Malaria Parasite Detection

Ronak Doshi
ronakvipul.doshi@iiitb.org
IMT2017523

R Arvind
r.arvind@iiitb.org
IMT2017519

Nithin Raj
nithin.raj@iiitb.org
IMT2017511

***Abstract*—This paper is a report on the project that was conducted to detect malaria parasite in an individual by analyzing segmented cells from images of thin blood smear slides.**
We propose a model that detects if a cell is infected by the malaria parasite or not - using concepts of Machine Learning and Image Processing.

I. Introduction

Malaria is a deadly, infectious, mosquito-borne disease caused by Plasmodium parasites that are transmitted by the bites of infected female Anopheles mosquitoes.
Internationally, malaria is responsible for approximately 1-3 million deaths per year. Of these deaths, the overwhelming majority are in children aged 5 years or younger, and 80-90% of the deaths each year.
If an infected mosquito bites you, parasites carried by the mosquito enter your blood and start destroying oxygen-carrying red blood cells (RBC). These deadly parasites can live in your body for over a year without causing symptoms, and a delay in treatment can lead to complications and even death. Therefore, early detection can save lives.
Malaria is the world's fourth leading cause of death in children younger than age 5 years. This is a motivation to make malaria detection and diagnosis fast, easy, and effective.

Here we propose two different models that helps in detecting malaria parasite from a blood smear slide image.
These three models are formed by:

1) Applying Image Processing techniques to detect blobs which represent the image being stained by parasite
2) Extracting features from the cell image and applying Machine learning models on them

II. Dataset

The infected and normal cell image collection used in this project is made publicly available by National Library of Medicine (NLM), National Institutes of Health (NIH) in their archives.
Giemsa-stained thin blood smear slides from 150 P. falciparum-infected and 50 healthy patients were collected and photographed at Chittagong Medical College Hospital, Bangladesh. The smartphone's built-in camera acquired images of slides for each microscopic field of view. The images were manually annotated by an expert slide reader at the Mahidol-Oxford Tropical Medicine Research Unit in Bangkok, Thailand

The dataset includes 27,558 cell images with equal instances of parasitized and healthy RBCs. Cells containing Plasmodium are labeled as positive samples while normal instances contain no Plasmodium but other objects including impurities and staining artifacts.

III. Data Visualization

Data visualization is an approach to analyze the dataset and summarize its main characteristics through some form of visualization.

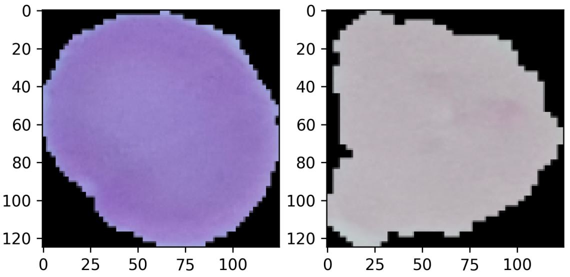


Fig. 1. Two different images of uninfected cells

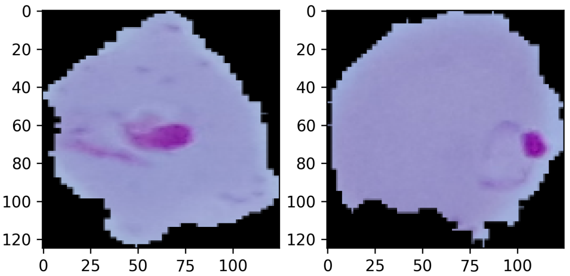


Fig. 2. Two different images of infected cells

As a simple visualization we can see that parasitic cell contains blobs in the inner region of the cell where as uninfected cell's inner region is clear. However there are some cells that does not contain any blobs but are infact parasitic.

IV. Data Pre-Processing

Data pre-processing is essential to filter out unnecessary data and convert the image into a proper form. The pre-processing of cell images are done in three steps:

- Scaling and brightness adjustment
- Gaussian filtering and Image segmentation

- Removing noisy regions

The first step we performed was to scale all the images to 100x100px size. Then we also increased the contrast and the brightness of the image in order to make the minor details discernible.

The next step is to apply filters to the image. Gaussian smoothing is an image processing technique used widely to reduce image noise and reduce detail. We can remove the sharp pixelated edges of the cell using Gaussian smoothing. But this also results in the image becoming blur. Therefore, the filter specifications have to be chosen carefully to make sure that we retain all the important information even after filtering and smoothing.

