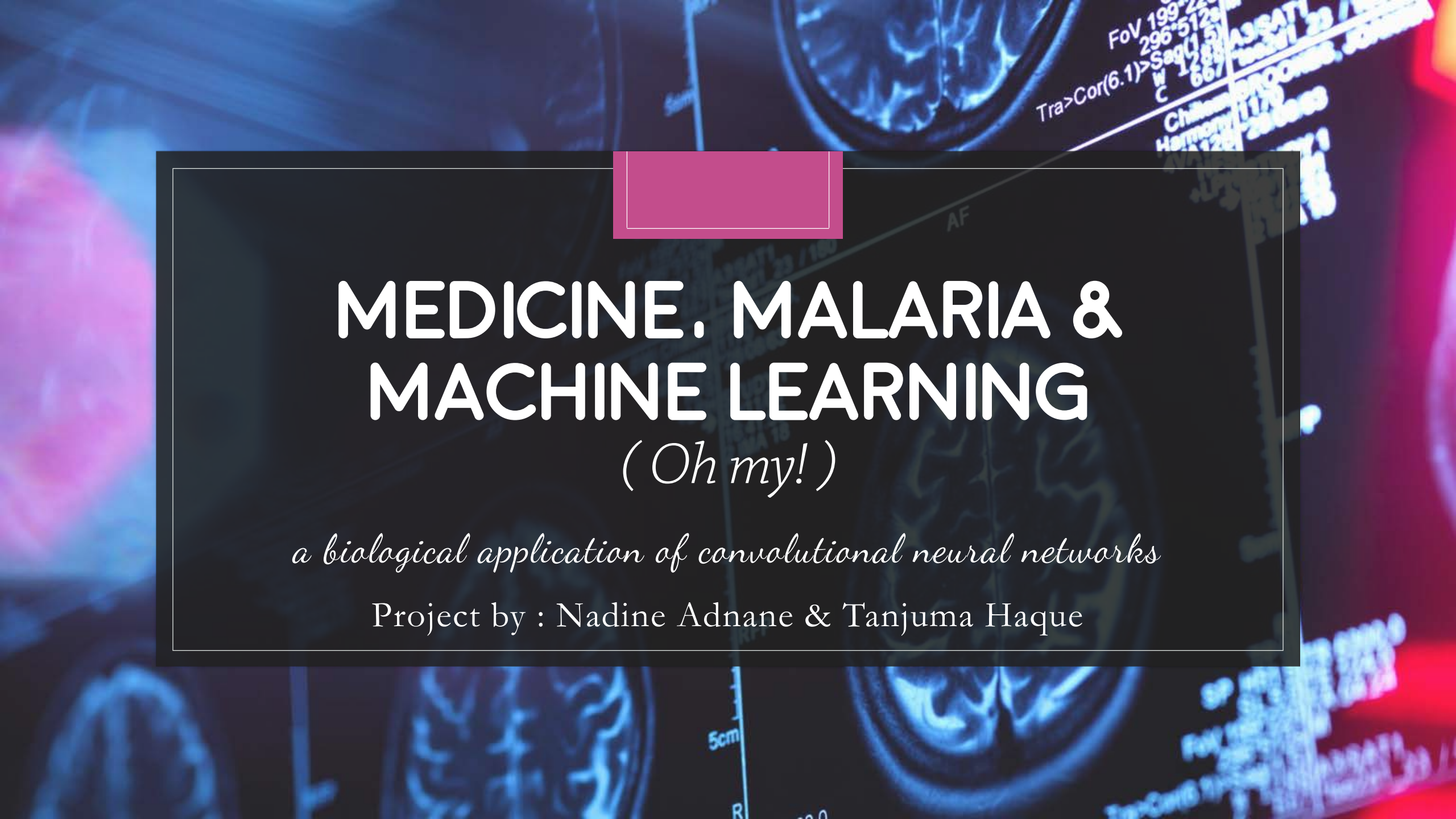


MEDICINE. MALARIA & MACHINE LEARNING

(Oh my!)

a biological application of convolutional neural networks

Project by : Nadine Adnane & Tanjuma Haque

Inspiration

- Nadine is Pre-med!~
- Tanjuma is interested in computational biology!
- Sara's work is inspiring!
- Diseases are interesting (from an academic perspective!)

