Deep Learning for Malaria Parasite Detection

Luciana da Costa Marques
School of Electrical and Computer Engineering
University of São Paulo
São Paulo, Brazil
Email: luciana.marques@usp.br

Celso Oviedo da Silva
Instituto de Matemática e Estatística (IME),
Universidade de São Paulo (USP)
São Paulo, Brazil
Email: celsovislo@gmail.com

Mauro César C. Morais
Instituto do Câncer do Estado de São Paulo (ICESP),
Faculdade de Medicina
Universidade de São Paulo (FMUSP)
Email: mauro_morais@usp.br

***Abstract*—Malaria is a blood disease caused by the *Plasmodium* parasites. The gold standard diagnostic method is bright-field microscopy images in red blood cells smears. This task can be time consuming and error prone. This report presents the identification of the *Plasmodium sp.* parasites in images of red blood cells by using deep learning. We applied a convolutional neural netwrok (CNN) ResNet with TensorFlow backend to classify between infected and non-infected red blood cells images. Our results showed that the use of CNN can be a promising tool in the malaria diagnostic.**

1. Introduction

Malaria is a mosquito borne-infectious disease, caused by a single-cell microorganism of the *Plasmodium* species. It is transmitted mainly by the female individuals of the *Anopheles* mosquitoes. Because of it's endemic episodes and consequent health risks, it is relevant to research for detection and prevention methods.

Malaria is a disease which is present in certain regions that have the following in common:

1) High poverty level;
2) Poor access to health care;
3) Density of the disease's transmission vector.

The last cause leads to the conclusion that tropical areas are more susceptible to Malaria, due to the presence of *Anopheles* mosquito be higher in warmer weather locations. Also, once a human is infected with it, if he or she is bitten by a not infected mosquito, this mosquito will then carry the parasite, consequently increasing the number of infected people in nearby areas. Since the more affected regions usually lack of a proper health system or it is inaccessible to most of the its population, endemic episodes of Malaria have a high probability in such areas.

1.1. Malaria Parasites Detection

Despite of the development of polymerase chain reaction (PCR) for the detection of malaria parasites, it is of limited use [1]. The gold-standard for malaria detection is the image analysis of red blood cells. A patient's blood is smeared on a glass slide and stained with a contrasting agent that facilitates identification of parasites within red blood cells [2]. A trained clinician examines 20 microscopic fields of view at 100 X magnification, counting red blood cells that contain the parasite out of 5,000 cells. Manually counting this cells is a repeated and tedious task, which can create staff burden and lead to errors. Therefore, a process automation with machine learning can be useful to avoid both staff burden and classification errors [3].

1.2. Image Segmentation using Deep Learning

Once there is an infection, the parasite enters the patient's red blood cells. Because of this, it is possible to detect the infection by analysing the images of these cells. A typical infected red blood cell will contain the parasite as a round dense coloured area in part of its borders as in figures 1 and 2. Therefore, this can be treated as an image segmentation and classification problem.

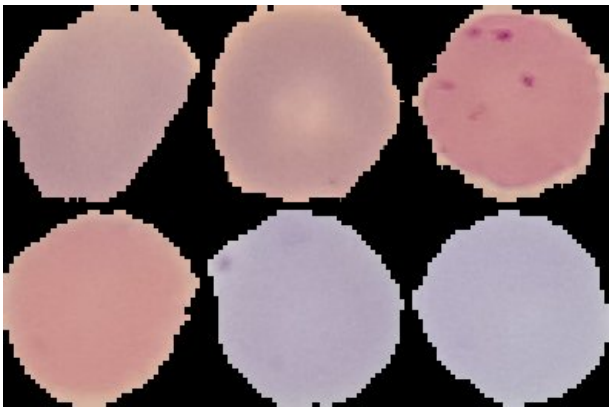

Figure 1. Examples of uninfected cells

There are several image segmentation techniques available. Deep learning is a technique which received attention in the past years for its relevant performance in segmentation and classification problems. It has also gained notoriety with hardware advancements and GPU implementations [4].