Now we apply a binary threshold to the image. Thresholding is an image segmentation technique. It takes in two parameters, a threshold and a color value. If the intensity of a pixel is greater than the threshold, it sets the color of that pixel to the input value, else it makes it white.

$$img(x,y) = \begin{cases} val & intensity(x,y) \geq thresh \\ 0 & intensity(x,y) \leq thresh \end{cases}$$

After thresholding we will be able to clearly distinguish between different parts of the cell. The cell mainly has two regions. The first one is the cell itself, which will be white in color and a pink region inside the cell which is the result of the stain on the blood cell. But thresholding also results in the appearance of small blobs all over the cell. These appear because these regions have their intensity above the threshold.

The last step is to remove these unwanted blobs. Using Morphological operations, we can easily remove small blobs from the cell. We take a square mask of size 1x1px and perform 50 iterations of erosion and dilation over the cell image. Then we take a 2x2px mask and perform a few iterations again. For an object present in an image with a certain background, the operation of erosion converts pixels associated with the object's boundary to pixels in the background. In Dilation, the bordering background pixels are changed to ones associated with the object. Objects become smaller during the erosion process, and enlarge or even merge during dilation. Filaments and isolated pixels in an image can be removed from the object in this way to smooth out the image.

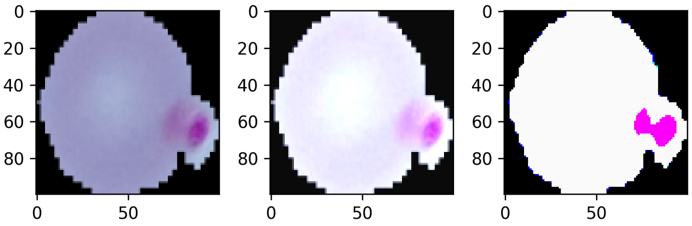


Fig. 3. Different Stages of pre-processing

The three phases of the processing stage has been shown in Fig 3. We can see the effect of changing the brightness in the second image. The third image is a result of thresholding and noise filtering. With this the pre-processing has been done the image can be used for further analysis.

V. Feature Engineering

A. Blob Detection

A simple feature to use for classification is the presence of a blob in the cell. Most of the cells affected with Malaria have a stain on the cell. This is not the case all the time, but this is applicable to most of the cells.

Therefore all we have to do is identify the pink spot in the cell. But to do that we have to take the image from RGB color space to the HSV space. Any color can be written as combination of RGB but that ratio wont help us at all in this case. A better representation for this is the HSV space. This color space describes colors (hue or tint) in terms of their shade (saturation or amount of gray) and their brightness value

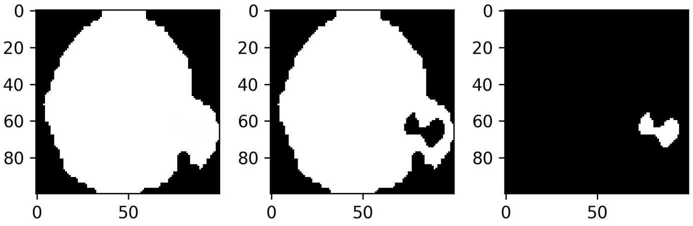


Fig. 4. HSV Color Space

The image in HSV space can be split into its constituent channels. The first two images of Fig 4 are the H(hue) and the S(saturation) channel of the image. On subtracting these two images, we can find the identify the location of the blob.

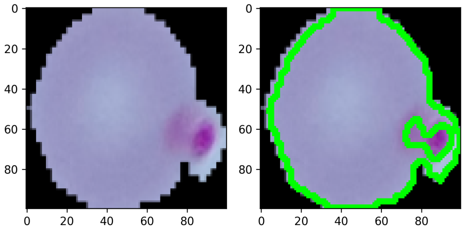


Fig. 5. Contour Detection for a Parasitic Cell

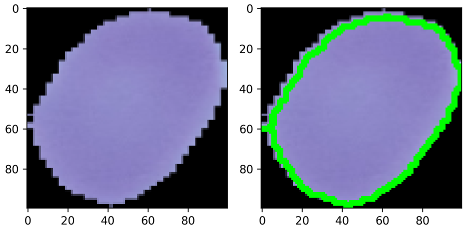


Fig. 6. Contour Detection for an Uninfected Cell

Once we can separate out the blob, we can apply contour detection algorithms to locate all the boundaries in the image.

