Jessica Balke
Final Submission

Malaria Detection

I. Executive Summary

Key problem being solved

In this problem, we were tasked with predicting whether an image of a cell is parasitized or not. Being able to classify images as “parasitized” or “uninfected” would allow health care providers to much more readily screen and test people for malaria. The current solution is manual screening in a laboratory by a skilled technician, which is time consuming, costly, and most importantly, inaccurate. Improving malaria detection using deep learning methods could revolutionize how the disease is handled, how soon patients are treated, and increase chances of survival.

Final proposed model specifications

Convolutional Neural Networks (CNN) is a natural candidate for this type of problem, as it is designed for image data. We evaluated a variety of CNN techniques including batch normalization, data augmentation, as well as transfer learning. The final model that performed the best was a CNN model with batch normalization and data augmentation, with an accuracy rate of 95.6%, a precision of 96%, and recall of 94%.

Key benefits and takeaways

The key benefits from this final model will be rapid and accurate detection of a malaria infection present in red blood cells. A time-consuming, costly, and inaccurate manual process in a laboratory will be replaced by a trained neural net.

Roadmap for implementation and next steps

The roadmap for implementation will consist of some next steps, which include building software that can be used in hospitals / medical offices. The lab tech uploads the images of the malaria cells, and the program returns a list of which cells are infected. It is up to that medical professional to use that information to make a diagnosis, have the patient undergo further testing, etc. Efficiency, cost, and computation time are important considerations as this process must be done in developing countries where there might not be infrastructure and technology to support a highly complicated and expensive solution.

II. Problem and Solution Summary

What was the problem being solved?

Nearly half the world's population lives in locales at risk of malaria transmission, according to the CDC. Malaria mainly affects young children and pregnant women in impoverished, tropical areas. Undoubtedly this is an important problem to solve affecting 229 million people with reported cases and causing 400K deaths. Africa is the most affected due to a mosquito responsible for high transmission rates, the presence of Plasmodium falciparum, scarce healthy resources and an instable economy.

If Malaria is detected early, it is treatable, according to the Cleveland Clinic. Traditional diagnosis requires a bloodsmear and careful analysis of the red blood cells in the laboratory by a trained technician. Manual evaluation of the red blood cells is inconsistent at best, not to mention a time-consuming and expensive process, that is not scalable.

Early and accurate detection is key to preventing deaths. Deep learning techniques can be proven in this paper to be more accurate, faster, and likely less expensive than the current method.

Key questions being answered

- What is the baseline for accuracy, time and cost, for the current approach?
    - This question is still unanswered but would be helpful in communicating the value of our approach.
- Can deep learning methods result in acceptable accuracy and precision rate, without overfitting?
    - The answer is yes, with discussion to follow in paper.
- Is the data evenly distributed between parasitized and uninfected?
    - The data is evenly distributed with 14049 parasitized cells and the same number of uninfected cells.
- What are our KPI's?
    - Accuracy and Precision of infected class

Key insights

The good news is that the problem of detecting cells infected by the Plasmodium parasite that causes malaria is an extremely solvable problem. Right from the beginning, our baseline model that contained only a few layers and lacked complexity had an accuracy rate of nearly 95%. This is extremely good for deep learning in general, as a much worse model would still be usable in the industry.

It was difficult to get above a 95.6% accuracy rate without overfitting. Some recommendations from Milestone 2 included fining tuning and hyper parameterization to see if any additional improvement could be reached. We've tried a multitude of approaches to fine tune and improve that model (outlined in appendix) but the accuracy rate did not budge.

Approaches used to solve the problem

The overall methods performed were Convolutional Neural Networks, with varying levels of complexity and techniques as well as a Pre-Trained Model. We also tried using Data Augmentation, meaning computer generated images, and Gaussian blurring.

Convolutional Neural Networks is a type of Neural Network designed for image data. They use hidden layers which perform convolutional operations. Generally speaking, it consists of a convolutional layer, a pooling layer, and a fully connected layer. Each convolutional layer scans the image for a specific feature, whether it's horizontal edges, vertical edges, etc. The essence of CNN is to reduce the image into a matrix of 1's and 0's that is easy to process, while retaining features good for prediction.

Another variation of this approach was to first use an image generator to augment the image prior to running the CNN model.

The last approach was using a Pre-Trained Model. The idea of this approach is that if your data set is similar to other data sets on the internet, time and money can be saved by using a pre-trained model that is already tuned to a specific data set. It was not surprising that this model did not perform as well as the CNN models. Images of malaria infected cells may not be common problem reflected in internet libraries of images, unlike say if the images were of cars and traffic or of numbers.

