Capstone Project

Malaria Detection - By Nishad Khudabux

Executive Summary

The project proposes an automated system using computer vision as a replacement for both RDT and microscopy for the early detection of malaria in developing parts of the world. Using a convolutional neural network (CNN), this model can provide more accurate detection at a lower cost than the alternative. This is critical in regions where malaria remains most prominent due to economic and geographic limitations. Below we outline the model's design recognizing that it is only one part of the automated system needed to detect and treat cases on a large scale. It is recommended that we take the next steps to build out the additional systems required for that automated system. As well as a more robust analysis of the best methods to implement the completed system into the existing infrastructure.

Problem Summary

Malaria is a contagious disease caused by parasites transmitted through the bites of infected mosquitoes. The parasites enter the blood and begin damaging red blood cells. Late treatment can cause complications and even be fatal, but as the symptoms mimic the flu and stay dormant for a year or more, **early detection is key**.

Malaria is known as a **disease of poverty**. Despite the treatment being highly effective and relatively inexpensive, more than 241 million malaria cases and 672 thousand malaria-related deaths were reported worldwide in 2020, 95% of those found in Africa alone. This challenge primarily stems from several obstacles that make early detection difficult in poorer parts of the world:

- Collecting blood samples requires using expensive, large and heavy equipment (centrifuge and microscope), often in remote areas that do not have access to electricity.
- Gold standard diagnosis of those samples requires inspection by experienced professionals called **microscopy** but is often not feasible. Instead, cheaper but less accurate **rapid diagnostic tests (RDTs)** are more commonly used.

While the first problem has largely been solved through inspired innovation (paperfuge and foldscope), the latter continues to be an issue. The CNN model presented here provides a cheaper and more accurate alternative. Combined with frugal and mobile methods, an automated diagnostic system using machine learning can have a tremendous positive impact.

Solution Design

After concluding there is no simpler solution, it is apparent that detecting malaria is best done through training a neural network classification model. Both an ANN model and a CNN model have the potential to work. But given EDA showed infected cells are primarily characterized by the shape of parasites in different positions, a CNN model will do much better.

Critically, this project aims to build a model that surpasses the accuracy of RDTs **(~88%)** and matches that of microscopy **(~98% best case)**. If possible, we want to optimize for **recall** as the cost of a false negative could potentially lead to death, while there is little risk of false positives (malaria drugs have minor side effects, and patients would likely be tested a second time to confirm).

The base model showed a high level of test accuracy (98.35%), not surprising given EDA showed clear detection characteristics for parasitic cells in both colour and shape. But this means it will be difficult to optimize further. It should also be noted that with such a small margin to improve upon, it will be difficult to separate which changes are improving the model, as random variance in compiling can lead to changes that small.

Normally it would not be constructive to optimize for an additional 2%, but this case is an exception in both scale and cost of error. For example, in Africa, it was estimated to have 229 million malaria cases in 2020, a 2% error equates to roughly 4.5 million potential misdiagnoses. In those cases, a missed diagnosis has potentially lethal consequences.

A number of variations of CNN models were tested, including different loss functions (binary cross-entropy and categorical cross-entropy), different levels of complexity, different activation functions, data augmentations, and transfer learning (a high-level view of the progression of these models and their accuracy can be found in *appendix 1*).

From the multiple variations of CNN models, model_2 showed the highest efficacy. Using its architecture, multiple hyperparameters (including the number of layers, their respective filters, and batch size) were tuned to find the optimal combination (the final design structure is laid out in *appendix 3*). Key tunning included changing to LeakyRelu activation functions and dropping batch size to 16. From this, we compiled model_2o with a test accuracy of **98.69%**, which well surpasses that of RDTs and proves to be slightly better than microscopy. Critically this means it is capable of replacing both as a cheaper and more accurate diagnostic test.