Fig 5 and Fig 6 show the results of contour detection on cell with and without blob. Once we have the contours of blob, we can classify the image as parasitic or uninfected.

B. Key points detection

After detecting the number of contours, we needed more features that the contour detection would have missed so that we are able to classify the image more accurately. To get more features we used an opencv algorithm called SIFT (Scale-Invariant Feature Transform) and SURF (Speeded-Up Robust Features). SURF is nothing but a faster vesrion of SIFT.

SIFT algorithm is a corner detection algorithm. It extracts the key-points from an image and computes its descriptors. Keypoints are the “stand out” points in an image, so no matter the image is rotated, shrink, or expand, its keypoints will always be the same. The descriptor is the description of the keypoint.

There are mainly four steps involved in SIFT algorithm. We will see them one-by-one.

1) **Scale-space Extrema Detection:** To detect corners, scale-space filtering is used. It finds the Laplacian of Gaussian for the image with various σ values. σ acts as a scaling parameter. SIFT uses Difference of Gaussians technique. Difference of Gaussian is obtained as the difference of Gaussian blurring of an image with two different σ, let it be σ and $k\sigma$. Once this DoG is found, images are searched for local extrema over scale and space. If it is a local extrema, it is a potential keypoint.

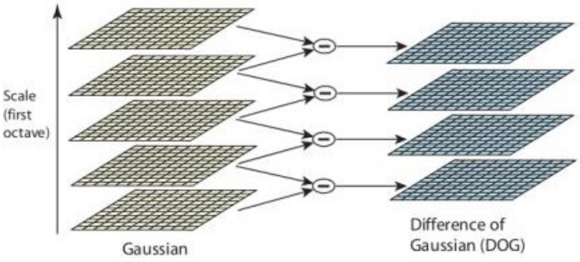


Fig. 7. Generation of Difference of Gaussians

2) **Keypoint Localization:** Once the location of potential keypoint is found, they have to be refined to get more accurate results. They used Taylor series expansion of scale space to get more accurate location of extrema, and if the intensity at this extrema is less than a threshold value (0.03 as per algorithm), it is rejected.
3) **Orientation Assignment:** A neighbourhood is taken around the keypoint location depending on the scale, and the gradient magnitude and direction is calculated in that region. An orientation histogram with 36 bins covering 360 degrees is created. The highest peak in the histogram is taken and any peak above 80% of it is also considered to calculate the orientation. It contribute to stability of matching.
4) **Keypoint Descriptor:** Now keypoint descriptor is created. A 16x16 neighbourhood around the keypoint is taken. It is divided into 16 sub-blocks of 4x4 size. For each sub-block, 8 bin orientation histogram is created. So a total of 128 bin values are available. It is represented as a vector to form keypoint descriptor.

C. Bag of Visual Words

Bag of visual words (BOVW) is commonly used in image classification. Its concept is adapted from information retrieval and NLP’s bag of words (BOW).

The reason behind using BOVW is that the SURF (Faster version of SIFT algorithm) detects many keypoints of an image and each keypoint has a corresponding descriptor vector of size 64. Now to convert all these descriptor vectors into one n sized features vector, we use bag of visual words approch.

The general idea of bag of visual words (BOVW) is to represent an image as a set of features. Features consists of keypoints and descriptors. We use the keypoints and descriptors to construct vocabularies and represent each image as a frequency histogram of features that are in the image. From the frequency histogram, later, we can find another similar images or predict the category of the image.

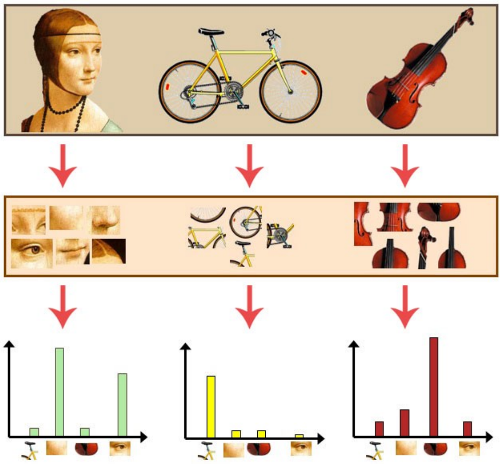


Fig. 8. Example of frequency histogram of image features

To build a bag of visual words, we first extract keypoints and descriptors using SURF algorithm. Next, we make clusters from the descriptors (using K-Means algorithm). The center of each cluster will be used as the visual dictionary’s vocabularies. Finally, for each image, we make frequency histogram from the vocabularies and the frequency of the vocabularies in the image. Those histograms are our bag of visual words (BOVW).

