JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue

JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)

An International Scholarly Open Access, Peer-reviewed, Refereed Journal

A CLASSIFICATION OF HUMAN CELLULAR IMAGES AS 'PARASITIC' OR 'UNINFECTED' USING CONVOLUTIONAL NEURAL NETWORKS

[1]Elio Jordan Lopes, [2]Shaolin Kataria, [3]Shashank Keshav

[1]BTECH CSE Student, [2]BTECH IT Student, [3]BTECH CSE Student

[1]School of Computer Science and Engineering, Vellore Institute of Technology, Vellore, India

Abstract : Malaria is a serious infection transmitted by Plasmodium genus protozoan parasites. It is transferred among people primarily by the female Anopheles mosquito bites, and it mostly attacks red blood cells (RBCs) in human beings. Malaria kills the majority of children in Africa, with new incidents reported every other moment. As per the 2016 World Malaria Report, an around these three billion people across 100 countries are at greater risk of contracting Malaria, with a billion people at very high risk. Malaria claimed the lives of 450,000 people worldwide in 2016, with a projected 214 million people infected. More than two-thirds of all deaths were reported among children aged below 5 in Africa. Africa as a whole contributed to 92% of all deaths caused by Malaria. The symptoms caused by Malaria are painstaking, the likes of which can lead from anything like a fever or headache to something as fatal as death. To get an idea about the severity of the disease and proceed with the diagnosis, is imperative for the count of the parasite to be accurate. This will lead to better testing and measuring of drugs resistance and effectiveness. The process of diagnosis is not standardized in all places of the world, and in some low-resource settings and countries, lagging technological advances hinge on the repertoire of the microscopists who work in isolation. This frequently leads to inaccurate outcomes as a result of poor microscopic assessment decisions. As a result of the misleading consultations, people are often given small concentrations or antibiotics drugs that are otherwise unwarranted, resulting in potential adverse effects such as diarrhoea, nausea, and abdominal pain. As a result, automating the diagnosis process will allow for accurate disease diagnosis and, as a result, will provide dependable universal healthcare to resource-scarce locations. Malaria diagnosis has been computerised using a variety of machine-learning algorithms. This paper details my recent achievement in using deep convolutional neural networks to categorise malaria-infected cells. The presented scheme goes over the methods for developing an image dataset given by the National Library of Medicine better abbreviated as (NIH). There has been a list of augmentation approaches for data that have been used to enhance the dataset in place to avert the problem of overfitting. The dataset was trained, verified, and tested on a variety of patterns to evaluate the performance of the Convolutional Neural Network in the use-case of classification. Several autoencoders once intrinsically and extrinsically interpolated the resulting set of data.

***IndexTerms* - Malaria, RBC, Machine Learning, Data Augmentation, Convolutional Neural Network, Autoencoder.**

I. INTRODUCTION

Plasmodium Falciparum, Vivax, Malariae, Ovale, and Knowlesi are the five Plasmodium types that induce Malaria in humans. P. Falciparum and P. Vivax are the two most prevalent species. P. Falciparum is the most dangerous form of malaria, causing the majority of deaths worldwide every year.

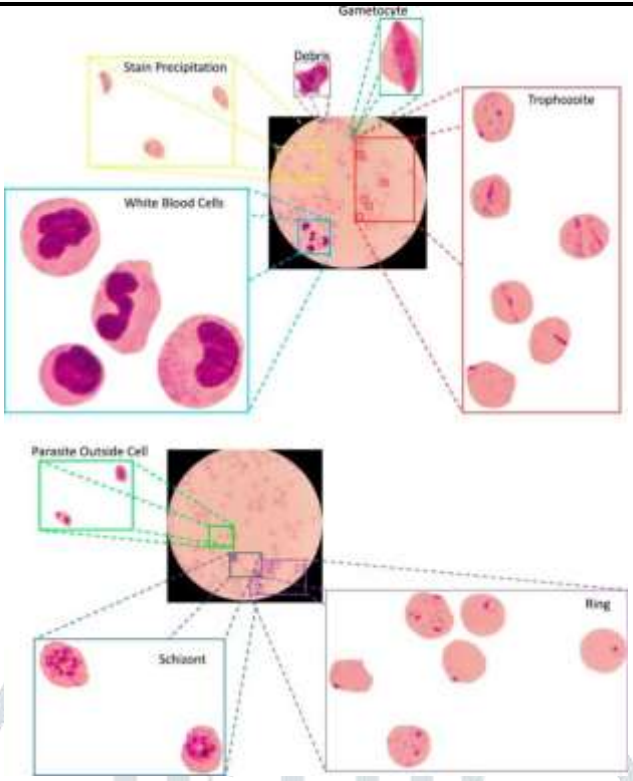


Fig.1. Parasite Stages.