Figure 1 shows the model training and validation accuracy over 8 epochs. We can clearly see the model does **not show signs of overfitting** and **generalizes well**. Also evident is validation accuracy peaking at epoch 7. It is therefore recommended we save our model with both the 7th and final epoch.

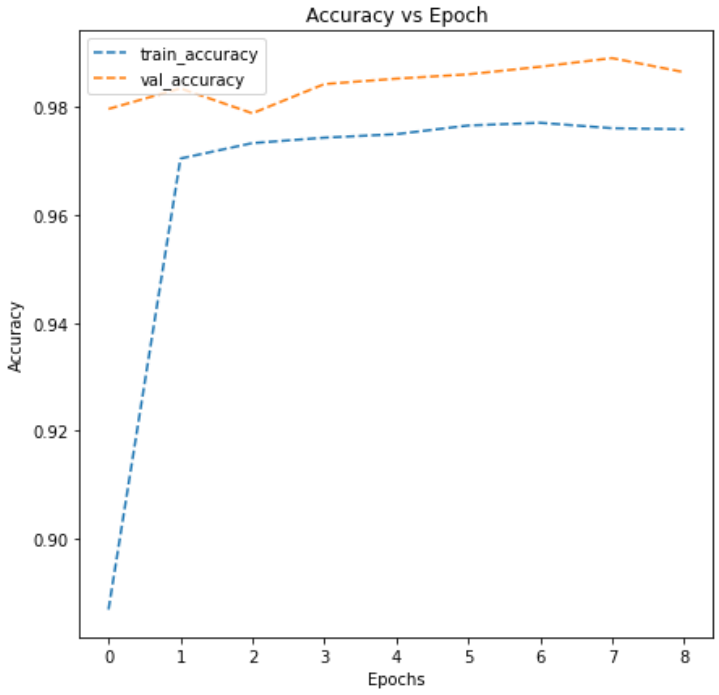


Figure 1 - Model_2o Accuracy Plot

Figures 2a and 2b show the model plotted on a confusion matrix. We can see the model sorted equally well for both categories (Parasitized and Uninfected) and therefore scored relatively the same for precision and recall. Given that both are such high accuracy, our **goal of a high recall was met**.

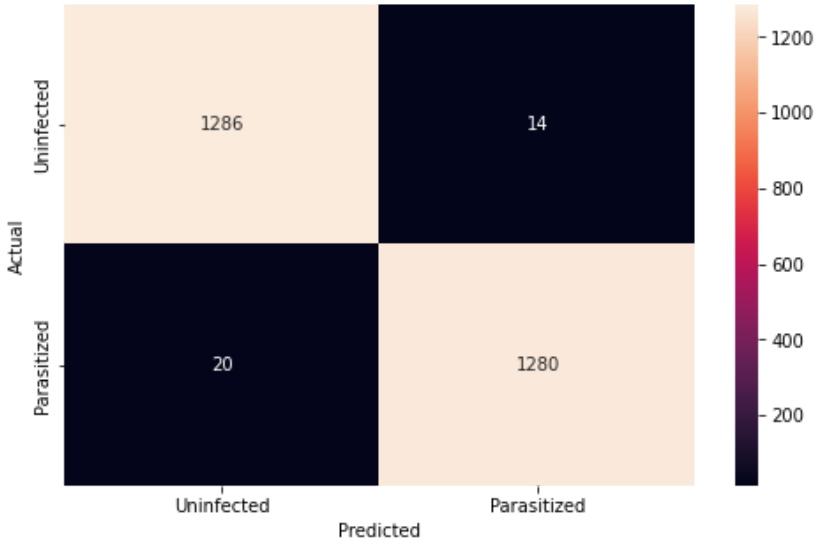


Figure 2b - Modle_2o Confusion matrix

	precision	recall	f1-score	support
0	0.98	0.99	0.99	1300
1	0.99	0.98	0.99	1300
accuracy			0.99	2600
macro avg	0.99	0.99	0.99	2600
weighted avg	0.99	0.99	0.99	2600

Figure 2a - Model_2o Precision and Recall Scores

The final proposed solution uses *model_2o* as a replacement for RDTs and Microscopy as it surpasses our objectives for accuracy and proves to be a strong generalized model. It also meets our objective of scoring high on recall, preventing potential false negatives. The model was saved as both a **.H5** and **.tflite** files (.H5 for general use and .tflite to use on cheaper hardware).