Below is a summary of the models and their relative performance:

- Baseline Model: Convolutional Neural Networks – 3 layers
- Model 1: CNN Model with 5 layers
- Model 2: CNN Model with 5 layers & Batch Normalization & Leaky Relu
- Model 3: Same as Model 2 but with Data Augmentation
- Model 4: Pre-Trained Model (VGG16)

Model	Model Name	Accuracy	Precision	Recall
Baseline Model	CNN Model – Simple (3 layers)	94.58%	96%	94%
Model 1	CNN Model – 5 layers	94.69%	93%	95%

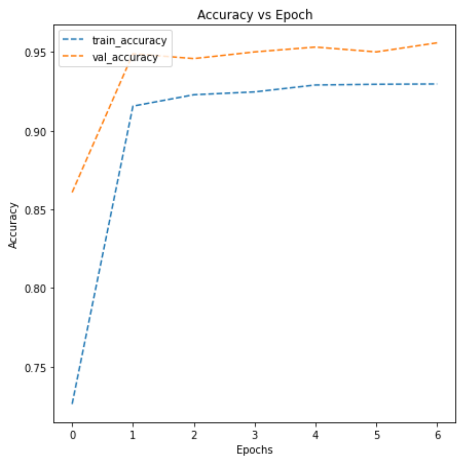
Model 1.1	Same as Model 1 with no Dropout	93.85%	93%	95%
Model 2	CNN Model – 5 layers with Batch Normalization & Leaky Relu	95.19%	96%	94%
Model 3	CNN with Data Augmentation	95.58%	97%	93%
Model 4	Pre-Trained Model	93.77%		

Please note that variations of the above models were tried, with hyper parameter tuning, for a total of 18 models. Ten variations of model 3 were tried as this was the best performing model and the most time was spent fine tuning it.

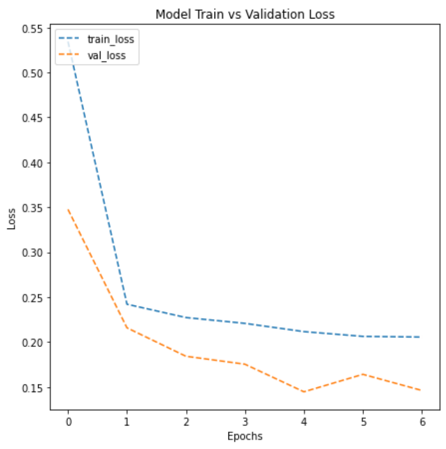
Which one is performing relatively better?

Using generated images with a CNN model (Model 3), with Batch Normalization and Leaky Relu, yielded the best results of all the approaches we attempted, in terms of accuracy and precision. The model was not overfitted, although the validation accuracy did not increase much after the first epoch (in Model 3 Accuracy graph below). The validation loss in model 3 had a nice downward trend, which is what we want to see. The goal of neural nets is to minimize loss as fast as possible. The smaller the loss, the better job our classifier is doing at modeling the relationship between our input and output data.

Model 3 Accuracy

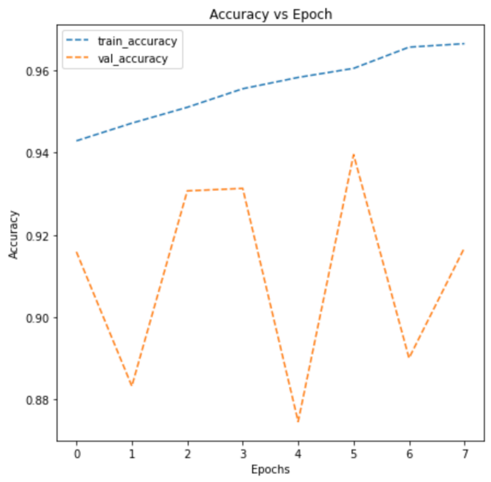


Model 3 Loss



Model 4, the transfer learning model was very overfitted, which can be seen in the below graph, since the test data accuracy was so far below the training data accuracy. In addition to being overfitted, it took more computation time, which is common with transfer learning models.

Model 4 Accuracy



The other models were not overfitted until more complexity was added to them while testing various parameters. The accuracy of the other models were not much lower than model 3.

Final Solution Design

A CNN Model using Generated images, using Batch Optimization and Leaky Relu should be adopted. Since the last milestone, we have experimented with optimizing additional parameters such as number of layers, neurons, dropout, activation functions, and optimizers. Unfortunately, not a single one of these efforts boosted performance. Increasing the number of neurons while reducing Max Pooling layers improved the accuracy rate ever so slightly, but it caused slight overfitting.

Below is a discussion on the different parameters tried within the context of Model 3.

Parameter	**What we Tried**	**What worked best**	**Why**
Number of Hidden Layers	3 layers 5 layers 7 layers	5 layers	7 layers resulted in slightly lower accuracy.
Number of Neurons	512 256 128 64 32	32 & 64	512 did not work. 256 and 128 had slightly lower accuracy and precision
Activation Functions	Relu (model 1 & 3) Leaky Relu (model 2) Softmax	Leaky Relu (and Softmax in output layer)	Leaky Relu layers along with relu activation function

