Malaria cell classification based on CNN and Transfer Learning using slide images

Md Younus Ahamed
Lane Department of Computer Science and Electrical Engineering
West Virginia University, Morgantown, USA

***Abstract*—Malaria is a disease caused by a plasmodium parasite transmitted by the bite of infected female mosquitoes of the genus Anopheles. According to World Health Organization, more than 400,000 deaths occur due to Malaria. Malaria parasite destroys red blood cells, which may cause jaundice and anemia. If it is not detected early and treated with care, it can become severe and ultimately fatal. The major problem here is only medical experts can interpret these. Humanmade classification or detection of these diseases is tedious and remarkably prone to error. There is a limited number of medical experts, and the results may differ from person to person based on experience. We can use Machine Learning (ML) or Deep Learning (DL) to overcome this problem. In this work, we proposed a deep learning-based malaria classification—the malaria cell dataset used to train a custom CNN model and the VGG16 model. Then compared, the results from these two models. The CNN gave 95% test accuracy where as VGG16 reached 93% test accuracy. Then trained, the same CNN model was with another dataset to see how it performed on a different task. For the new dataset, the model reached 91% test accuracy.**

***Index Terms*—Malaria, Deep Learning, CNN, Transfer Learning, VGG16.**

I. Introduction

The Healthcare sector is different from other industries. It is a high-priority sector, and people expect the highest care and services regardless of cost [15]. Malaria is caused by protozoan parasites of the genus Plasmodium that are transmitted through the bites of infected female Anopheles mosquitoes and infect the red blood cells [14]. Africa is the most tropical of all continents, and Malaria is a severe problem in the tropical regions. According to the World Malaria Report 2016, an estimated 3.2 billion people in 95 countries and territories are at risk of being infected with Malaria and developing the disease, and 1.2 billion are at high-risk [13]. Fig. 1 depicts the worldwide death rate caused by Malaria. According to the World Health Organization, approximately 438,000 deaths resulted from 214 million infections in 2015 [2]. In most cases, the only available method of malaria diagnosis is a manual examination of the microscopic slide [10]. Detection of this disease is very subjective. Medical experts can only do it. It is a critical problem, especially in rural areas where finding an expert pathologist is challenging. Nowadays, a vast amount of image data is available. Deep learning, a state-of-the-art machine learning approach, has shown outstanding performance over traditional machine learning in identifying intricate structures in complex high-dimensional data, especially in computer vision [8]. In this work, a Convolutional Neural Network (CNN) was proposed to detect Malaria. The VGG16 model was used for transfer learning, and the result of CNN and VGG16 was compared for classification. The Dataset was publicly available, and it was collected from Kaggle. The Dataset contained two balanced classes (infected and uninfected). It was a binary classification problem. A deep CNN was trained with the cell images. The performance of a deep neural network depends on many hyperparameters. For CNN, the list of significant hyperparameters was the number of hidden layers, number of epochs, batch sizes, kernel size, number of kernels, activation function, optimizer, learning rate, etc. A hyperparameter tuning was done to find the optimal hyperparameters for the model. In transfer learning, a pre-trained model was reused as starting point for a model on a new task. A pre-trained model like VGG16 was trained on the ImageNet Dataset. The pre-trained layers were used in transfer learning, but the output layer was changed. All the weight for the pre-trained network was used for the new classification task. Loss, accuracy, sensitivity, and specificity were used as performance metrics.

