Malaria Cell Detection using Convolutional Neural Network

A Deep Learning Term Project Report

Afifatul Mukaroh[1], Naufal Suryanto[2], and Harashta Tatimma Larasati[3]
{afifatul.mukaroh, naufalsuryanto, harashta}@pusan.ac.kr

I. INTRODUCTION

Malaria is an infectious and sometimes lethal disease caused by a parasite that commonly infects a certain type of mosquito which feeds on humans [1]. This disease is responsible for more than 600,000 deaths worldwide each year, with symptoms include fever, chills, body aches, vomiting, and fatigue. Infected patients become weak and may experience severe anemia, painful headaches, and organ failure [2].

Malaria, which is induced by *Plasmodium falciparum* in protozoa, can affect the blood cells in humans. Malaria can modify the red blood cells and exports its own protein, causing deformities that will lead to the various mentioned symptoms. Blood smear, a test used to look for abnormalities in blood cells [3], can be utilized to detect this disease.

Neural networks are a class of algorithms loosely modeled on connections between neurons in the brain [4]. This approach has been very popular in the machine learning era and has given a substantial enhancement in various kinds of fields. Convolutional Neural Network (CNN) is a specialized kind in deep neural network for processing data that has a known grid-like topology. The network employs a mathematical operation called *convolution*, which is a specialized kind of linear operation [5]. CNN is considered very successful in image recognition and classification.

In this study, we implement Convolutional Neural Network (CNN) to distinguish malaria-infected cells from the uninfected ones. We refer to the CNN architecture presented on [6]. The result is then evaluated by using confusion matrix with accuracy value is 0.8, recall value is 0.96, Precision value is 0.64, and F-measure value 0.77.

II. DATASET AND METHODOLOGY

A. Dataset

In this project, we utilize Malaria dataset from the National Institutes of Health (NIH), U.S. National Library of Medicine [7]. The repository consists of images of segmented cells from the thin blood smear images from the Malaria Screener research activity. The dataset contains a total of 27,554 cell images with equal instances of parasitized and uninfected cells. Figure 1 and Figure 2 show the overview of uninfected and infected cell images.

On splitting the dataset, we follow the recommended data partitioning methods. We split the datasets into the proportion of 70%:15%:15% for training, testing, and validating purpose, respectively, since it is said to be commonly suggested in the field of machine learning [8]. Note that in this paper, testing data refers to the portion of data used to generate the model while validation data refers to the portion of data used for evaluating the model.

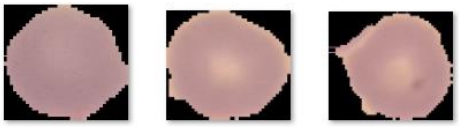

Figure 1 Overview of Uninfected Malaria Cell Dataset

Figure 2 Overview of Infected Malaria Cell Dataset

B. Methodology

In order to distinguish malaria-infected cells from the uninfected ones, there are three steps performed as follows:

1. *Preprocessing:* in this step, we perform transformations such as rescaling and image flipping to the raw data before they are fed into CNN algorithm. For our case, we implement four stages of preprocessing: rescaling, horizontal flipping, and adjusting shear (intensity) range and zoom range. This is done to enhance classification performance and to speed up the training process [9].
2. *CNN training and testing:* CNN algorithm is used to train this dataset and generate the CNN model. The architecture implemented is referred from [6], which leverages a series of convolution and max pooling layer. The detail of implemented CNN architecture is as shown on Figure 3.
3. *Validation*: After the CNN model is acquired, validation is conducted to evaluate the performance of the model.