Recommendations for Implementation

With an average of 2000 cases in the United States yearly, malaria is not highly present in the western world. The in-frequency of cases is a known problem for microscopic detection, as labs in the developed world do not see many cases, and therefore technicians have less experience. There is, therefore, a potential use case for a computer vision model that could perform more reliably and cheaply.

However, with an estimated global 241 million cases and 672 thousand deaths in 2020, the **more relevant use case** for a computer vision model is in the **developing world,** where the disease is significantly more prevalent and deadly. As 95% of those cases are present in Africa, that will be the focus of our implementation.

Given that this case is of a philanthropic nature applied at a large scale, the **cost will be our primary metric** to focus on. In the words of Manu Prakash from Stanford's frugal labs, and the inventor of foldscope and paperfuge:

"It can’t cost a ton of money because every zero that you add to the cost of scientific equipment, probably.. millions and billions of people are cut off"

Minimizing cost while maintaining a high degree of accuracy and accessibility is critical in deploying and implementing this technology.

Estimated Cost Benefit Analysis

When dealing with diseases of poverty, in particular, viewing a solution through an economic lens is critical. A technological solution can't just perform better; it must also be cheaper to be implemented realistically. Two sources estimate the cost of implementing and maintaining a machine learning model:

Source 1:	Source 2:
~60K total over 5 years (USD)	~10-30K initial cost + 9k yearly maintenance (USD),

Which works out to be roughly the same.

A high cost for implementing our model would be using a cloud or edge-based system (described below)

- Cost of cloud system is providing satellite internet to technicians (~20-50/month).
- Cost of an edge system is purchasing hardware (5-100$)

If we go with a cloud system (at quick glance looks to be the more expensive option), an average of 35$/month for ~1500 technicians is 3.15M over 5 years.

To determine the cost using traditional methods of detection, we can look at the cost of a single malaria diagnosis in the United States (this cost varies by country, but the cost in the US is a good baseline, and it will likely be on the cheaper side meaning we are erring in favour of the traditional methods)

- Cost of RDT 2.19$ USD (labour + materials)
- Cost of Microscopy 6.98$ USD (labour + materials)

Given our focus is on the developing world where RDTs are used significantly more than Microscopy, let's assume a detection cost of 2.19$/person (we are not taking into account cases that require multiple tests, again erring in favour of the traditional methods). In 2020 there were an estimated 241 million cases of malaria globally. Looking at Africa (95% of the cases = 229M cases), we can multiply that by the cost of an RDT test (229M*2.19$) gives ~500 million dollars.

Estimated Cost of Machine Learning Model	**Estimated Cost of RDTs in Africa**
~3.23M over 5 years	**~500 Million/ year**

This significant gap is due largely to the fact that a machine learning model is **nearly infinitely scalable**. Even though the relative cost of traditional testing is low, the sheer number of cases skews heavily in favour of a scalable system.

Admittedly this is a rough cost-benefit analysis, but given the large separation in cost, even a few hidden factors would not likely bridge that gap. For this reason, it is likely beneficial to implement a machine learning solution.

Cloud vs Edge System

In order to physically deploy our model, we have to choose between a cloud or edge-based system.

1. **Cloud System** - Would load the model online where it could be accessed through the internet. This would have the benefit of being able to load and receive information remotely, without the

need to carry and maintain (pottentially fragile) equipment. It would also be easy to update the model in one location continually. The cost of running and maintaining this server would be minimal and easily factored into our yearly maintenance costs. However, the limiting cost would be providing satellite internet to those technicians in remote regions (cost ~20-50$/month).