II. Related works

Numerous recent studies have been conducted to understand Malaria disease detection in a better way. It is a fatal disease and a critical problem in rural areas due to the lack of expert pathologists. Deep learning can solve this expert human resources problem with automated detection of malaria disease. Jdey et al., 2022 discovered the potential of deep learning techniques as intelligent tools with a broader applicability for malaria detection, which benefits physicians by assisting in diagnosing the condition [7]. Kalkan et al., 2019 proposed an image processing-based Malaria detection system that showed an outstanding classification rate with 97% training accuracy and 95% test accuracy [9]. Shekar et al., 2020 proposed a new and highly robust machine learning model based on a convolutional neural network (CNN) that automatically classifies and predicts infected cells. Three types of CNN models were compared based on their accuracy Basic CNN, VGG-19 Frozen CNN, and VGG-19 Fine Tuned CNN . Dong et al., 2017 used three well-known convolutional neural networks, including the [17] LeNet, AlexNet, and GoogLeNet. It showed that all these deep convolution neural networks achieved classification accuracies of over 95%, higher than the accuracy of about 92% attainable by using the support vector machine method [3]. Narayanan et al., 2019 proposed an architecture for classifying cell images that achieved 96.7% testing accuracy and area under the Receiver Operating Characteristic (ROC) curve value of 0.994 [12]. Hung et al., 2017 used a Faster Region-based Convolutional Neural Network (Faster R-CNN), one of the top-performing object detection models in recent years, pre-trained on ImageNet but fine-tuned with their data [5]. Liang et al., 2016 proposed a 16-layer CNN model that achieved 97.37% accuracy. A transfer learning model only reached 91.99% on the same images [11]. Gourisaria et al., 2020 used a CNN algorithm on the microscopic image of the malaria-infected blood cells to predict if an organism is suffering from Malaria. The proposed model got an accuracy of 95.23% [4]. Kumar et al., 2020 presented a CNN model with 96.62% accuracy.

III. Dataset

A. Malaria cell dataset

The malaria cell images Dataset is taken from the official NIH Website. The Dataset has a total of 27,558 cell images divided into two classes. Infected and uninfected malaria cells. These were RGB images with three color channels.

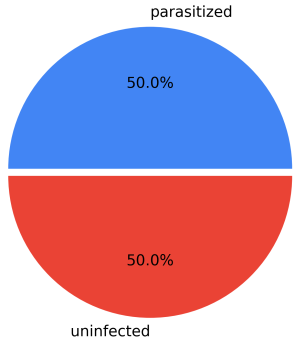

Fig. 1: Class distribution Malaria dataset

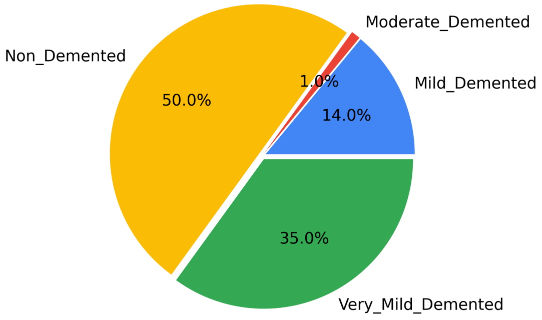

Fig. 2: Class distribution of Alzheimer's Dataset.

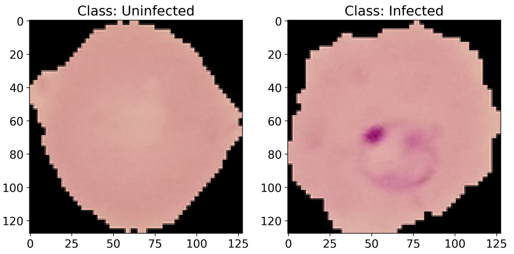

Fig. 3: Sample image from Malaria dataset

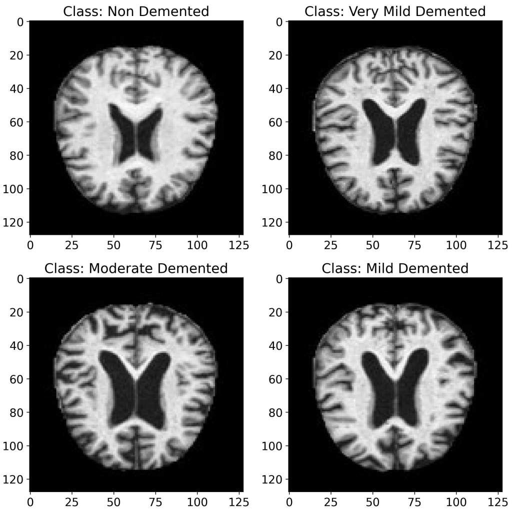

Fig. 4: Sample image from Alzheimer's Dataset

B. Alzheimer's Dataset

The Alzheimer's Dataset was collected from Kaggle. The Dataset consisted of 6400 Preprocessed MRI (Magnetic Resonance Imaging) Images divided into four classes. The classes and the number of images in each category for both dataset are presented in Table I and Table II. The size of the image was 128*128. Fig 3, Fig 4 depicted images from each class in both Dataset.