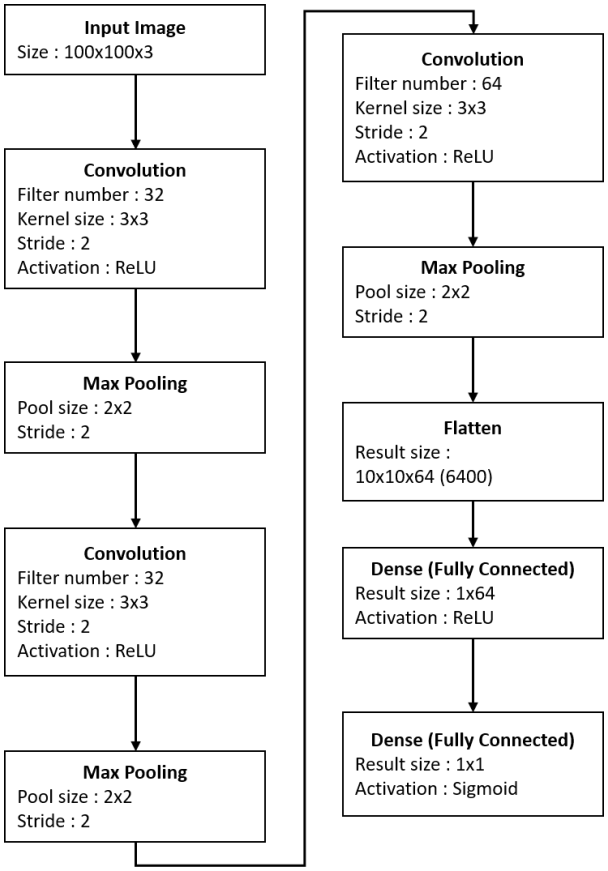

Figure 3 Implemented CNN Architecture

C. Preprocessing

Preprocessing is the general term of performing transformation to the data before feeding them into the model. This may include centering, normalization, shift, rotation, shear, etc. Generally, there are two purposes of preprocessing. The first one is to clean up the data from noise (also called artifacts). This will make the learning process easier for the model. Secondly, preprocessing is done to augment the data in the case of insufficient number of datasets. Small dataset size might cause the deep model to learn sufficiently well. Augmenting, which is the process of transforming each data sample in many possible ways and then adding all the augmented samples to the dataset. Hence, the effective size of the dataset can be increased [10].

D. Convolutional Neural Networks (CNN)

Convolutional Neural Networks (CNN) is a branch of artificial neural networks which is inspired by visual cortex of human leverages the concept of convolution and pooling. CNN can extract features without omitting the spatial correlations of the input [11], hence very suitable to be used for image recognition and image classification.

CNN typically consists of convolutional layers, pooling layer, and fully connected layers. Training the datasets is done by applying it to a series of convolutional layer, filters (also known as kernels), and pooling layer, and then to the fully connected layer for image classification.

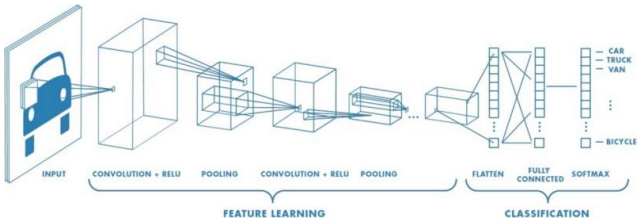

Figure 4 CNN Process Overview [12]

III. EXPERIMENT SETUP

In terms of the environment setup for this project, we use Desktop Computer to process the classification algorithm. The following is the summary of hardware used, the designation of team roles, and the overall contribution for this project.

1. Environment

- Hardware: desktop computer
- CPU: Intel(R) Core (TM) i7-7700 CPU @ 3.60GHz
- RAM: 16GB
- Software Environment: Python3
- Data and Algorithm:
  - Training set: 19.290 data
  - Test set: 4132 data
  - Validation set: 4132 data

2. Role Designation and Overall Contribution

In the implementation, all team members are involved in every stage of the project. However, the main responsibility of each member and the overall contribution for this project is as stated on Table 1.

Table 1 Role Designation and Overall Contribution of Team Member

No	Name	Role	Overall Contribution (%)
1	Naufal	Preprocessing	30
2	Afifatul	Training and Testing	40
3	Harashta	Validation	30

IV. EXPERIMENT RESULT AND ANALYSIS

For preprocessing, we perform 4 steps: rescaling, sheer range, zoom range, and horizontal flipping. Rescaling is to normalize the input value, which is the pixel of the image for this instance. All the pixel of the images is divided with the value of 255 to make it fall into 0 to 1 range value. Sheer range, zoom range, and horizontal flip are for data augmentation purpose. Shear range is shearing the angle in counterclockwise direction in degrees. The degree that we implement here is 0.2. Zoom range is randomly zoom the input. Horizontal flip is randomly flipping inputs horizontally. Finally, all the data is resized into 100x100 to be fed as the input.

For CNN training, we employ Adam for the optimization algorithm and Cross Entropy function to calculate loss. The

total epoch is 50, with total step per epoch equals the total training data divided by 10 (1,929). Hence, there are a total of 96,450 iterations in this training. From the total of 27,554 data, we split it into 70% for training (19.290), and 15% each for testing (4132), and validating (4132).

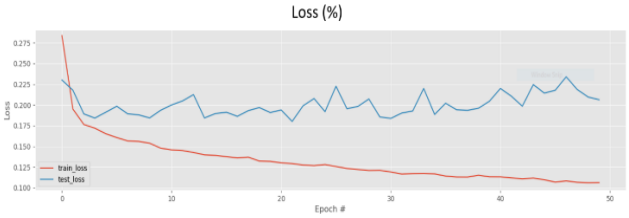

Figure 5 Training Curve based on Loss

Figure 5 and 6 show the comparison of training and testing result. Figure 5 represents the loss (also termed as error) while Figure 6 depicts the accuracy. As shown in the Figure 5, at the first epoch times, the value of training and testing error is quite similar. As the epoch time increases, the training error becomes lower, which means the model can learn sufficiently well. However, the testing error is not lowered. In fact, it is slightly increasing. This is the sign of overfitting, which is the condition where the gap between the training error and testing error is too large [13].