2. **Edge System** - Would load the model on a physical computer or chip, which would have the benefit of not requiring internet connection in rural areas. However there is a upfront cost of providing the hardware (5-100$/unit), as well as potentially replacing it. It would also be difficult to update the model as it would require updating each individual piece of hardware.

While new low cost edge chips have now allowed them to be competitive with a cloud based system, it is not entirely clear which would ultimately cost less. It would be necessary to gather more information:

- Do the technicians already have access to internet as part of their work?
- Do the technicians already have access to capable computing as part of their work?
- Is an option cheaper in non rural areas and what percent of that is within our scope of work?
- How sustainable are both systems?

By answering these and other questions a more thorough cost-benefit analysis can be done to determine which system is ultimately best.

Implementation

It is important to recognize that our model is only one part of the potential **automated system** that is required to diagnose and ultimately treat malaria. Once a blood sample is drawn for testing, it must be first put into a centrifuge before being looked at under a microscope. Innovations like paperfuge and foldscope allow this to be done inexpensively while remaining portable. Commonly, some form of smartphone (as they are often carried by the technicians) is held up to the foldscop to create a digital image of the sample (**figure 3**).

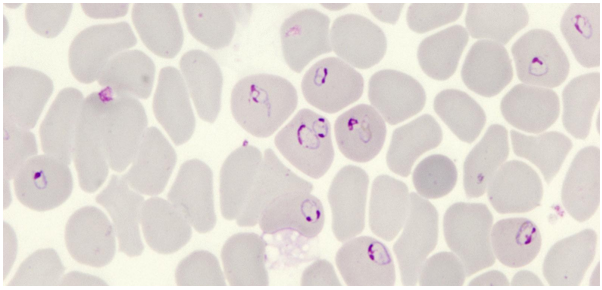


Figure 3 - Blood Sample Containing Malaria

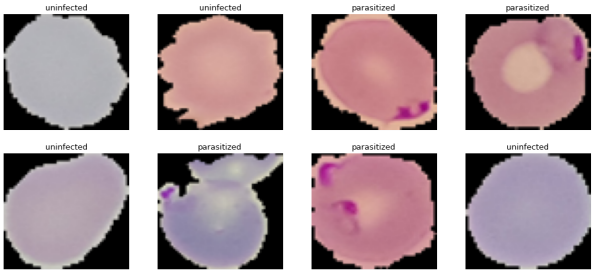


Figure 4 - Samples From our Dataset

Our model however has been trained on images of isolated blood cells (**figure 4)** meaning we would need to build a program that isolates a samples blood cells that we could then apply our model to (potentially a segmentation computer vision model could be used). From there a second program could aggregate the information to determine if the number of infected cells detected indicates a positive diagnosis (since a blood sample is likely to contain multiple infected cells, our model 1% error would likely be eliminated by this step). Once a positive diagnosis is received there are of course all the additional processes involved in treatment.

This is of course nested inside the greater infrastructure of organizations, facilities, and technicians required to detect and treat malaria. It is recommended that this automated system be added to that already existing infrastructure as a tool to improve the work already being done.

Risks and Challenges

While our model can detect if a patient is infected with malaria it is unable to diagnos the type or severity of symptoms. Therefore there is a risk of not recognizing a severe case (eg: advanced plasmodium falciparum) and not responding with the correct urgency. However, as we are targeting early detecting and sever symptoms are easily recognizable this risk is minimal.

As with all systems, they should be periodically checked to ensure they are functioning properly and to the same efficacy. While this is a solution that can be applied on mass cheaply, it is still vital that medical proffessionals remain vigilant and practicing in the field. There is always the potential for a slightly different strain to develop that our model is not trained on. Or even a small change in the collecting of data. Our model is trained on a specific data set and any deviation from that will not perform well unless it is reintergrated.

