# Open sample image data

Image= v2.imread ("./dataset_train_valid_test/train/ncd/ncd.13.jpg")
image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB) # convert to pixel numpy array#
transform filp and shift
transformed_flip = transform_flip(image=image)['image']
transformed_shift= ransform_shift(image=image)['image']
transformed_noise = transform_noise(image=image)['image']
# Show origin and transform image

plt.imshow(image)

plt.title("Origin image", size=15)

plt.show() plt.imshow(transformed_flip)

plt.title("Filp image", size=15)

plt.show()

plt.imshow(transformed_shift)

plt.title("Shift image", size=15)

plt.show() plt.imshow(transformed_noise) plt.title("Noise image", size=15)
```

 https://doi.org/10.59568/JASIC-2024-5-1-03

```
plt.show()
```

Original Image (Figure 1)

Flip Image (Figure 2)

Shift Image (Figure 3

Noise image (Figure 4)

```
# loop for use augmentation of each image in train directory class ncd
src_directory = "dataset_train_valid_test/train/ncd/"
for file in listdir(src_directory):
image = cv2.imread(src_directory + file)
image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
# transform and save new flie
transformed_flip = transform_flip(image=image)['image']
cv2.imwrite(src_directory + file[:-4] + "flip.jpg", cv2.cvtColor(transformed_flip, cv2.COLOR_RGB2BGR))
transformed_shift = transform_shift(image=image)['image']
cv2.imwrite(src_directory + file[:-4] + "ship.jpg", cv2.cvtColor(transformed_shift, cv2.COLOR_RGB2BGR))
transformed_noise = transform_noise(image=image)['image']
cv2.imwrite(src_directory + file[:-4] + "noise.jpg", cv2.cvtColor(transformed_noise, cv2.COLOR_RGB2BGR))
```

Considering figure 1 & 2 under image augmentation, it was observed that the network is relatively stable once the images looks absolutely stable compared to original image when flipped or shifted and even when some traces of noise was introduced.

4.4 Convolutional Neural Network Model Training and Testing

For this purpose, Tensor flow is used for training deep learning model, the algorithm is made of up of various components

```
In [1]: # Import librarys
# normal
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
# open file
from IPython.display import Image
import os
import PIL
# model CNN (Deep learning network)
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Dense,\
GlobalAveragePooling2D, Dropout, Flatten
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.preprocessing.image

import ImageDataGenerator

In [2]: # Create function plot loss function and accuracy score graph
def plot_graph(model_values):
''' Input: Model_values of keras.callbacks.History
Return: Graph of Loss function and accuracy score between training dataset and vaildation dataset'''
# Subplots
```

 https://doi.org/10.59568/JASIC-2024-5-1-03