Figure 1 displays two views of a blood slide with various parasite stages in the very same image. Trophozoites and Gametocytes (larger nucleus comparatively) can be observed alongside white blood cells(WBCs) in the first image. Falciparum ring phases and schiznoits and furthermore, staining disturbances and parasites outside the cells can be seen. Due to incompetence or human error, an untrained microscopist may mistake the staining disturbances for parasite diseases.

Although a lot of work has been progressing in this area, there's really no specific vaccine for malaria yet. Because malaria is a highly infectious disease, early detection thrrough timely screening of the pathogen is crucial to avoid the condition escalating to severe cerebral infections due to malaria, which can have a devastating effect on the body's cognitive state. The most widely used detection measures include fluorescence light microscopy and rapid diagnostic tests (RDTs), as well as a few alternative approaches which are not so well adapted to isolated malaria areas.

In terms of reducing costs, simplicity of use, and parasite screening, the existing diagnostic measures have a lot of opportunity for enhancement. Furthermore, the accuracy of the diagnosis is dependent on the dexterity of the person analysing the blood smears and also the amount of parasitic organisms present. Also while there are larger number of specimen, the examination's monotonicity has a significant impact on the quality of the examination. Furthermore, the worldwide scarcity of pathologists has a significant influence on poor country medical systems in general, and malaria will be no exception. Many inhabitants of underdeveloped nations seek treatment abroad due to a complete lack of trustworthy or non-existent diagnostic techniques, posing a challenge.of economic feasibility for most of the poor population.

Machine learning techniques are utilized in the stride assist human microbiologists and microscopists in making the accurate diagnosis, across the whole world.

II. Background

The research conducted shows classical machine learning algorithms used in the process of classification of infected and unifected cells. Some algorithms like Support Vector Machines, K-Nearest neighbors and Bayesian learning have given accurate outcomes.

A lot of work is going on in Removing the impulse noise in the and stains from the blood smear samples. The research involves use of SVMs, KNNS, and logistics regressions in categorization of samples. Additionally, measures have been undertaken to eliminate smears and excessive noise from central blood smears. Thresholding is amongst the most significant picture pre-processing steps in the classification task, as suggested by Mustafa et al [1]. CNNs have been used to locate infectious agent from microscopic pictures in multiple studies [2][3]. by Shen et. al [4]. Hongda Shen et al. [5] was using a dataset with approximately thousand training and testing data samples, and so applied a deep learning approach using transfer learning, as well as testing all the outcomes on a LeNet model of CNNs[6]. A group of six researchers created and evaluated a novel technique based on object recognition employing a quicker region based CNN , perfectly alright model, finely tuned and trained on INet[7] framework. To improve the clinical diagnosis, appropriate machine vision methodologies such as adaptive non-linear grayscale conversion, dataset augmentation followed by white balancing methods have been applied.

Because the information in picture collections of datasets that is frequently inadequate for categorization, other techniques such as transfer learning have been used in conjunction for an already taught Convoluted Neural network to aid machine assisted infection diagnosis. The suggested scheme will also demonstrate that sign transfer learning, which is a common approach in machine learning and vision based learning, may yield quite decent results in comparison to other standard ml algorithms that necessitate meticulous characteristic engineering and research followed by pipelining of data.

III. Proposed Model

The dataset we have used contains about 28000 images of pathogenic as well as uninfected human cells taken from a website.

The website contains images of human cells taken by using thin blood smear slides under a microscope. The microscopic images are photographed by the smartphone camera which is then used by the android application which is connected to the microscope.

200 P. falciparum-infected and 100 healthy patients were examined using Giemsa-stained blood smear slides under a microscope. Phone camera then took photographs of the microscopic view of the cells which was done by using the android application which was then manually annotated accordingly. The dataset has approx. 28000 images with the same quantity of infected and non infected images. First the proposeed system trains the model, with one image each from the infected and non-infected datasets as shown below in the figures.

