Malaria Detection using Deep Learning

By Danush Sridhar

Problem Definition

- Malaria is a parasitic disease that is transported by mosquitoes and can be very deadly in many cases if not treated correctly or quickly
- Using microscopy cells can be observed and images can be captured to aid in the process of detection through the analysis of source images and our test images
- Through the process what we are trying to gain from this process is to be able to get a software that can gather data from source images or given data and be able to correctly identify and categorize cells with malaria present in them. Using this data then we can build a process that can be further improved in detection of Malaria.

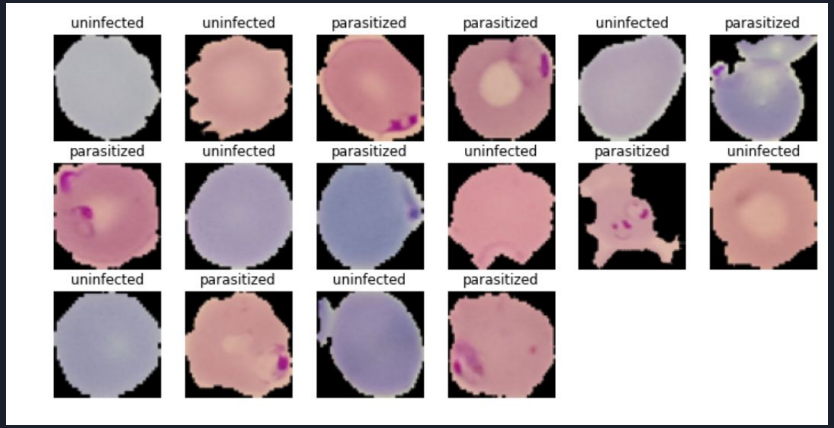
Source: wikipedia

Problem Approach

- The Way we are going ot approach this problem is by creating a data visualization and creating a software that can aid us detecting the Malaria present in the cells

- Through this process we look to gain a better understanding and approach to be able to use image detection to be able use our source images to compare and have a process that can accurately and proficiently detect Malaria in the cell images

- We can then use this approach to aid and grow the process and make it easier for scientist and researchers to be able to create a process from them to have better success rate in Malaria Detection.



Our process

- In our test run we are given a data set with images of cells that could be parasitized (malaria present) or they could be Uninfected

- Using the data we needed to categorize and be able to create a detection process that allowed us to be able create a model on our detection process.

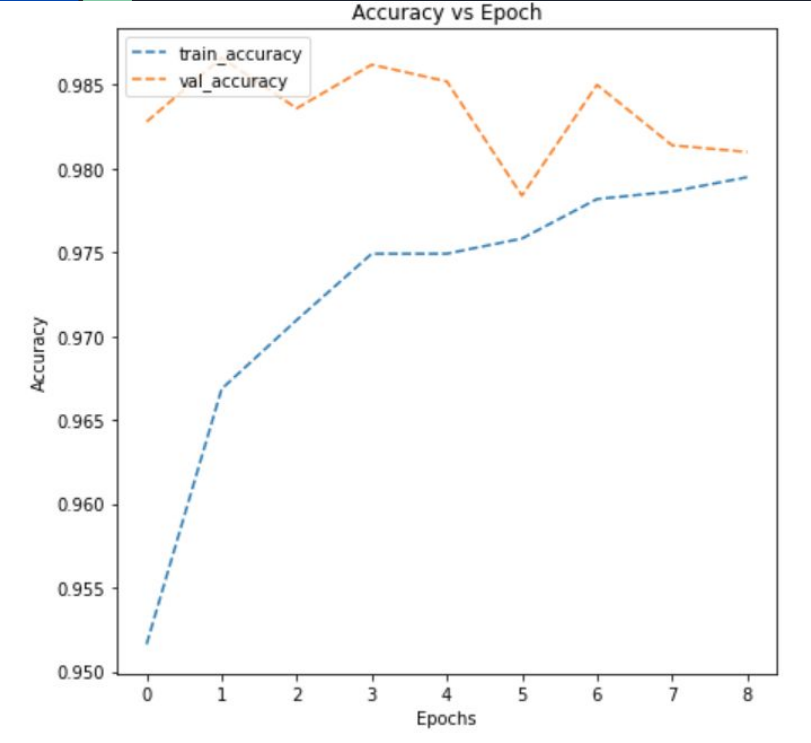
- We need to use this process to be able to create a model to help us understand how accurate our detection process is.

- With the deeper understanding of how the detection software works then we can create a process that can help us detect Malaria with more accuracy.

Executive Summary

- We can create Convoluted Neural Network model based on the images given.
- Through the model we look to analyze and understand how accurate the detection software works and how we can create a heat map that can entail our success rate of comparisons between each individual cells.
- We can use malaria detection software in many ways and the detection of malaria can be very useful in treatment plans and be able to create a solution to malaria.
- Malaria detection software can be useful in cell detection when seeing a spread of cell structures we can then use it to aid in bioinformatic process or even be able to use it in a large spread where we analyze and use the detection software to understand how the virus spreads.