			worked well in model3
Optimizer	Adam (model 2-4) Adamax (model 1,3)	Adam	Adam works better than Adamax in model3
Batch Size	128 – Model 1 64 – Model 3 & 4 32 – baseline & model	64	64 worked better than 128 in model 3
Learning Rate	0.1 – model 1.1, 1.2, 1.3, 2.1 0.01 – model 1, 3.6 0.001 – model 2 & 3	0.01	0.001 worked better than 0.01 in Model 3
Dropout	Baseline, Model 2 & 3 contained dropout. Model 1 had none.	Model performs well with dropout at rate of .2 at every layer, plus final dropout layer at .4	Experimented with less dropout layers in model 3 but it reduced model performance.
Batch Normalization	Model 2 & 3	Batch Normalization	Model performed better with a Batch Normalization layer.
Dense Layer	All models had dense layer of 256 neurons.	Dense layer with 256 neurons	
Max Pooling	All models had Max Pooling at every layer but we experimented with removing some.	Max Pooling layer of pool size 2 at every layer	Works best at every layer.
Kernel Size	3x3, 5x5, 7x7	3x3	Model 3 worked best with 3x3 on all layers rather than 3x3 on first 3 and 5x5 on last 2.

Why is this the best solution to adopt?

We determined that the important metrics were accuracy and Precision of the infected class. The CNN model 3 with Image Augmentation using Leaky Relu and Batch Normalization has the best accuracy of 95.58% and precision of 96%. It is not overfitting and has reasonable computation time compared to others.

As far as why precision is an important KPI, we are concerned with detecting as many parasitized cells as possible, and don't want to miss any and say they are uninfected when they actually are. It would be better to accidentally label some as parasitized when they are not. The person can go in for further testing, and later conclude they don't have malaria.

Deploying this final model to production would result in faster detection of malaria and more prompt treatment, and ultimately more control of the disease.

III. Recommendations for Implementation

What should the implementation roadmap look like?

Once the final model is trained and deployed, we'll need to develop an interface for health care staff where they can upload images of the patients' cells, and receive an output of which cells were infected. It will be up to the doctor or health care provider to interpret the results and diagnose the patient with malaria or not. There should be an image processing layer to make sure the images are the same size and taken the same way. It should be easy to use and intuitive.

The computational time and expense should be kept in consideration to make it affordable as possible.

What are the expected benefits or costs?

The benefits are cost and time savings that it would take to manually analyze images in a lab. By a more accurate and faster solution, malaria detection can be deployed at scale. It could make the disease much more manageable, treatable, and less life-threatening if early detection became the norm. Managing such a deadly disease that wreaks havoc not only on peoples' lives but the economy would increase stability both mentally of the people in affected areas and also economic stability. The benefits of saving people's lives are endless and can allow the countries and people affected to focus on other problems and challenges than survival.

The costs include data warehousing and storage, computation time, software expense, hardware expense, training, and one-time start up costs. Training technicians and to use this software, and ongoing maintenance costs should also be factored in. Luckily, this will be offset by the savings of the expensive process of manually analyzing cells in a lab which would be rendered a less effective method.

What are the key risks and challenges?

A key risk is finding a solution that is practical in a developing country with limited infrastructure, wifi access, hardware tools, hospital systems, etc. Some unknowns are the hardware capabilities of health care providers in areas affected by malaria and if they can not only afford such hardware and software, but if it will be compatible with the systems they currently have in place.

A model that is too complex and computationally expensive could be risky if it cannot be supported in the hospitals/laboratories at hand, or if there is no way to pay for it. A sponsor of some sort that can cover one-time install and ongoing maintenance may be extremely helpful.

Understanding the realities of what the health care systems in developing countries can handle as far as an advanced neural networks model is key. The goal would be to learn what these limitations are, and then ensure that the model is compact and simple enough to be scalable and sustainable. Luckily, if the model has to be simplified, the accuracy rate is still very high with some complexity removed.

What other problems need to be explored and in what priority / order?

There are some additional questions to answer in regards to implementation. Are there any limitations with how up to date the cell imaging equipment, hardware and software equipment of the hospitals that will be utilizing this software? Does any hardware or special equipment need to be installed for the whole system to work? Is there a budget limitation, given that these are impoverished countries, or is a donor or foundation sponsoring this effort?

Can the computers that exist at the hospitals in these countries handle a neural networks model, and what server or cloud computing system will power the model and hold the data of the images?

Is there scope to improve the performance further?

Using stride rather than Max Pooling is a technique that could be explored.

When we added additional layers to the model, and/or increased the number of neurons, the performance dipped. Because these are not high definition images, but images of cells, The model worked better with less complexity and lots of dropout. Increasing the layers from 3 to 5 only had a marginal impact on performance. Anything above 5 layers or 64 neurons, the model started to perform worse.

This leads me to believe that training the model with a larger image size (if there were no hardware limitations) might yield a negligible improvement, if any. It’s almost as if less information is better for the model – and it learns better when images are more simple and less noisy. It’s not like facial recognition where higher resolution would improve the model.

Very little, if anything, could be done to improve the performance of the model. A conversation about reducing model complexity, to increase speed and reduce computational expense, would make more sense, and might only shave of a half percentage point of accuracy.

Ensuring that the training data is coming from a diverse variety of patients – ages, races, genders, would be key to ensuring the model is inclusive.