A challenge that all new technologies face is the learning curve in implementing them. This is especially difficult in developing regions that may not have as much experience with digital technologies. This lack of experience can also lead to fear of the unknown that may cause them to reject a new tool for one that is already known. The end goal should be that these regions can become self sufficient in using these tools, and that requires a large amonth of education and patience.

Recomendations

Our model clearly shows a high decree of efficacy which suggests it is ready to be implemented in the field. What is missing is the surrounding automated systems that our model would work within to provide diagnosis.

Our rough estimates are robust enough to show viability, but implementation requires a more detailed analysis. Specifically reaching out established organizations already working in the region for specific numbers and to understand how our tool can best be utilized.

Despite the scalability of this technology, it would be better to start with a small test case (perhaps including RDTs to begin with to ensure safety) to test the systems and work out potential bugs. Once established it can be quickly scaled up to have a greater positive impact.

Bibliography

- Foldscope https://foldscope.com/
- Paperfuge https://news.stanford.edu/2017/01/10/whirligig-toy-bioengineers-develop-20-cent-hand-powered-blood-centrifuge/
- Cost of deploying machine learning model https://www.phdata.io/blog/what-is-the-cost-to-deploy-and-maintain-a-machine-learning-model/
- Cost of deploying machine learning model source 2 https://medium.com/cognifeed/the-cost-of-machine-learning-projects-7ca3aea03a5c#:~:text=On
- Number of peace core volunteers sent to Africa 2019 https://journals.sagepub.com/doi/10.1177/2164956120976107#:~:text=prevention%20and%20control.-,In%20Fiscal%20Year%202019%2C%201408%20Volunteers%20contributed%20to%20malaria%20prevention,15%20are%20PMI%20focus%20countries.
- Cost of RDT (india) https://srlworld.com/blogs/malaria-symptoms-causes-types-treatment-test-cost

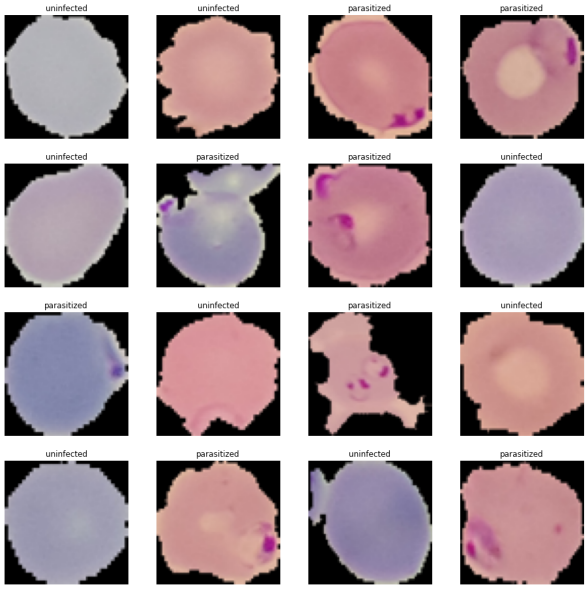
- Cost of diagnosis (USA) https://idpjournal.biomedcentral.com/articles/10.1186/s40249-020-00745-9#:~:text=In
- Number of cases in Africa https://www.who.int/news-room/fact-sheets/detail/malaria
- Cost of internet in Africa https://africa.konnect.com/en/what-is-satellite-internet/price-of-satellite-internet
- Edge detection chips options and cost
    - https://canaan.io/product/kendryte-k510
    - https://coral.ai/products/
    - https://www.raspberrypi.com/products/raspberry-pi-4-model-b/
- Accuracy of the two main types of malaria detection (microscopy and RDT) https://www.cdc.gov/malaria/diagnosis_treatment/diagnosis.html
- Accuracy of the two main types of malaria detection (microscopy and RDT) source 2 https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6668302/
- Microscopy in the US and number of cases https://www.cdc.gov/malaria/diagnosis_treatment/diagnostic_tools.html
- Accuracy RDT source 3 https://malariajournal.biomedcentral.com/articles/10.1186/1475-2875-10-176