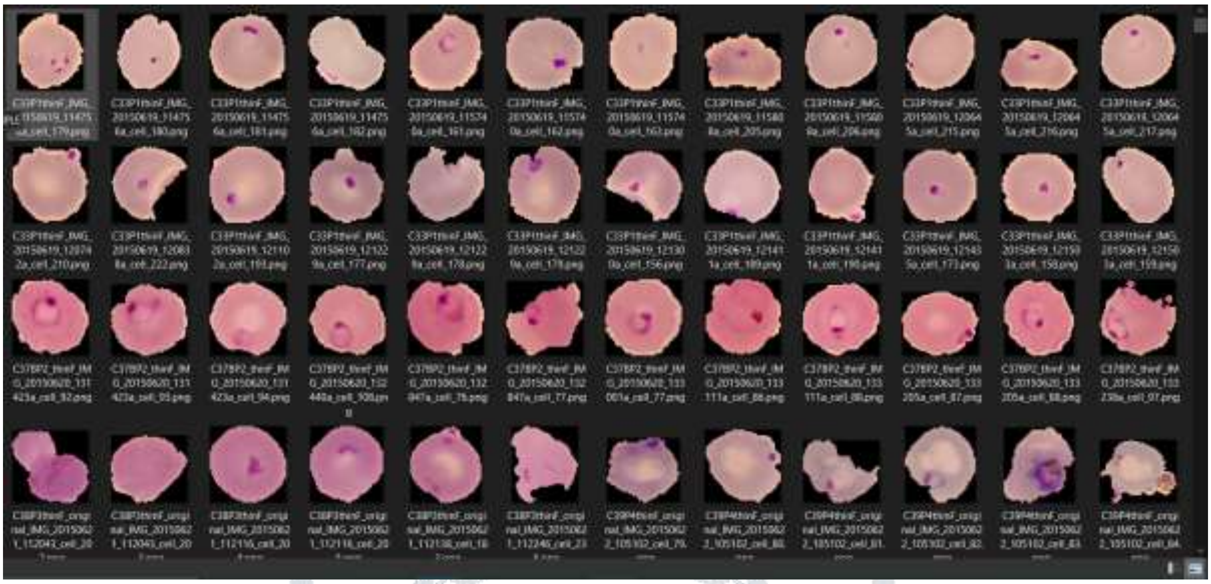


Fig.2. Infected or Parasitized Cells.

Fig 2 shows samples drawn from NIH Malaria dataset that are affected by various forms of malarial parasite. These also include stains that occur during the process of acquiring data. The most important aspect of working with image data is denoising the image by getting rid of the stains and impurities in an image which would otherwise cause errors or flaws in the accuracy in the outcome of the Convolutional Neural Network.

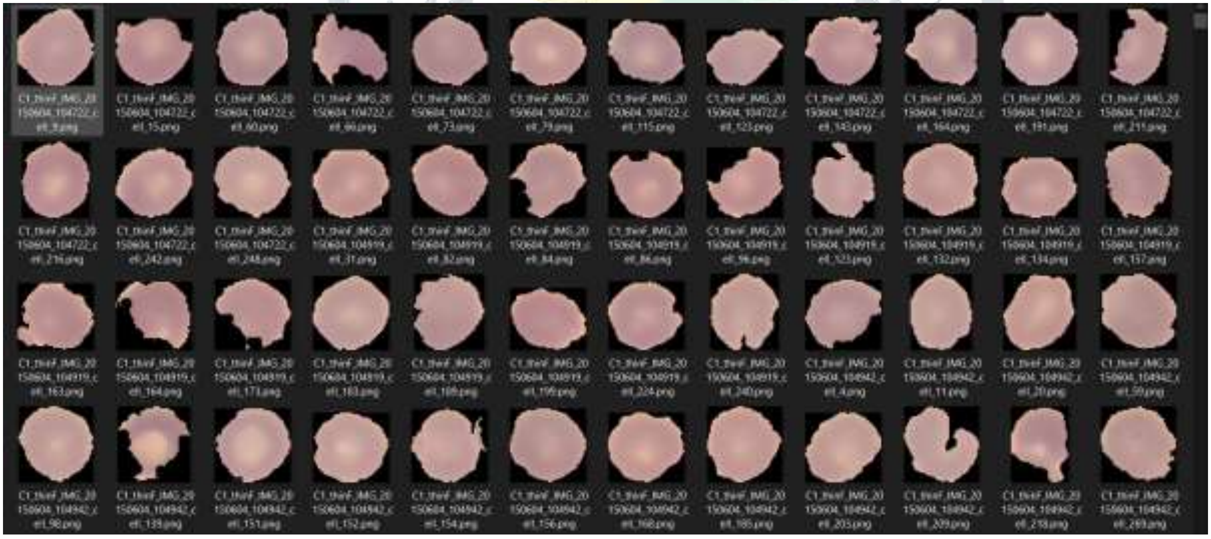
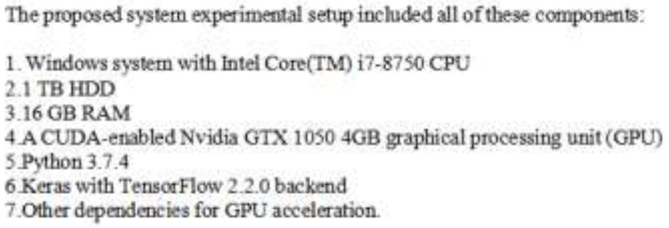


Fig.3. Uninfected Cells

Fig 3. Shows samples having uninfected RBCs from the NIH dataset and we can also infer that the images have different types of color distribution of the cells, caused by stains while conducting the process of data acquisition.