National Library of Medicine
Malaria Dataset
https://lhncbc.nlm.nih.gov/publication/pub9932

Cases

228 million

malaria cases worldwide in 2018

Deaths

405 000

malaria deaths worldwide in 2018

Malaria Quick Facts

"Every 2 minutes, a child dies of malaria. And each year, more than 200 million new cases of the disease are reported. Although countries have dramatically reduced the total number of malaria cases and deaths since 2000, progress in recent years has stalled. Worryingly, in some countries, malaria is on the rise." *

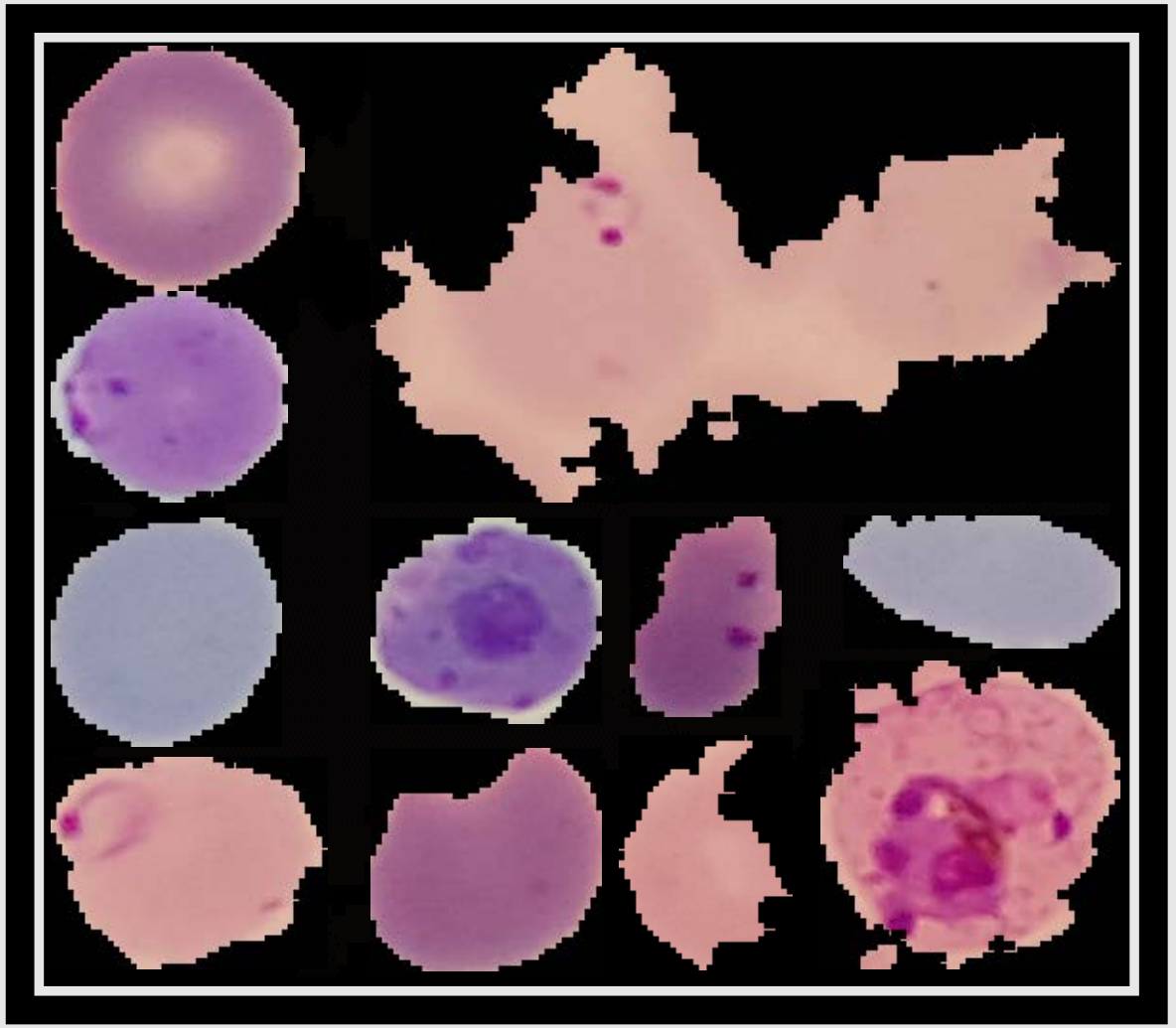
*https://www.who.int/news-room/facts-in-pictures/detail/malaria