Appendix

Appendix 1: Table of different models tested and their accuracy

Model	Description	Test Accuracy
Model_0	Base model (given)	98.35%
Model_1	Change from binary_crossentropy to categorical cross_entropy	97.85%↑
Model_2	Added 4rth layer (64 filters)	98.07%↑
Model_3	Change to LeakyRelu and added Batch Normalization Layers	96.23%↓
Model_4	Data Augmentation	97.73%↓
Model_5	Transfer Learning (VGG16)	92.69%↓
Model_2o	Optimizing our best model by testing multiple micro-adjustments.	98.69%↑

Appendix 2: Images of isolated red blood cells in contain in our dataset. Note the clear characteristics of colour and shape in parasitic infected cells.



Appendix 3: Architecture of Model_2o

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 64, 64, 32)        416

 leaky_re_lu (LeakyReLU)     (None, 64, 64, 32)        0

 max_pooling2d (MaxPooling2D  (None, 32, 32, 32)       0
 )

 dropout (Dropout)           (None, 32, 32, 32)        0

 conv2d_1 (Conv2D)           (None, 32, 32, 32)        4128

 leaky_re_lu_1 (LeakyReLU)   (None, 32, 32, 32)        0

 max_pooling2d_1 (MaxPooling  (None, 16, 16, 32)       0
 2D)

 dropout_1 (Dropout)         (None, 16, 16, 32)        0

 conv2d_2 (Conv2D)           (None, 16, 16, 32)        4128

 leaky_re_lu_2 (LeakyReLU)   (None, 16, 16, 32)        0

 max_pooling2d_2 (MaxPooling  (None, 8, 8, 32)         0
 2D)

 dropout_2 (Dropout)         (None, 8, 8, 32)          0

 conv2d_3 (Conv2D)           (None, 8, 8, 64)          8256

 leaky_re_lu_3 (LeakyReLU)   (None, 8, 8, 64)          0

 max_pooling2d_3 (MaxPooling  (None, 4, 4, 64)         0
 2D)

 dropout_3 (Dropout)         (None, 4, 4, 64)          0

 flatten (Flatten)           (None, 1024)              0

 dense (Dense)               (None, 512)               524800

 leaky_re_lu_4 (LeakyReLU)   (None, 512)               0

 dropout_4 (Dropout)         (None, 512)               0

 dense_1 (Dense)             (None, 2)                 1026

=================================================================
Total params: 542,754
Trainable params: 542,754
Non-trainable params: 0
_________________________________________________________________
```

Compiling the model

```
[ ]  model_2o.compile(loss = 'categorical_crossentropy', optimizer = 'adam', metrics = ['accuracy'])
```

Using Callbacks

```
[ ]  callbacks = [EarlyStopping(monitor = 'val_loss', patience = 2),
                  ModelCheckpoint('.mdl_wts.hdf5', monitor = 'val_loss', save_best_only = True)]
```

Fit and Train the model

```
[ ]  history = model_2o.fit(train_images, train_labels, batch_size = 16, callbacks = callbacks, validation_split = 0.2, epochs = 20, verbose = 1)
```

Appendix 6: Model_2o Accuracy and Loss

```
82/82 [==============================] - 4s 43ms/step - loss: 0.0468 - accuracy: 0.9869

 Test_Accuracy:- 0.986923098564148
```

Appendix 5: Addressing Latency

Latency in this case study is **not a relevant factor**, compared to for example computer vision in self-driving cars where increased computation time can result in an accident. While we can run a machine-learning model on a database of malaria samples for many hours with no consequences. For that reason, the model compilation time is observed in the notebook, but it will be excluded here.