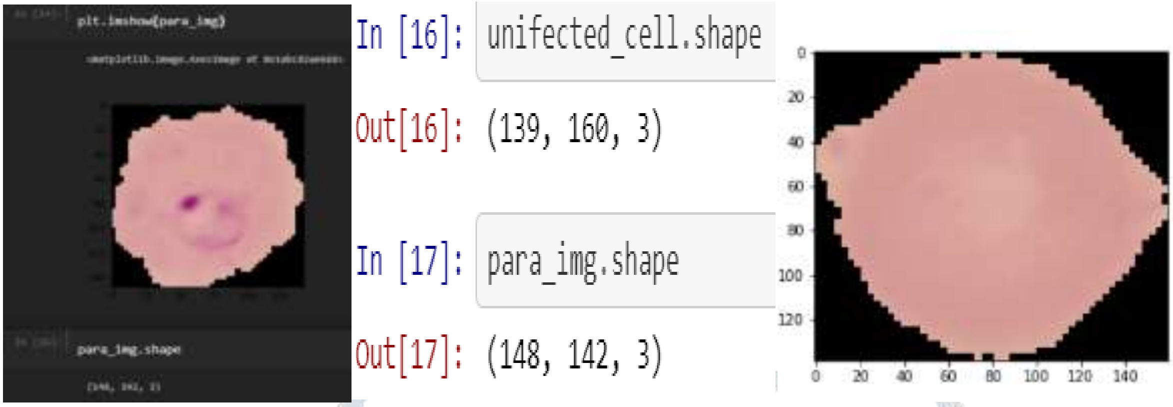
The proposed system experimental setup included all of these components:

1. Windows system with Intel Core(TM) i7-8750 CPU
2. 1 TB HDD
3. 16 GB RAM
4. A CUDA-enabled Nvidia GTX 1050 4GB graphical processing unit (GPU)
5. Python 3.7.4
6. Keras with TensorFlow 2.2.0 backend
7. Other dependencies for GPU acceleration.

Segmented patches of the images have been used as input and then we have applied deep learning for predicting malaria parasitic infected cells. After thorough research on the malarial human cells dataset, the proposed system has come up with three distinct approaches:

Moreover, numerous preprocessing and post-processing methods were devised and tests were carried out on a test set to obtain optimal performance. As a part of the proposed system, we also carried out a five-fold cross validation using 5 different test datasets and each time the proposed system had to be tested against almost 25000 training samples and 2800 test samples.

It has to be noted that the segmented patches of RBCs are of three- channels (RGB) and varying floating precision constitute 100-170 pixels. The output has been re-configured to fit a 200x200-pixel dimension and FP32 floating precision thus making the dataset suitable as the apt input requirements for distinct classification techniques and algorithms employed to construct the model.



```
plt.imshow(para_img)

para_img.shape
(148, 142, 3)
```

```
In [16]: unifected_cell.shape
Out[16]: (139, 160, 3)

In [17]: para_img.shape
Out[17]: (148, 142, 3)
```

Fig.4. shows the dimensions of the image using the .shape() python function. It displays the result as (148, 142, 3) which shows a 3dimensional RGB image of 148x142 pixel size. Also the unifected cell is labelled as (139, 160, 3).

Evaluation metrics used to evaluate performance of the model: Accuracy, Loss, validation loss, validation accuracy, f1 score and precision.

The data in the dataset was divided into training, validation and test datasets containing images and data in the ratio 80:10:10. The training set found 24958 images belonging to 2 classes (infected and uninfected). The architecture was checked by the technique of 3-fold cross validation. A k value of 3 was chosen for cross validation upon repeated testing as this value returns neither high bias nor high variance. Obviously it goes without saying that the images in the real world scenario are very different from the training dataset so we deploy out imaging algorithm accordingly.

The images have different sizes and are given an average dimension by navigating through the array of images and taking dimension of each and every image and taking their average.

The proposed system has used some inbuilt functions in Keras to automatically process the data, thus generating a flow of batches from the directory having the dataset and also manipulating the images.

Data pre-processing and augmentation :

Since, different types of films are used having giemsa or wright stains or some other stains. Due to the concentration and staining method and pattern color variations in the images become an issue. So to handle so many color variations, a very complex algorithm is required which increases the error probability. So we have used standardization ton deal with this issue. All the blood films are stain normalized for standardization on the red blood cell smear patches using normalization.