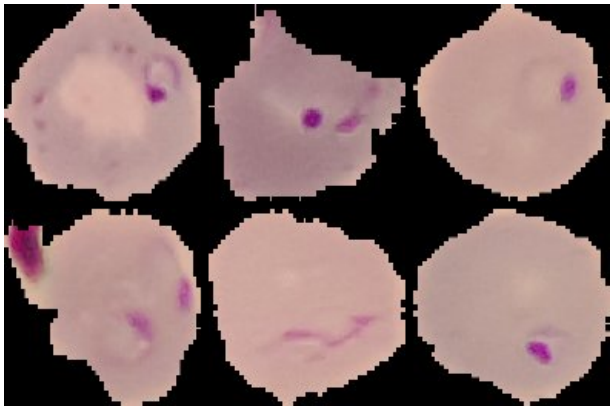

Figure 2. Examples of parasitized cells

2. Materials and Methods

The technique chosen for segmentation and classification was the Convolutional Neural Networks, more specifically the ResNet architecture [5]. The data set used was formed by the images of cells obtained from 50 healthy patients and 150 patients infected with *P. falciparum*. The data set is publicly available at the official NIH webpage [6].

2.1. Data set preparation

The data set used consisted of 13.780 images of infected and uninfected cells. For proper model training and validation, the data set was split accordingly to the following:

1) Training images set: 72% of total images;
2) Validation images set: 8% of total images;
3) Testing images set: 20% of all images.

Initially, the whole set was split in 20% for testing and 80% for training, and then 10% of the later into the validation set.

The data set was also reshaped with a step of data augmentation by randomly shifting, translating, and flipping each training sample. It is quite simple to perform data augmentation using Keras [7], as in the following example extracted from its official website:

```
# initialize the training training data
    augmentation object
trainAug = ImageDataGenerator(
  rescale=1 / 255.0,
  rotation_range=20,
  zoom_range=0.05,
  width_shift_range=0.05,
  height_shift_range=0.05,
  shear_range=0.05,
  horizontal_flip=True,
  fill_mode="nearest")

# initialize the training generator
trainGen = trainAug.flow_from_directory(
  config.TRAIN_PATH,
  class_mode="categorical",
  target_size=(64, 64),
  color_mode="rgb",
  shuffle=True,
  batch_size=BS)
```

The `ImageDataGenerator` object allows real time image generation and can later be used in the model's training, validation and testing. More examples can be found in the reference page.

2.2. The ResNet architecture

The ResNet architecture was proposed with intentions of reducing learning complexity in deep learning neural nets. As bigger the number of layer is, usually it becomes harder to train the given model. Apart from that, as in the original article in which this architecture was proposed, "when deeper networks are able to start converging, a degradation problem has been exposed: with the network depth increasing, accuracy gets saturated (which might be unsurprising) and then degrades rapidly. Unexpectedly, such degradation is not caused by overfitting, and adding more layers to a suitably deep model leads to higher training error (...). The degradation (of training accuracy) indicates that not all systems are similarly easy to optimize" [5]. A proposed solution to this problem is using residual learning.

2.2.1. Residual Learning. Residual learning architectures are based on feedforwarding alimentation. The main difference from an architecture that embbeds residual learning and a one that does not is that in the first case, instead of forcing the layer output to match a mapping, it induces the layers to fit a residual mapping. The main hypothesis is that it is easier to optimize the residual mapping instead of the mapping itself.

The feed forwarding is implemented by short cutting connections 3. In the ResNet architecture case, the shortcut connection simply performs an identity mapping (that is, it maps to the input itself) and their outputs are added to the layers' stack output. The layers are then reformulated with as residual functions with reference to the layer inputs, instead of learning unreferenced functions. Such residual networks become easier to optimize and hence gain accuracy with increased depth. [5].