1) SMOTE: From Fig. 1, we can see the Moderate Demented class has only 1% image of the total size, and the Mild Demented type has only 14% of the entire Dataset. It was a clear case of an imbalance classification problem. A balanced multiclass classification was done by oversampling the minority class. A popular over-sampling technique called SMOTE(Synthetic Minority Over-sampling Technique) was used to solve the imbalance problem.

TABLE I: Classes in Malaria Dataset

Class	Number of images
Infected	13779
Uninfected	13779

Synthetic samples were generated from the minority class using SMOTE. This oversampling caused a new data sample but did not give any new variation or information in the model. After using SMOTE, the performance improved remarkably.

TABLE II: Classes in Alzheimer's Dataset

Class	Number of images
Non Demented	3200
Mild Demented	896
Very Mild Demented	2240
Moderate Demented	64

Before oversampling, the accuracy was around 65%. After using SMOTE, we got an accuracy of 91%.

C. *Train test split*

Both Datasets were split into a train test set. There was 80% data in the train set and 20% in the test set. The model was trained using the train set and validated using the validation set. Then the trained model was tested on the test set. A learning curve was drawn between training and testing datasets. For train test split, a function called train test split() from Python's open-source sci-kit-learn library was used [1].

IV. Convolutional Neural Networks (CNN)

Artificial neural networks (ANNs), non-linear models inspired by the brain's neural architecture, were developed to model the learning capacity of biological neural systems [16]. Convolutional Neural Networks (CNN) have dominated machine vision in recent years [6]. The CNN architecture is depicted in Fig. 5. Just like the MLP, CNN also consists of one input layer, one or more hidden layers, and one output layer. But in CNN, the hidden layers are convolutional layers instead of fully connected layers. Convolutional layers are the primary building block of CNN. This work used a combination of convolutional and separable convolutional layers. Separable convolution is a variation of traditional convolution. It's more efficient and computes faster than conventional convolution. It performs a depthwise spatial convolution followed by a pointwise convolution. A max pooling layer and batch normalization followed most convolutional and separable layers. The pool size for each max pooling layer was (2*2). The first hidden convolutional layer had 16 kernels of size (3*3) the next convolutional layer had 32 kernels of size (3*3). The first group of separable convolutional layers had 64 kernels, the next group had 128 kernels, and the final group had 256 kernels. The size of the kernels for these layers was (3*3). Each filter convolved with the image and created feature maps. Max pooling layers selected the maximum component from the area of feature maps. After convolution, there were three fully connected hidden layers with 512,128 and 64 nodes. Each dense layer was followed by batch normalization and dropout with a random dropout rate. The rectified linear activation unit (ReLU) was used in each layer. It is a popular activation function in the deep learning domain. It is an activation function that returns the value itself if it's positive or returns 0 if the input is negative.

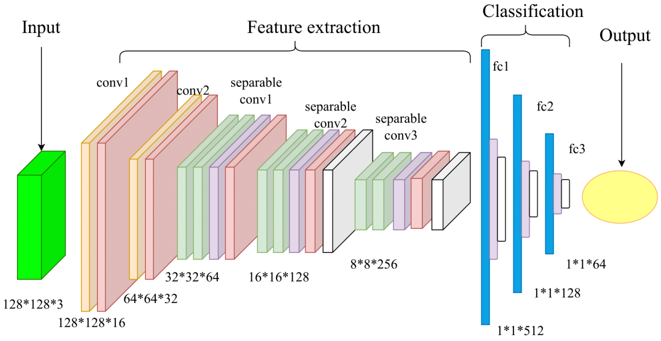

Fig. 5: The CNN architecture

$$f(x) = max(0, x) \tag{1}$$

Eq.(1) represented the ReLU activation function. ReLU is more computationally efficient than another popular activation function, sigmoid. It helps the model to prevent the vanishing gradient problem. This function only activates some of the hidden units at the same time. It is a binary classification problem. The output layer had one node with the sigmoid activation.

$$\Phi(x) = \frac{1}{1 + e^{-x}} \tag{2}$$

Eq.(2) represented the sigmoid activation function. The sigmoid function produces similar results to the step function in that the output is between 0 and 1. If the sigmoid neuron's output is larger than or equal to 0.5, it outputs 1; if the outcome is smaller than 0.5, it outputs 0. The CNN was compiled with Adam optimizer, binary cross-entropy loss, and performance metrics. Adam is a stochastic gradient descent method to find the optimal weights during the backpropagation and the optimal learning rate. The loss function compares each of the predicted probabilities to the actual class output. After building and compiling the model, it was trained on the training dataset and validated using the validation dataset. The number of epochs was 50.