```
fig, ax = plt.subplots(1, 2, figsize=(14,5))
# Plot loss
plt.subplot(1, 2, 1)
plt.plot(model_values.history['loss'],label='Training Loss');
plt. Plot ( model_values .history ['val_loss'], label='Testing Loss');
plt. Legend ( fontsize = 12,loc='upper right')
plt .title('Training and Validation Loss')
plt. xlabel('Epochs')
plt.ylabel('Loss');# Plot MSE
plt .subplot(1, 2, 2)
plt.plot (model_values. History ['accuracy'], label= 'Training Accuracy')
plt.plot(model_values.history['val_accuracy'],label='Validation Accuracy')
plt.legend(fontsize=12, loc='lower right')
plt.title('Training and Validation Accuracy')
plt.xlabel('Epochs')
plt .ylabel('Accuracy');
```

CNN model code to generate result

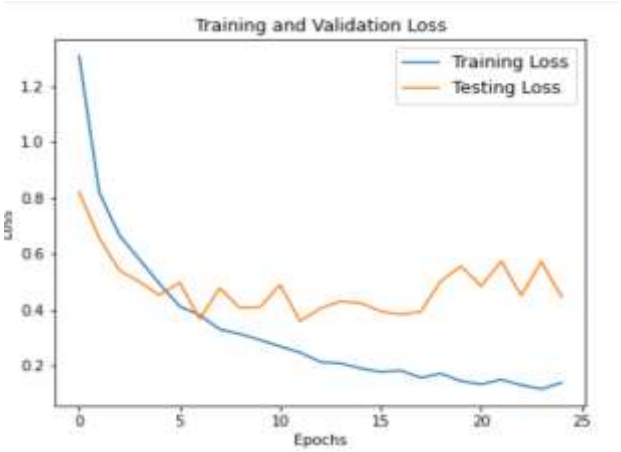

Figure 5: Graphical representation for Training and Validation Loss (Convolutional Neural Network)

The model's accuracy is evaluated through loss, which represents the disparity between actual and expected values. The loss function refers to the specific method employed to calculate this difference. It's worth noting that different loss functions may yield distinct values for the same loss. Moreover, variations in loss correspond to fluctuations in the performance of the corresponding model. From figure 5, under CNN, the graph indicated that the training loss and the validation loss continue to decline toward zero. This is good indication for the model training, once our target is to have zero error if possible. We also notice that the testing loss curve fluctuated for some time.

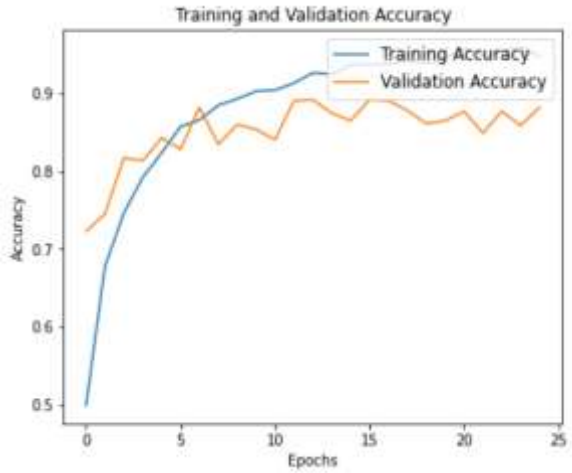

Figure 6: Graphical representation for Training and Validation Accuracy (Convolutional Neural Network)

A measure of accuracy is the difference between our prediction and the actual value; in other words, it is the degree to which our predictions match the real values. The network's loss function, often referred to as the objective function, evaluates accuracy by comparing the network's predictions with the true or expected target values. These expected values represent the desired outcomes from the network. The loss function then calculates a score indicating the extent to which the network performed relative to these targets. As depicted in Figure 6, the training and validation accuracy curves demonstrate consistently high performance, affirming the efficiency of our model.

 https://doi.org/10.59568/JASIC-2024-5-1-03

Table 1 summary behavior of the CNN as indicated in figure 5 and figure 6

	loss	accuracy	val_loss	val_accuracy
0	1.307840	0.498461	0.821401	0.723039
1	0.820596	0.677912	0.657307	0.745098
2	0.667660	0.746280	0.541760	0.817402
3	0.580296	0.792329	0.500250	0.813725
4	0.493697	0.823884	0.453565	0.843137

5. Conclusion

Mobile deep learning models based on images play a crucial role in diverse applications, including facial recognition, medical image segmentation, and biological identification (Karthikeyan, N. 2022). Conventional image classification models necessitate manual development of feature extraction techniques, resulting in time-consuming processes and lesser accuracy (Loddo, A., et al., 2021). Deep learning models, particularly convolutional neural networks (CNNs), have demonstrated effectiveness in extracting image features, filtering low-weight features, and achieving enhanced classification speed and accuracy. (Alem, A., & Kumar, S. 2022). Research conducted by Mercaldo, F., & Santone, A. (2020) demonstrated the ability of these models to rapidly analyze and interpret remote sensing images for tasks such as land cover and land use classification, environmental monitoring, agricultural decision-making, and urban planning. Moreover, deep learning models have been used to discriminate between malicious and legitimate samples in mobile environments, improving the effectiveness of anti-malware technologies. (Machiraju, H., et al., 2023) Moreover, there have been advancements in developing deep learning models to bolster the resilience of models against a broader spectrum of real-world image corruptions, thus overcome the shortcomings of the current image corruption datasets. This study discussed deep learning models in the context of image classification, mobile malware detection and vulnerability analysis.

The study highlighted the limitations of current approaches and proposed alternative solutions to enhance model robustness and accuracy. This study also emphasized the advantages of deep learning-based models, such as effective feature extraction, low-weight features filtering, and higher classification speed and accuracy. Future research directions suggested by these papers include the need for more comprehensive approaches to handle real-world image corruptions, alternative solutions to current anti-malware technologies, assessing and contrasting deep learning models for classifying remote-sensed images, as well as investigating parameter adjustments to enhance model efficacy. Overall, these papers contribute to the understanding of deep learning models and provide insights into potential research directions for further advancements in the field.

6. References

[1] Karthikeyan, N. (2022). Review of deep transfer learning models for image classification. *Int. J. Recent Contrib. Eng. Sci. IT (iJES)*, *10*(01), 17-28.

[2] Alem, A., & Kumar, S. (2022). Deep Learning Models for Remote Sensed Hyperspectral Image Classification. In *2022 13th International Conference on Computing Communication and Networking Technologies (ICCCNT)* (pp. 1-7). IEEE.

[3] Mercaldo, F., & Santone, A. (2020). Deep learning for image-based mobile malware detection. *Journal of Computer Virology and Hacking Techniques*, *16*(2), 157-171.

[4] Gupta, K., & Bajaj, V. (2023). Deep learning models-based CT-scan image classification for automated

 https://doi.org/10.59568/JASIC-2024-5-1-03

screening of COVID-19. *Biomedical Signal Processing and Control*, *80*, 104268.

[5] Józsa, T. (2020). The relevance of mobile applications helping in doctor–patient relationships. *British Journal of General Practice*, *70*(692), 109-109.

[6] Vega-Ramírez, L., Notario, R. O., & Ávalos-Ramos, M. A. (2020). The relevance of mobile applications in the learning of physical education. *Education Sciences*, *10*(11), 329.

[7] Ahmadzadegan, M. H., Izadyar, M., Deilami, H. A., & Ghorbani, H. (2020). Detailed Study on the Features of Mobile Applications. In *2020 International Conference on Electronics and Sustainable Communication Systems (ICESC)* (pp. 901-907). IEEE.

[8] Loddo, A., Loddo, M., & Di Ruberto, C. (2021). A novel deep learning based approach for seed image classification and retrieval. *Computers and Electronics in Agriculture*, *187*, 106269.

[9] Alem, A., & Kumar, S. (2022). Deep Learning Models Performance Evaluations for Remote Sensed Image Classification. *IEEE Access*, *10*, 111784-111793.

[10] Machiraju, H., Herzog, M. H., & Frossard, P. (2023). Frequency-Based Vulnerability Analysis of Deep Learning Models against Image Corruptions. *arXiv preprint arXiv:2306.07178*.

 https://doi.org/10.59568/JASIC-2024-5-1-03