The Dataset

- NIH Malaria Cell Dataset
- 50 healthy patients, 150 infected patients
- Chittagong Medical College Hospital, Bangladesh
- 13779 “Infected” cell slide images

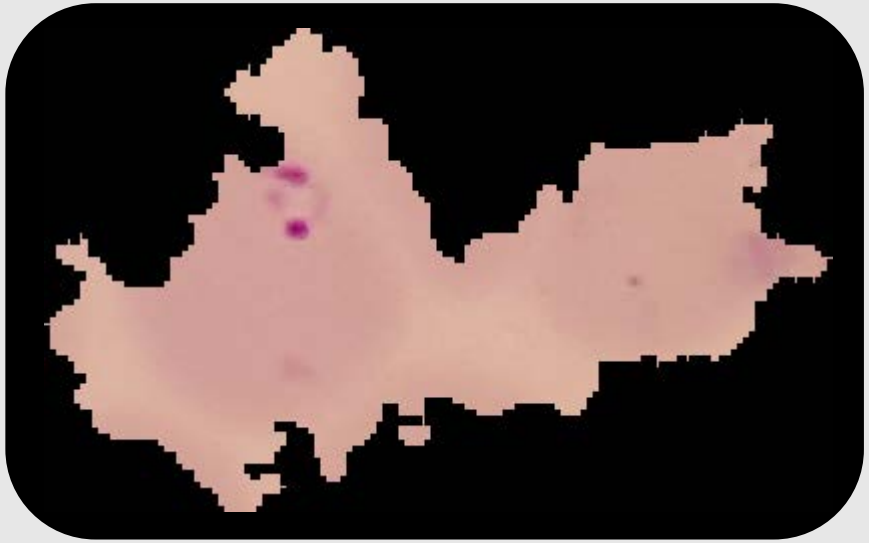
- 13779 “Healthy” cell slide images

- Total: 27558 images



Pre-processing

- Images are all different sizes!
  - Smallest image: 49 x 58
  - Largest image: 394 x 241
- Padding
  - Advantage – no data lost
  - Disadvantage – more pixels to deal with
  - Runtime = BAD
- Resize
  - Advantage – less pixels, better runtime!
  - Disadvantage – data lost
  - Tried several different sizes
    - 500 x 500, 128 x 128
  - Found that 64 x 64 was ideal
  - X_train shape is: (22046, 64, 64, 3)