VI. Model Building

A. Blob Detection Model

Blob detection model is a subtle model where we just try to detect a blob in the inner region of the cell. The way we detect a blob is by using the contour detection method mentioned above.
As we saw in the data visualization, a parasitic cell contains a small blob in the inner regions of the cell where as an uninfected cell has a clear interior. Using this condition, if we are able to find any blob in the inner regions of the cell, model will consider the cell to be parasitic otherwise it will consider it to be uninfected. This model does not require any sort of training, given an image, it will detect if it is parasitic or not.

B. Machine learning Model

In this model we are extracting features from the image by using algorithms called SURF/SIFT mentioned above. After getting all the key-points and descriptors of the image by SURF we use Bag of visual words approach to generate a histogram with 20 categories using K-Means algorithm(with number of clusters as 20), which will then be a 20 sized feature vector of the image. Number of clusters for K-Means algorithm came out to be 20 by using trial and error approach ranging from (5 to 800), and got the highest accuracy when number of clusters were 20.
After generating the 20 sized feature vector of the image, we added number of blobs as one more feature using blob detection to this vector making it size of 21.
After generating this data-set, we apply different machine learning models such Random Forest, Support Vector Machine and Logistic regression to classify whether cell in parasitic or uninfected.

Observations

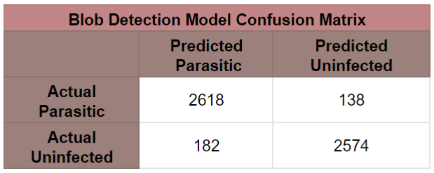
After generating the models, we trained each of the models with 20000 images of segmented cells from the thin blood smear(10000 cell images that were parasitic and 10000 cell images that were uninfected).
We tested the model with 5512 segmented cell from the thin blood smear images. Accuracy of each and every model is show in Table 1.

TABLE I
Model and their corresponding accuracy

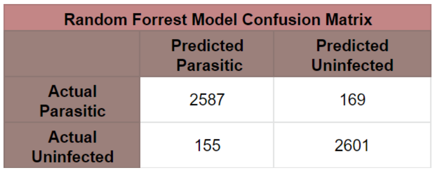
Model	Accuracy
Blob Detection	94.19%
Random Forrest	94.067%
Support Vector Machine	89.47%
Logistic Regression	85.86%

Above table demonstrate the accuracy of each model. Highest accuracy is seen using Random Forest Model and Blob Detection Model.
To get more insights on these two models, the confusion matrix for both models is shown below.

TABLE II
Confusion matrix of the generated model



Blob Detection Model Confusion Matrix		
	Predicted Parasitic	Predicted Uninfected
Actual Parasitic	2618	138
Actual Uninfected	182	2574



Random Forrest Model Confusion Matrix		
	Predicted Parasitic	Predicted Uninfected
Actual Parasitic	2587	169
Actual Uninfected	155	2601

From confusion matrix we can see that blob detection model predicted a cell being parasitic more accurately than random forest model. Whereas random forest cell was more accurate in predicting if cell is uninfected.
The most important factor is wrongly predicting parasitic cell, because if model wrongly predicts an uninfected cell, it's not a major concern as the patient can take precautions but if the model wrongly predicts a infected cell, that's where the problem lies. So, from the confusion matrix we can see that blob detection model predicted uninfected when the cell was actually parasitic less number of times than random forest model.

Conclusion

From the above statistics we can conclude that, highest accuracy was achived using blob detection model as well as random forest model. But best model for practical purpose is Blob Detection Model
Accuracy of the selected model is 94.19%

References

[1] OpenCV documentation : https://docs.opencv.org/
[2] SURF : https://docs.opencv.org/master/df/dd2/tutorial_py_surf_intro.html
[3] https://ieeexplore.ieee.org/document/7867644