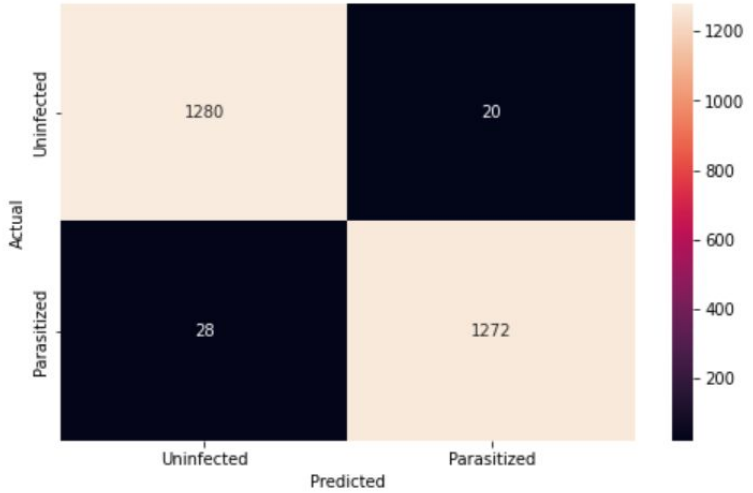
Models



- This model depicts the accuracy of detection and how the validation is accurate across the different epochs (the number passes the training data set took through the process)
- The validation accuracy and the training sets accuracy across the test was above 95% in this model.
- The data shows that the detection was pretty accurate across the board and the data collected really showed that the accuracy from the run can be useful for helping us use the code to be able to provide detection across other data sets.

Model Accuracy

	precision	recall	f1-score	support
0	0.98	0.98	0.98	1300
1	0.98	0.98	0.98	1300
accuracy			0.98	2600
macro avg	0.98	0.98	0.98	2600
weighted avg	0.98	0.98	0.98	2600



- So what we see here is a heat map that compares the the actual values of parasitized and uninfected cells comparing them to our predicted values of the cells.

Cons and benefits of the model

Cons:

- The cons of the detection model that is created is not one model can fit all data sets and there needs to be different models that can be created to help us understand the overall accuracy of the detection software
- The detection of the parasitized malaria cells can not be a 100% accurate all the time and we won't be able to get high accuracy with all the data sets that we might test

Benefits:

- This model although does not have a 100% accuracy most models don’t and these models can be really helpful in detecting malaria in early stages when being tested.
- We can always build upon the model to increase variation test data and be able to get more accurate reading based on the issues that we see.
- It can grow and be used in most similar cases where we are given microscopic images of cells with similar structure and are able to create a similar detection model to provide the data that is being required

Overall conclusion and understanding

- This detection can really aid in the process of malaria detection and can make scientists life much easier.
- The process can be simplified for scientist rather than them individually observing the cells to be processed if they can use the model detection software they can get a higher accuracy in detection at a faster rate providing them time solve the issue.
- It can then be used in the medical field as well were one is trying to observe a spread case where they can use this software to understand how many are parasitized in a set group and how many are unifected.

The End

Appendix

Source code and Html Link

```
# Creating sequential model
model = Sequential()

model.add(Conv2D(filters = 32, kernel_size = 2, padding = "same", activation = "relu", input_shape = (64, 64, 3)))

model.add(MaxPooling2D(pool_size = 2))

model.add(Dropout(0.2))

model.add(Conv2D(filters = 32, kernel_size = 2, padding = "same", activation = "relu"))

model.add(MaxPooling2D(pool_size = 2))

model.add(Dropout(0.2))

model.add(Conv2D(filters = 32, kernel_size = 2, padding = "same", activation = "relu"))

model.add(MaxPooling2D(pool_size = 2))

model.add(Dropout(0.2))

model.add(Flatten())

model.add(Dense(512, activation = "relu"))

model.add(Dropout(0.4))

model.add(Dense(2, activation = "softmax")) # 2 represents output layer neurons

model.summary()
```

```
Model: "sequential"
```

Heat map

```
from sklearn.metrics import classification_report

from sklearn.metrics import confusion_matrix

pred = model.predict(test_images)

pred = np.argmax(pred, axis = 1)

y_true = np.argmax(test_labels, axis = 1)

# Printing the classification report
print(classification_report(y_true, pred))

# Plotting the heatmap using confusion matrix
cm = confusion_matrix(y_true, pred)

plt.figure(figsize = (8, 5))

sns.heatmap(cm, annot = True,  fmt = '.0f', xticklabels = ['Uninfected', 'Parasitized'], yticklabels = ['Uninfected', 'Parasitized'])

plt.ylabel('Actual')

plt.xlabel('Predicted')

plt.show()
```