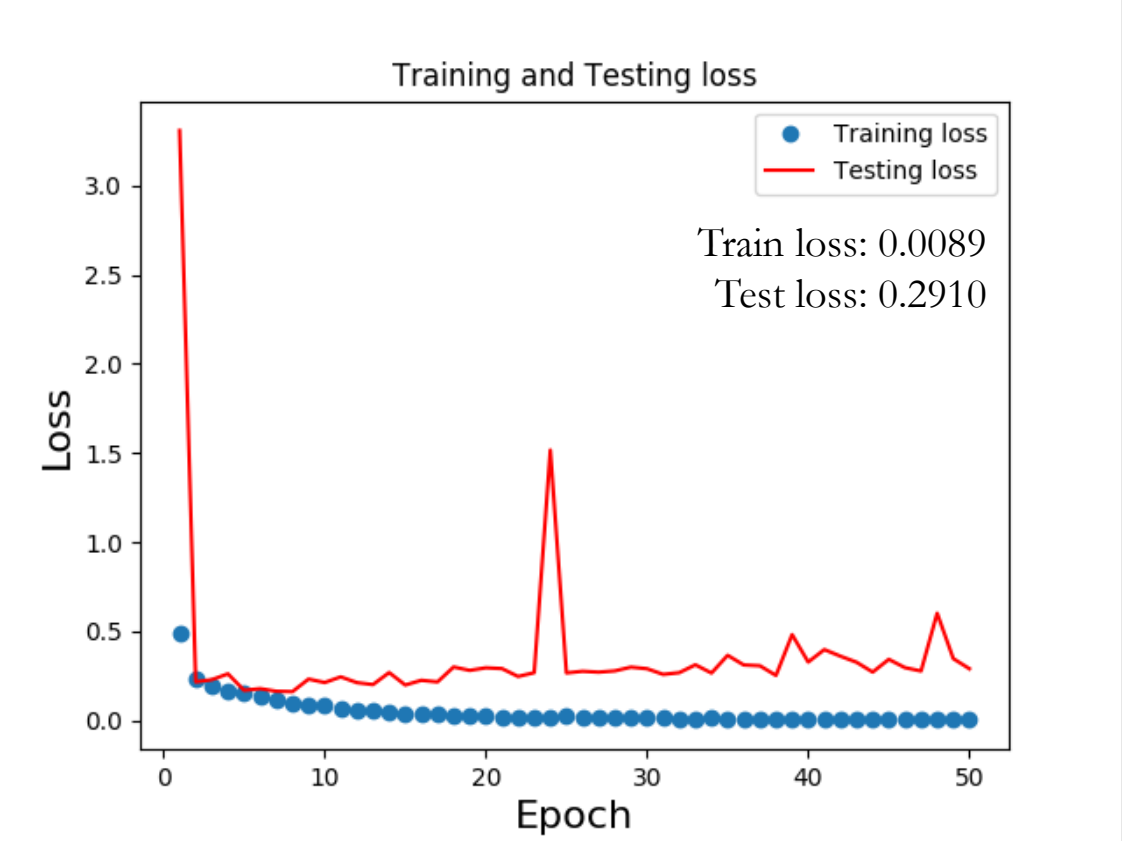
VS.