These images are then manipulated using resizing, rotations and scaling so that the model becomes robust to more set of images if trained and which are not present in the dataset. The proposed system import the Keras library called ImageDataGenerator to do this. The following parameters were fed to the ImageDataGenerator Keras library to achieve this.

1. rotation_range=20, # rotate the image 20 degrees
2. width_shift_range=0.10, # Shift the image width by a max of 5%
3. height_shift_range=0.10, # Shift the image height by a max of 5%
4. rescale=1/255, # Rescale the image by normalizing it.
5. shear_range=0.1, # Shear means cutting away part of the image (max 10%)
6. zoom_range=0.1, # Zoom in by 10% max
7. horizontal_flip=True, # Allow horizontal flipping
8. fill_mode='nearest' # Fill in missing pixels with the nearest filled value

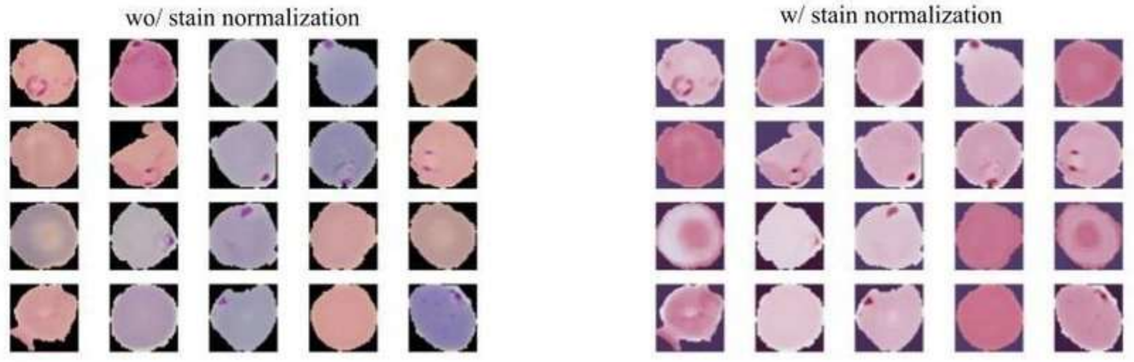


Fig.5.

Fig 5 shows images of RBCs with and without stains. Stains can cause errors in classifications and are usually removed my applying normalization techniques in deep learning. It is a process that alters the upper and lower limit of the values pertaining to pixel intensity.

Dataset augmentation is made possible by a series of processes that include both horizontal and vertical flips and shifting, altering of pixel intensity, rotation and gaussian blur.

To train the model, the proposed system imported the following libraries provided Tensorflow and Keras:

Sequential, Dense, activation, Dropout, Flatten, MaxPooling2D and Conv2D.

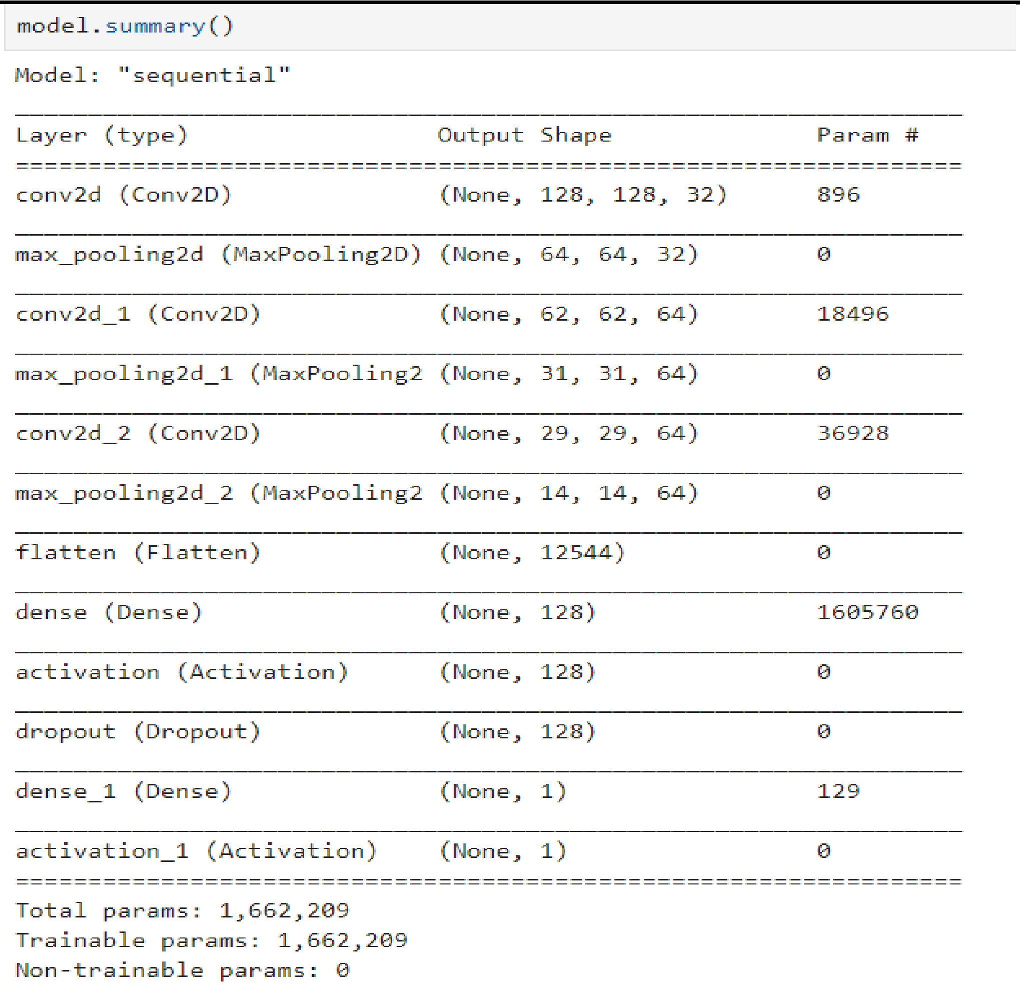
The architecture designed consists of a variety of layers, namely, 3 convolution and max pool, 2 dense and 1 flatness, 1 50% dropout layer and 1 fully connected layer.