A. *Hyperparameter tuning*

Hyperparameter tuning: A deep learning model has some learnable parameters like bias and weights. A model can learn their parameters by training. But some parameters are not learnable, called hyperparameters. The performance of a model immensely depends on hyperparameter tuning. In this work, the CNN was trained multiple times with different hyperparameters to find the optimal number of hyperparameters. The CNN model was trained with varying numbers of hidden layers, different activation functions (ex., tanh, sigmoid, and relu, etc.), different optimizers (Adam, AdaDelta, Adagrad, and RMSprop)batch sizes, and the number of epochs, a different number of kernel and kernel sizes. A list of hyperparameters is shown in Table III.

B. *Precaution for overfitting and vanishing gradient problem*

To prevent overfitting, a callback was used for training the models. This callback monitored the validation loss. The training was stopped if the validation loss was not improving. The dropout layers with random dropout rates also helped the model from being overfitted by turning off some percentage of

TABLE III: List of hyperparameters

Hyperparameters	CNN
Optimizer	Adam
Batch size	128
Epoch	50
Loss function	Binary cross entropy
Kernel size	(3*3)

hidden nodes. Another callback was used to adjust the learning rate. This callback also monitored the validation loss. The learning rate was readjusted if the validation loss needed to be improved. In deep neural networks vanishing gradient is a common problem. The ReLU activation function can take care of the vanishing gradient problem. But on top of that, I used batch normalization and Xavier weight initialization.

V. Transfer Learning, VGG16

In transfer learning, a pre-trained mode is reused for a new task. The knowledge gained from a job is used to solve a different task. The pre-trained model is used as a starting point for a model in another study. First, a pre-trained model was chosen (VGG16). Two options were available when working with transfer learning: download the network weights or train the model from scratch. In this work, the weights were downloaded. VGG16 had more neurons in the final output layer than required. The final output layer was removed and used as an output layer required for this 2-class classification. The starting layers of the pre-trained model were frozen. VGG16 model was trained on the ImageNet Dataset for 1000 classes. However, this classification problem had only two categories.

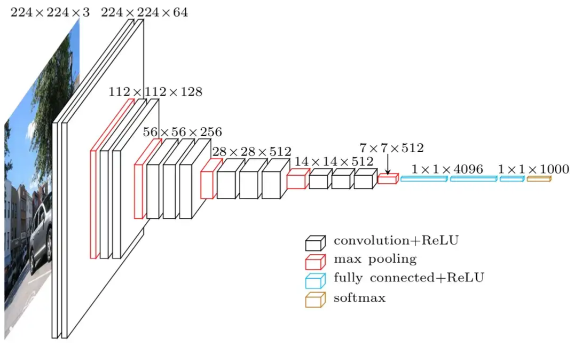

Fig. 6: Pre-trained VGG16 architecture

VI. Results

A. Malaria cell classification

This proposed system aims to detect Malaria disease with CNN and Transfer Learning and compare the performance of each approach. The experimental environment was developed using Keras, a Python Deep Learning library. Fig. 7 represents the learning curve of training vs. validation accuracy for CNN. Here an early stopping callback occurred. The training stopped after over 15 epochs to prevent over-fitting. The training vs. validation loss score for CNN is shown in Fig. 8. CNN is best suited for image classification. CNN gave a better performance than the VGG16 model. The CNN model for this classification task achieved 95% test accuracy with a loss score of 0.14. Fig. 9, and Fig. 10 represented the training vs. validation accuracy score and loss of the VGG16 model, respectively. The VGG16 model achieved 93 % test accuracy with a loss score of 0.18. Fig. 11 and Fig. 12 represent the confusion matrix for CNN and VGG16 models, respectively. The confusion matrix depicts the true positive, false positive, true negative, and false negative values.