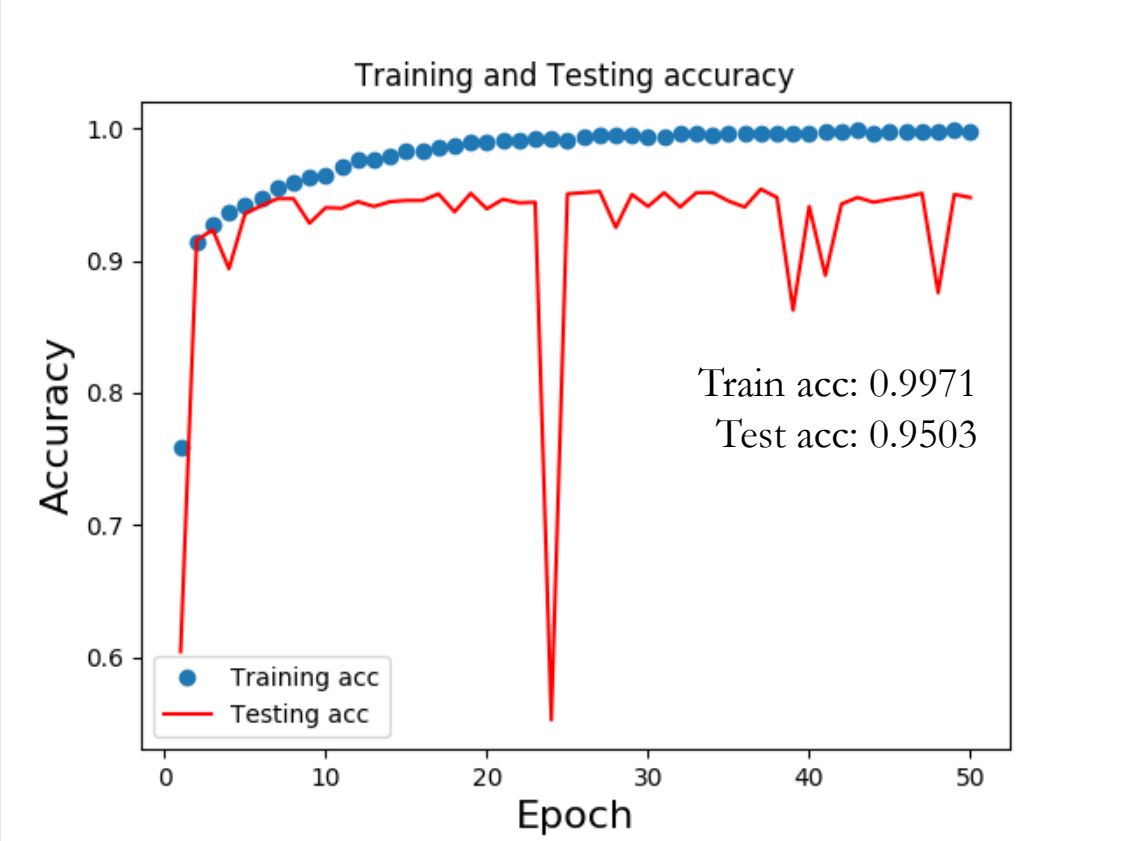
CNN Model

- Convolutional layers:
  - x2
    - Conv2D
    - MaxPool2D
    - BatchNormalization
    - Dropout
- *Flattening*
- Full Connection:
  - x2
    - Dense
    - BatchNormalization
    - Dropout
  - *Dense*