Filters of size 3 x 3 are used in the convolutional operation with the first layer handling the initial input data and preprocessing and sending the data to the next layer as input. 2 x 2 pool size is used in maximum pooling layers, which further serves as the required input finally, for the flatness layer.

The CNN send input from the convolution layer to the dropout layer that discards 50% of the data. At last, sense layer generates output that acts as input to the activation layer. Softmax is used as an activation function here. Binary Cross Entropy is used as the loss fucntion for getting the error between the predicted output and the original output.

Adam optimizer is employed to give optimised weights and also gives biases using the concept of backpropagation. The activation function handles 64 data items at once with the weights randomized and the biases 0. This is continued for 20 epochs. Every block acts as an input to every other activation function using ReLu(Rectifier Linear Unit. Here the image shape is taken as 130px X 130px X 3(r,g,b). With these parameters and a kernel size of 3x3, and using the Rectifier Linear Unit (ReLU). Also a Pool size is mentioned to convert the 3x3 kernel here to a 2x2 kernel. The model is trained using a Dense network of 128 neurons.

A Dropout function was initiated for the model. This function helps in reducing overfitting any model, like for instance a random Dropout of 50% is added to turn half of the neurons OFF during model training. Since the layer is binary (Parasitic and Uninfected) I have used the Adam Optimizer for compiling the model.

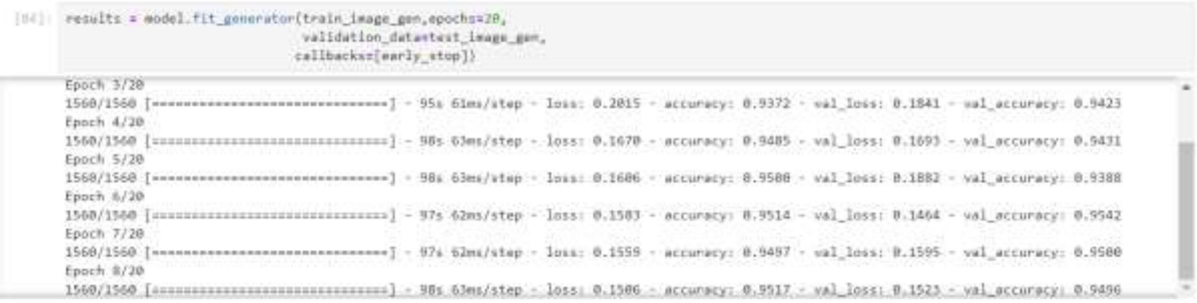


```
model.summary()

Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 128, 128, 32)      896
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 64, 64, 32)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 62, 62, 64)        18496
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 31, 31, 64)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 29, 29, 64)        36928
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 14, 14, 64)        0
_________________________________________________________________
flatten (Flatten)            (None, 12544)             0
_________________________________________________________________
dense (Dense)                (None, 128)               1605760
_________________________________________________________________
activation (Activation)      (None, 128)               0
_________________________________________________________________
dropout (Dropout)            (None, 128)               0
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 129
_________________________________________________________________
activation_1 (Activation)    (None, 1)                 0
=================================================================
Total params: 1,662,209
Trainable params: 1,662,209
Non-trainable params: 0
```

Fig.6.