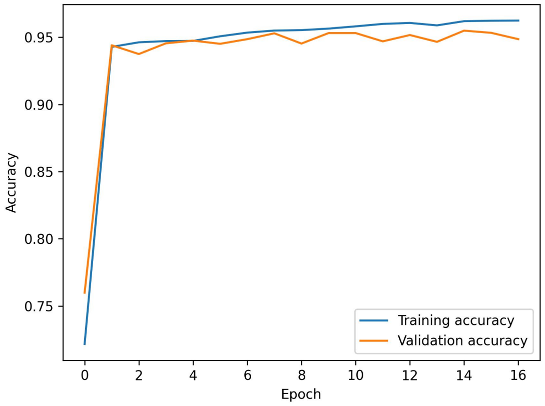

Fig. 7: Accuracy score for CNN of Malaria Dataset

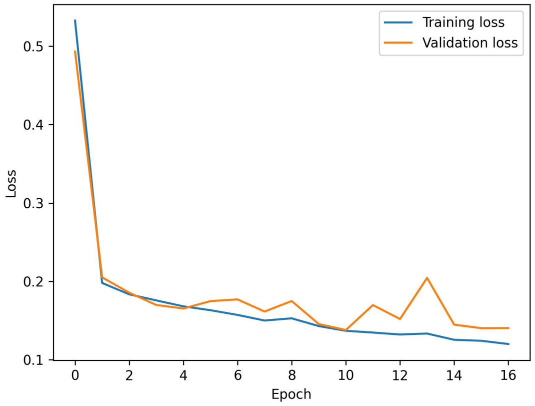

Fig. 8: Loss score for CNN of Malaria Dataset

$$Sensitivity = \frac{TP}{TP + FN} \tag{3}$$

$$Specificity = \frac{TN}{TN + FP} \tag{4}$$

For medical image classification, sensitivity and specificity can be two reasonable performance measures. Sensitivity means all the infected cells and the proportion of those actually tested infected. The CNN model has a sensitivity of 97.28%, and the VGG16 model has a sensitivity of 91.16%. Specificity means all uninfected cells and what proportion of those actually

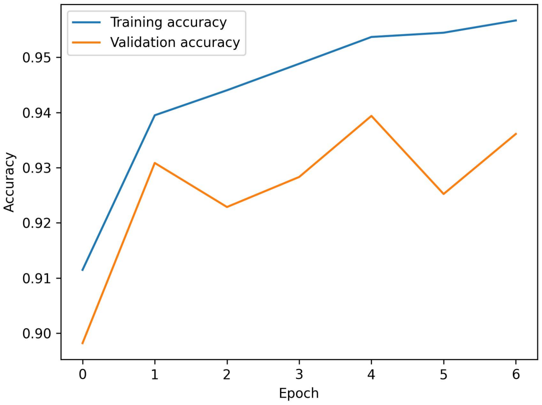

Fig. 9: Accuracy score for VGG16 of Malaria Dataset

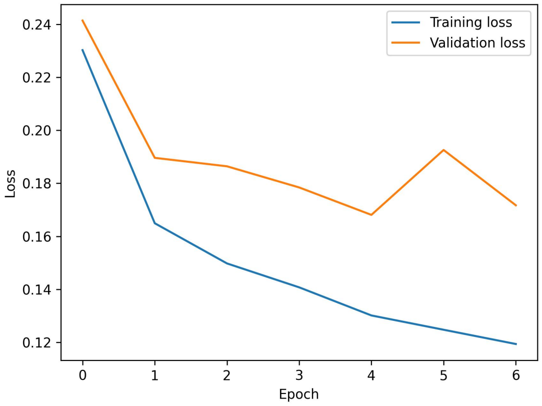

Fig. 10: Loss score for VGG16 of Malaria Dataset

test uninfected. The CNN model has a sensitivity of 92.65%, and the VGG16 model has a sensitivity of 95.27%. Higher sensitivity means a lower number of false negative values, and higher specificity means a lower number of false positive values.