This overfitting condition may be caused by several reasons. Firstly, the capacity of the model, which is the model's capability to fit a wide variety of functions. The high-capacity model can experience overfitting for memorizing too many properties of the training set, which in turn would not serve well on the test set [13].

Secondly, this high capacity also corresponds to the high complexity. In the implemented scenario, the CNN architecture applied is quite complex. Our hypothesis is that the overfit occurs because we apply a sophisticated method to solve a rather simple task, which may be able to be solved with a more simplistic way. It is in accordance with Occam's razor or principle of parsimony [13], which stated that we should choose the simplest hypothesis among others.

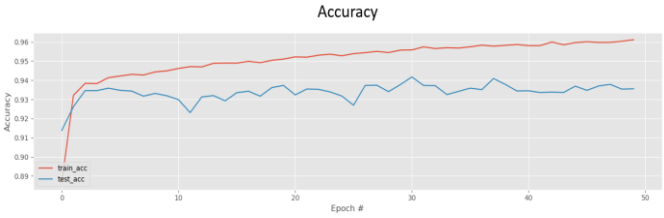

Figure 6 Training Curve based on Accuracy

To evaluate this model, we use confusion matrix. Table 2 shows the confusion matrix of our model. Based on this matrix, we can obtain that the accuracy value is 0.8, recall value is 0.96, Precision value is 0.64, and F-measure value 0.77.

Table 2 Confusion Matrix of the Implemented CNN

		Actual Values	
		Infected	**Uninfected**
Predicted Values	**Infected**	1326	740
	Uninfected	2018	48

V. Conclusion

Convolutional Neural Networks (CNN) is a branch of Neural Networks that can extract features from sets of images without eliminating the spatial correlations, which is proven to be very suitable for the case of image recognition and image classification. In this project, we implement CNN algorithm to classify malaria-infected and uninfected blood cells. We perform 4 steps of preprocessing: rescale, horizontal flip, sheer range and zoom range before continuing to the CNN algorithm implementation. From the experiment, it is concluded that the implemented Malaria-infected and uninfected cells using this architecture has precision value equals 0.64 and accuracy of 0.8.

.

VI. References

[1] C. f. D. C. a. P. (CDC), "Malaria," [Online]. Available: https://www.cdc.gov/malaria/about/disease.html. [Accessed 28 5 2019].

[2] I. A. Lobo, "How the Malaria Parasite Remodels and Takes Over a Human Host Cell," [Online]. Available: https://www.nature.com/scitable/nated/topicpage/how-the-malaria-parasite-remodels-and-takes-132627374.

[3] Healthline, "Blood Smear," [Online]. Available: https://www.healthline.com/health/blood-smear.

[4] N. Networks. [Online]. Available: https://www.sciencedirect.com/topics/neuroscience/neural-networks.

[5] C. Networks. [Online]. Available: https://www.deeplearningbook.org/contents/convnets.html.

[6] S. Rajaraman and e. al, "Pre-trained convolutional neural networks as feature extractors toward improved malaria parasite detection in thin blood smear images," *the Journal of Life and Environmental Sciences*, 2018.

[7] "Malaria Datasets," [Online]. Available: https://ceb.nlm.nih.gov/repositories/malaria-datasets/.

[8] "Researchgate," [Online]. Available: https://www.researchgate.net/post/Is_there_an_ideal_ratio_between_a_training_set_and_validation_set_Which_trade-off_would_you_suggest.

[9] handrienj, "https://www.freecodecamp.org/news/https-medium-com-hadrienj-preprocessing-for-deep-learning-9e2b9c75165c/," [Online].

[10] I. S. D. Zone. [Online]. Available: https://software.intel.com/en-us/articles/hands-on-ai-part-14-image-data-preprocessing-and-augmentation.

[11] Y. e. a. Dong, "Evaluations of deep convolutional neural networks for automatic identification of malaria infected cells," in *2017 IEEE EMBS International Conference on Biomedical & Health Informatics (BHI)*, 2017.

[12] [Online]. Available: https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148.

[13] I. Goodfellow, Y. Bengio and A. Courville, Deep Learning, http://www.deeplearningbook.org, 2016.

[14] B. Aksasse. [Online]. Available: https://www.researchgate.net/figure/K-fold-cross-validation-In-addition-we-outline-an-overview-of-the-different-metrics-used_fig2_326866871.