From Fig 6, it can be noted that Data augmentation caused the initial training set of 27,558 images extends to 1662209 trainable parameters.



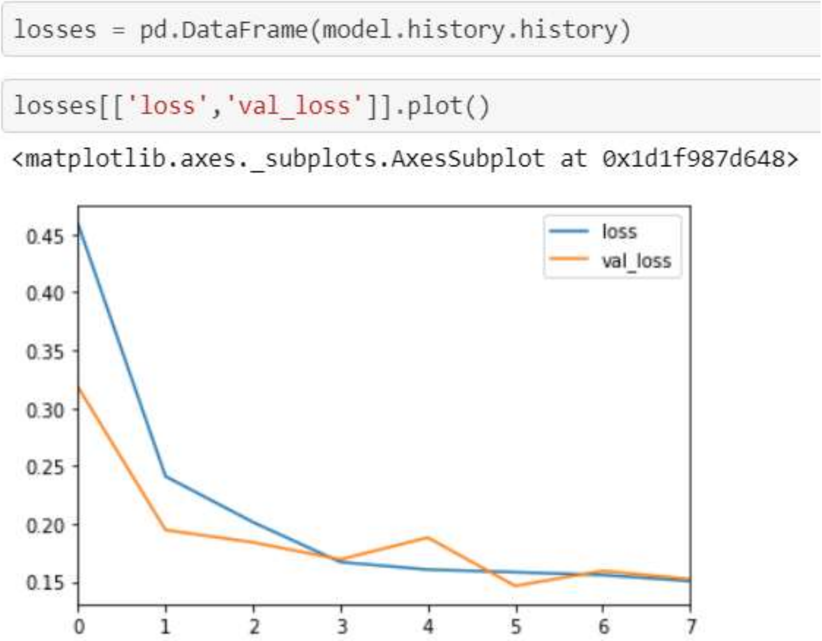
```
[84]: results = model.fit_generator(train_image_gen,epochs=20,
                                    validation_data=test_image_gen,
                                   callbacks=[early_stop])

Epoch 3/20
1560/1560 [==============================] - 95s 61ms/step - loss: 0.2015 - accuracy: 0.9372 - val_loss: 0.1841 - val_accuracy: 0.9423
Epoch 4/20
1560/1560 [==============================] - 98s 63ms/step - loss: 0.1670 - accuracy: 0.9485 - val_loss: 0.1693 - val_accuracy: 0.9431
Epoch 5/20
1560/1560 [==============================] - 98s 63ms/step - loss: 0.1606 - accuracy: 0.9500 - val_loss: 0.1882 - val_accuracy: 0.9388
Epoch 6/20
1560/1560 [==============================] - 97s 62ms/step - loss: 0.1583 - accuracy: 0.9514 - val_loss: 0.1464 - val_accuracy: 0.9542
Epoch 7/20
1560/1560 [==============================] - 97s 62ms/step - loss: 0.1559 - accuracy: 0.9497 - val_loss: 0.1595 - val_accuracy: 0.9500
Epoch 8/20
1560/1560 [==============================] - 98s 63ms/step - loss: 0.1506 - accuracy: 0.9517 - val_loss: 0.1523 - val_accuracy: 0.9496
```

Fig.7.

The model is trained using a sigmoid convolutional neural network, and is run for almost 9 epochs out of 20 epochs and is evaluated based on probabilities from a scale of zero to one. The loss and accuracy are evaluated based on my model and are calculated as 0.9528 ie; 95% accurate and the remaining as losses. The predicted probabilities > 0.5 are taken as FALSE(Parasitic) and then others as TRUE(uninfected) . An array of images showing cell imagery results as TRUE and FALSE is obtained, with binary values 1 and 0.

IV. Result Analysis and Discussion



```
losses = pd.DataFrame(model.history.history)
```

```
losses[['loss','val_loss']].plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1d1f987d648>
```

Fig. 8. The loss has been plotted for each epoch. The maximum validation loss can be observed to be 0.15

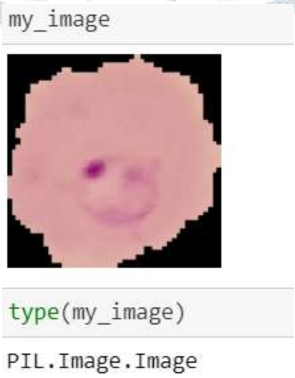
The training of the model is done for 30 epochs. A validation accuracy of 95% is yielded as the convergence of the model occurs within 10 epochs but the loss starts increasing thanks to overfitting, as the model continues to train.