TABLE IV: Comparison of CNN and Transfer Learning

Classifier	Accuracy	Loss	Sensitivity	Specificity
CNN	95%	0.14	97.28%	92.65%
VGG16	93%	0.18	91.16%	95.27%

B. Alzheimer's classification

To see how the model performs with a different task, I train the model with a new dataset. The Alzheimer's Dataset has four classes; hence it is a multiclass classification. The model for classifying Malaria is also used for this classification. Only the output layer is changed. The output layer has four nodes with a softmax activation function. The result here was also satisfying. The CNN model for Alzheimer's classification achieved 91% test accuracy with a loss score of 0.26. Fig. 13 represents the training vs. validation accuracy. The training vs. validation loss score for CNN is shown in Fig. 14.

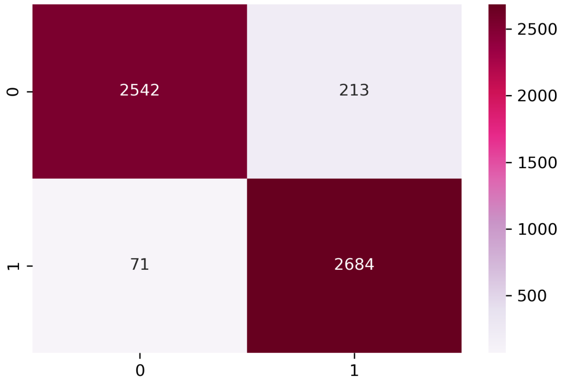

Fig. 11: Confusion matrix of CNN model

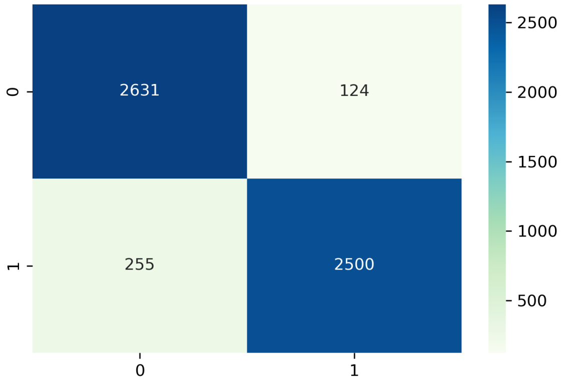

Fig. 12: Confusion matrix of VGG16 model

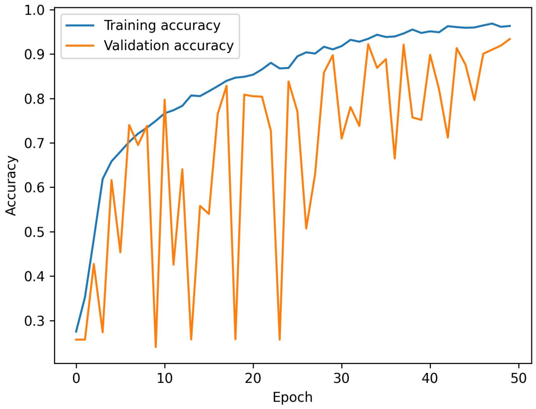

Fig. 13: Accuracy score for CNN of Alzheimer's Dataset

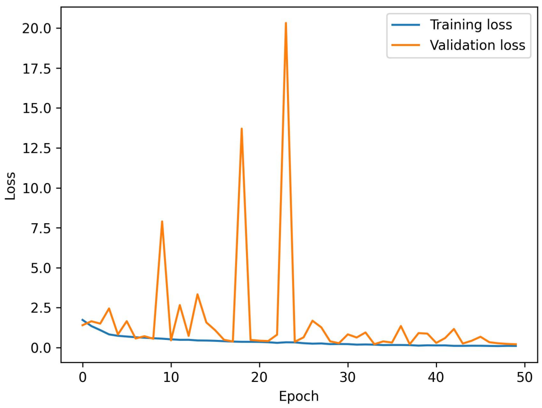

Fig. 14: Loss score for CNN of Alzheimer's Dataset

TABLE V: Comparison of Malaria and Alzheimer's classification

Dataset	Accuracy	Loss
Malaria Dataset	95%	0.14
Alzheimer's Dataset	91%	0.26

VII. Conclusion and Future Work

I have started my deep learning journey by doing this project. I learned how to build Multi-layer perceptrons, Convolution Neural Networks, and Transfer learning models. I had exposure to many topics for building MLP and CNN. Activation function, kernel, convolution, hyperparameter tuning, vanishing, and exploding gradient problem, dropout and batch normalization, and many more. It was the first time I was writing the entire report in LaTeX. I learned how to write an essay in LaTeX. The number of samples is crucial for deep learning. In future work, we will train the model with a large dataset. Here in this work, we used the weights of the pre-trained networks, which may be why the pre-trained models did not work better than the custom CNN. In future work, we will build those models (ex., VGG16) from scratch for this specific task.