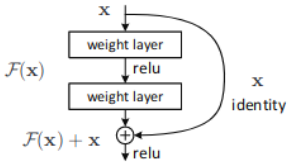

Figure 3. Residual learning and feedforwarding

2.2.2. Feedforwarding with short cutting connections. Considering $H(x)$ as an underlying mapping to be fit by a few stacked layers (not necessarily the entire net), with x denoting the inputs to the first of these layers. If multiple

nonlinear layers can approximate complex functions, then it is possible to assume that they can likewise approximate the residual functions, i.e., $H(x) - x$ (assuming that the input and output are of the same dimensions). So rather than expect stacked layers to approximate $H(x)$, let the layers approximate a residual function $F(x) := H(x) - x$. The original function thus becomes $F(x) + x$. Although both forms should be able to asymptotically approximate the desired functions (as hypothesized), the ease of learning is significantly different.

2.2.3. Description of layers. The network inputs are 64x64 pixel RGB images, and the output is of two classes, that is, infected or not infected. The first convolutional layer has 64 filters. Next, there are three residual modules with 32, 32 and 128 convolutional filters respectively. Finally, there are six stacks of residual modules, where each convolutional layer has 128, 128 and 512 filters.

2.3. Implementation

The main tools for implementation, both for the data set preparation and the model training and testing were:

1) The Keras deep learning framework [7];
2) Matplotlib, used for plotting the final results;
3) Numpy and Scikit-learn: numerical python packages;
4) cv2: Python opencv package for image processing.

These tools were chosen mainly because of its vast use in deep learning applications due to the increasing use of the python language for scientific purposes.

2.4. Training

The number of epochs initially chosen was 20, with total duration of $\approx$ 7 hours. Then, a second experiment was run for 50 epochs for results comparison, with total duration of $\approx$ 20 hours. The training was performed in CPU (Intel® Core™ i5-3470 CPU @ 3.20GHz; System memory 7791 Mb; Ubuntu 16.04.4 LTS Xenial). Apart from that, the input images were reshaped to be 64x64 pixels each to fit the input dimensions of the created network (each original image had different dimensions from each other).

2.5. Metrics

The evaluation metrics taken into account for model analysis were the following:

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

$$F_1 = 2 \cdot \frac{Precission \cdot Recall}{Precision + Recall} \tag{3}$$

with TP being the number of true positives, FP the number of false positives and FN the number of false negatives.

The precision metric evaluates the fraction of real infected cells among all that were classified as infected. The recall is the value of real infected cells classified correctly by the total number of infected cells, whereas the F-1 score measures the test's accuracy. The results and analysis for each metric value obtained during the testing phase is presented in the following section.

3. Results

After training the ResNet model, the results were obtained on the the prepared testing set. Here is also presented an overview of accuracy and training loss obtained during the training batches in both experiments.

3.1. Training Metrics Evolution

In figures 4 and 5 it is illustrated the evolution of accuracy and training loss through the training and validation batches. It is possible to see that accuracy is near to almost 1.0 after approximately 40 epochs, and training loss decreases as the model passes through batches (which is expected). The small difference between the training and validation data sets results indicates that the model is not overfitting.

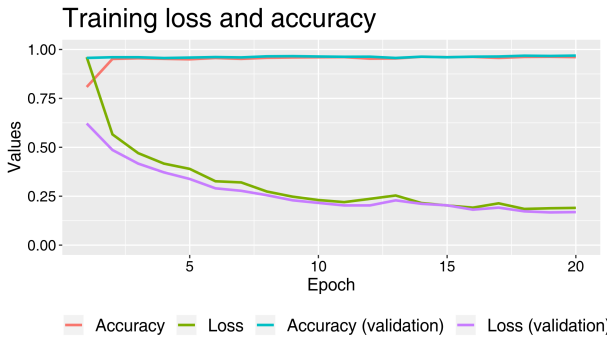

Figure 4. Accuracy and training loss values evolution for the first training experiment (20 epochs).

3.2. Testing results

The testing results for the first training are in 1 and the second one in 2. It shows better precision for parasited cells, but better Recall for uninfected cells. This same pattern is observed for both number of epochs sets (20 and 50), with the metrics values slightly bigger for 50 epochs.