Thus, the results are shown in the table below.

	precision	recall	F1-score	support
0	0.96	0.94	0.95	1300
1	0.94	0.96	0.95	1300
Accuracy			0.95	2600
Macro avg	0.95	0.95	0.95	2600
Weighted avg	0.95	0.95	0.95	2600

Table 1

The evaluation of the proposed system has been done by taking into various parameters within network architecture. The parameters constitute of the model Accuracy, further evaluated by its Loss and Precision, Recall, F1 score. The custom model trained predicted 50% cells as parasitic and 50% cells as uninfected with Boolean values 0 and 1 respectively.



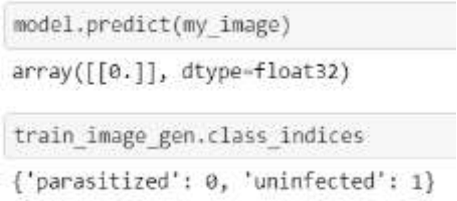
```
my_image
```

```
type(my_image)
```

```
PIL.Image.Image
```

Fig.9.

Fig 9 shows a sample image from the dataset that the proposed system has used for prediction using the model that trained. By examining the image we can find out that the RBC has been infected by a parasite(see pinkish color).



```
model.predict(my_image)

array([[0.]], dtype=float32)

train_image_gen.class_indices

{'parasitized': 0, 'uninfected': 1}
```

Fig.10.

Upon execution of the model.predict(my_image) function, we can infer that it returns an array have 0 boolean value of dtype as float32 and having class indices as ‘0’ for parasitized and ‘1’ for uninfected.

Thus the trained model has correctly predicted the probabilistic outcome and classified the test image as parasitized which is true in this case.

V. Conclusion

Convolutional layers are basically feature maps that are developed off developed feature detectors. The selection process of the significant feature detectors is done by training. This allows the network to perform the scanning and categorization of images better. The way this supercedes the experience of the naked eye is because the naked eye doesn’t offer as diverse detectors, so, the features detected by the network will be far advanced than that of a naked eye.

Here is a simple workflow that we used for classification of images as parasitized and uninfected.

Feature extraction:

1. Image are fed: Input.

2. Convolutional Theorem applied using RELU activation function.

3. Pooling is done to reduce the features in thousands of images in the dataset to a few features and the extracted features are further Flattened

Classification:

4. The convolutional layer must produce an output of a single dimension. This is a long feature vector and serves as the input for the next layer. This is also connected to the final classification model. The system as a whole is referred to as fully-connected layer. The process of conversion is referred to as Flattening.

5. After the completion of the “full connection”, softmax activation is applied. This way, each class is assigned decimal probabilities, reducing the burden of a multi-class problem.

6. Then classifying objects into discrete categories based on the criteria after examining the probability outcome.

References

[1] Mustafa, Wan Azani, Ragunathan Santiagoo, Irfan Jamaluddin, Nurul S. Othman, Wan Khairunizam, and M. N. K. H. Rohani. "Comparison of Detection Method on Malaria Cell Images." In 2018 International Conference on Computational Approach in Smart Systems Design and Applications (ICASSDA), pp. 1-6. IEEE, 2018.

[2] Mehanian, Courosh, Mayoore Jaiswal, Charles Delahunt, Clay Thompson, Matt Horning, Liming Hu, Travis Ostbye et al. "Computer-automated malaria diagnosis and quantitation using convolutional neural networks." In Proceedings of the IEEE International Conference on Computer Vision, pp. 116-125. 2017.

[3] Rajaraman, Sivaramakrishnan, Sameer K. Antani, Mahdieh Poostchi, Kamolrat Silamut, Md A. Hossain, Richard J. Maude, Stefan Jaeger, and George R. Thoma. "Pretrained convolutional neural networks as feature extractors toward improved malaria parasite detection in thin blood smear images."

[4] Shen, Hongda, W. David Pan, Yuhang Dong, and Mohammad Alim. "Lossless compression of curated erythrocyte images using deep autoencoders for malaria infection diagnosis." In 2016 Picture Coding Symposium (PCS), pp. 1-5. IEEE, 2016.

[5] Dong, Yuhang, 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),pp. 101-104. IEEE, 2017.

[6] LeCun, Yann. "LeNet-5, convolutional neural networks." URL: http://yann. lecun. com/exdb/lenet

[7] Deng, Jia, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. "Imagenet: A large-scale hierarchical image database." In 2009 IEEE conference on computer vision and pattern recognition, pp. 248-255. IEEE, 2009.

[8] Dataset from NIH website: https://lhncbc.nlm.nih.gov/publication/pub9932