References

[1] Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, et al. Api design for machine learning software: experiences from the scikit-learn project. *arXiv preprint arXiv:1309.0238*, 2013.

[2] Margaret Chan. World malaria report. *Geneva: World Health Organization*, 2015.

[3] Yuhang Dong, Zhuocheng Jiang, Hongda Shen, W David Pan, Lance A Williams, Vishnu VB Reddy, William H Benjamin, and Allen W Bryan. Evaluations of deep convolutional neural networks for automatic identification of malaria infected cells. In *2017 IEEE EMBS international conference on biomedical & health informatics (BHI)*, pages 101–104. IEEE, 2017.

[4] Mahendra Kumar Gourisaria, Sujay Das, Ritesh Sharma, Siddharth Swarup Rautaray, and Manjusha Pandey. A deep learning model for malaria disease detection and analysis using deep convolutional neural networks. *International Journal of Emerging Technologies*, 11(2):699–704, 2020.

[5] Jane Hung and Anne Carpenter. Applying faster r-cnn for object detection on malaria images. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 56–61, 2017.

[6] Mahbub Hussain, Jordan J Bird, and Diego R Faria. A study on cnn transfer learning for image classification. In *UK Workshop on computational Intelligence*, pages 191–202. Springer, 2018.

[7] Imen Jdey, Ghazala Hcini, and Hela Ltifi. Deep learning and machine learning for malaria detection: overview, challenges and future directions. *arXiv preprint arXiv:2209.13292*, 2022.

[8] Taeho Jo, Kwangsik Nho, and Andrew J Saykin. Deep learning in alzheimer's disease: diagnostic classification and prognostic prediction using neuroimaging data. *Frontiers in aging neuroscience*, 11:220, 2019.

[9] Soner Can Kalkan and Ozgur Koray Sahingoz. Deep learning based classification of malaria from slide images. In *2019 scientific meeting on electrical-electronics & biomedical engineering and computer science (EBBT)*, pages 1–4. IEEE, 2019.

[10] MM Kettelhut, PL Chiodini, H Edwards, and A Moody. External quality assessment schemes raise standards: evidence from the ukneqas parasitology subschemes. *Journal of clinical pathology*, 56(12):927–932, 2003.

[11] Zhaohui Liang, Andrew Powell, Ilker Ersoy, Mahdieh Poostchi, Kamolrat Silamut, Kannappan Palaniappan, Peng Guo, Md Amir Hossain, Antani Sameer, Richard James Maude, et al. Cnn-based image analysis for malaria diagnosis. In *2016 IEEE international conference on bioinformatics and biomedicine (BIBM)*, pages 493–496. IEEE, 2016.

[12] Barath Narayanan Narayanan, Redha Ali, and Russell C Hardie. Performance analysis of machine learning and deep learning architectures for malaria detection on cell images. In *Applications of Machine Learning*, volume 11139, pages 240–247. SPIE, 2019.

[13] World Health Organization. *Malaria microscopy quality assurance manual-version 2*. World Health Organization, 2016.

[14] Mahdieh Poostchi, Kamolrat Silamut, Richard J Maude, Stefan Jaeger, and George Thoma. Image analysis and machine learning for detecting malaria. *Translational Research*, 194:36–55, 2018.

[15] Muhammad Imran Razzak. Saeeda naz, and ahmad zaib. deep learning for medical image processing: Overview, challenges and the future. *Classification in BioApps*, 26:323–350, 2018.

[16] Linda M Salchenberger, E Mine Cinar, and Nicholas A Lash. Neural networks: A new tool for predicting thrift failures. *Decision Sciences*, 23(4):899–916, 1992.

[17] Gautham Shekar, S Revathy, and Ediga Karthick Goud. Malaria detection using deep learning. In *2020 4th international conference on trends in electronics and informatics (ICOEI)(48184)*, pages 746–750. IEEE, 2020.