TABLE 1. TEST RESULTS FOR THE FIRST TRAINED MODEL (20 EPOCHS)

	Precision	Recall	F1-Score	Support (n)
Parasitised	0.97	0.95	0.96	2756
Uninfected	0.95	0.98	0.96	2756

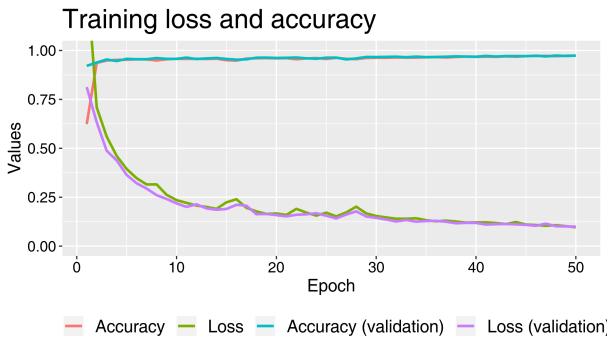

Figure 5. Accuracy and training loss values evolution for the second training experiment (50 epochs).

TABLE 2. TEST RESULTS FOR THE SECOND TRAINED MODEL (50 EPOCHS)

	Precision	Recall	F1-Score	Support (n)
Parasitised	0.97	0.96	0.96	2756
Uninfected	0.96	0.97	0.97	2756

4. Discussion

Here we presented the results of training and classification of red blood cells between infected and uninfected with *P. falciparum* parasites. The corss-validation at the patient level helps to avoid the overfitting in image classification and during the training stage assists in overfitting assessment [8]. ResNet Network presented good classification system for infected and uninfected cells.

We understand that we need a comparison with other CNN models to better evaluate the performance of this particular network. Other issues to be solved consists in improving the deployment of the trained model of the network and identify the optimal layer for feature extraction. Taken together, these results demonstrate the use of neural networks as a promising tool to help diagnose malaria.

MCCM: Alumas coisas que podemos fazer para melhorar o trabalho e publicar:
Podemos aumentar a precisão?
Quais as *layers* de melhor desempenho para extração de *features*?
Normalizar ou equalizar as cores antes de fazer novo treinamento melhora as métricas da rede?
Análise estatística comparando com o desempenho de outras redes? Quais?

References

[1] Hommelsheim CM, Frantzeskakis L, Huang M, Ülker B. PCR amplification of repetitive DNA: a limitation to genome editing technologies and many other applications. Scientific Reports. 2014 May;4:5052. Available from: https://www.nature.com/articles/srep05052.

[2] Brasil, Ministério da Saúde, Secretaria de Vigilância em Saúde. Manual de diagnóstico laboratorial da malária. Brasília: Editora MS; 2005. Available from: http://bvsms.saude.gov.br/bvs/publicacoes/malaria_diag_manual_final.pdf.

[3] Malaria Hero: A web app for faster malaria diagnosis;. Accessed: 2019-08-05. https://blog.insightdatascience.com/https-blog-insightdatascience-com-malaria-hero-a47d3d5fc4bb.

[4] Mihai-Sorin Badea LMFCV Iulian-IonutFelea. The Use of Deep Learning in Image Segmentation, Classification and Detection;Available from: https://arxiv.org/pdf/1605.09612.pdf.

[5] He K, Zhang X, Ren S, Sun J. Deep Residual Learning for Image Recognition. arXiv:151203385 [cs]. 2015 Dec;ArXiv: 1512.03385. Available from: http://arxiv.org/abs/1512.03385.

[6] Rajaraman S, Antani SK, Poostchi M, Silamut K, Hossain MA, Maude RJ, et al. Pre-trained convolutional neural networks as feature extractors toward improved malaria parasite detection in thin blood smear images. PeerJ. 2018;6:e4568.

[7] Keras: Deep Learning for Humans;. Accessed: 2019-08-05. https://keras.io/.

[8] Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research. 2014;15:1929–1958. Available from: http://jmlr.org/papers/v15/srivastava14a.html.