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 62, 62, 32)        896
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 31, 31, 32)        0
_________________________________________________________________
batch_normalization_1 (Batch (None, 31, 31, 32)        128
_________________________________________________________________
dropout_1 (Dropout)          (None, 31, 31, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 29, 29, 32)        9248
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 14, 14, 32)        0
_________________________________________________________________
batch_normalization_2 (Batch (None, 14, 14, 32)        128
_________________________________________________________________
dropout_2 (Dropout)          (None, 14, 14, 32)        0
_________________________________________________________________
flatten_1 (Flatten)          (None, 6272)              0
_________________________________________________________________
dense_1 (Dense)              (None, 512)               3211776
_________________________________________________________________
batch_normalization_3 (Batch (None, 512)               2048
_________________________________________________________________
dropout_3 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_2 (Dense)              (None, 256)               131328
_________________________________________________________________
batch_normalization_4 (Batch (None, 256)               1024
_________________________________________________________________
dropout_4 (Dropout)          (None, 256)               0
_________________________________________________________________
dense_3 (Dense)              (None, 2)                 514
```

Training vs Testing Accuracy & Loss





Data Augmentation

- Manipulate the dataset to generate new data to train on
- Helps our model better generalize!
- ImageDataGenerator

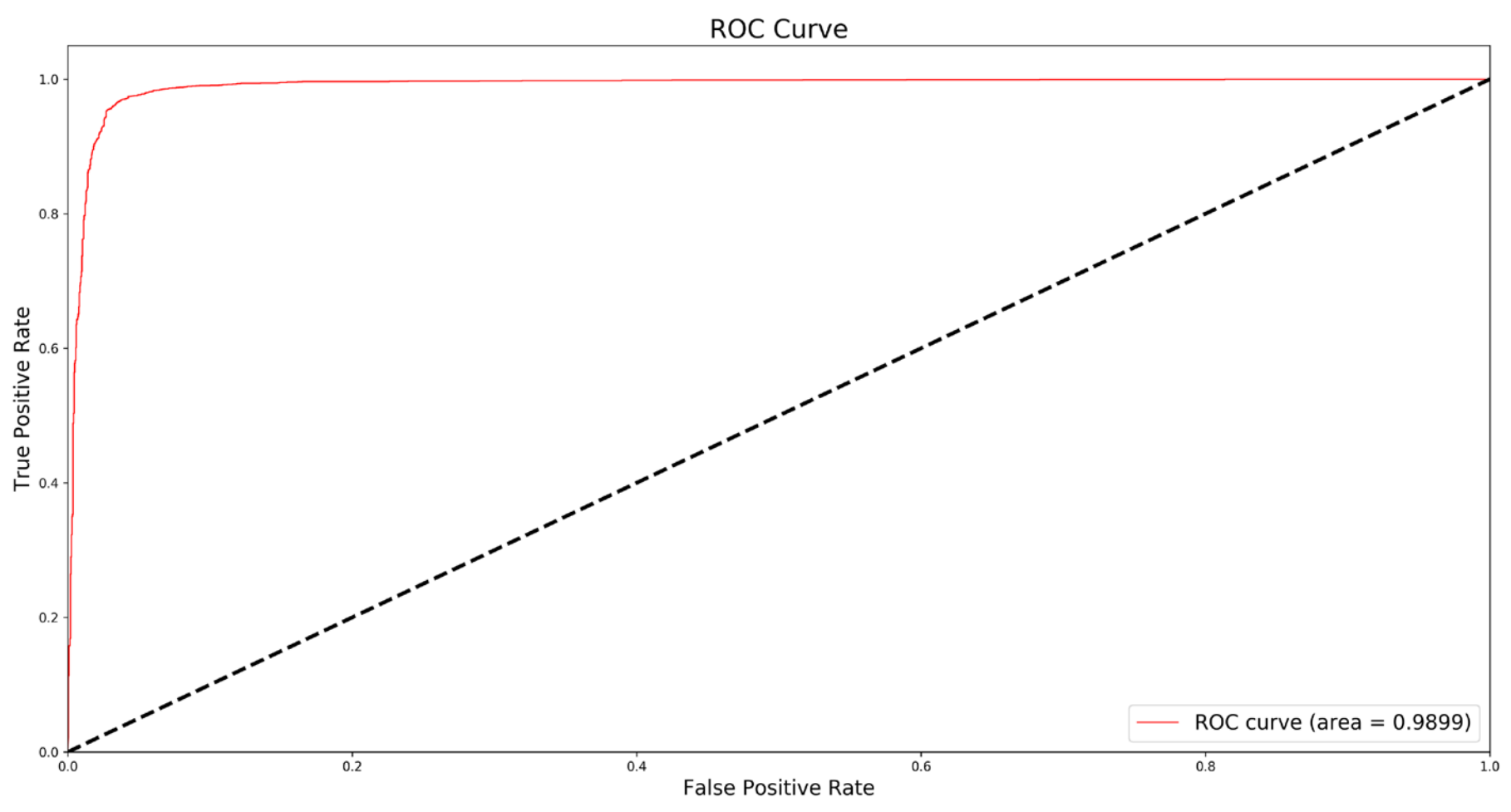
```
train_generator = ImageDataGenerator(rescale = 1/255,
                                     zoom_range = 0.3,
                                     horizontal_flip = True,
                                     rotation_range = 30)
```

Test_Accuracy(before augmentation): 95.03%

Test_Accuracy(after augmentation): 96.44%

Increased by 1.41%!

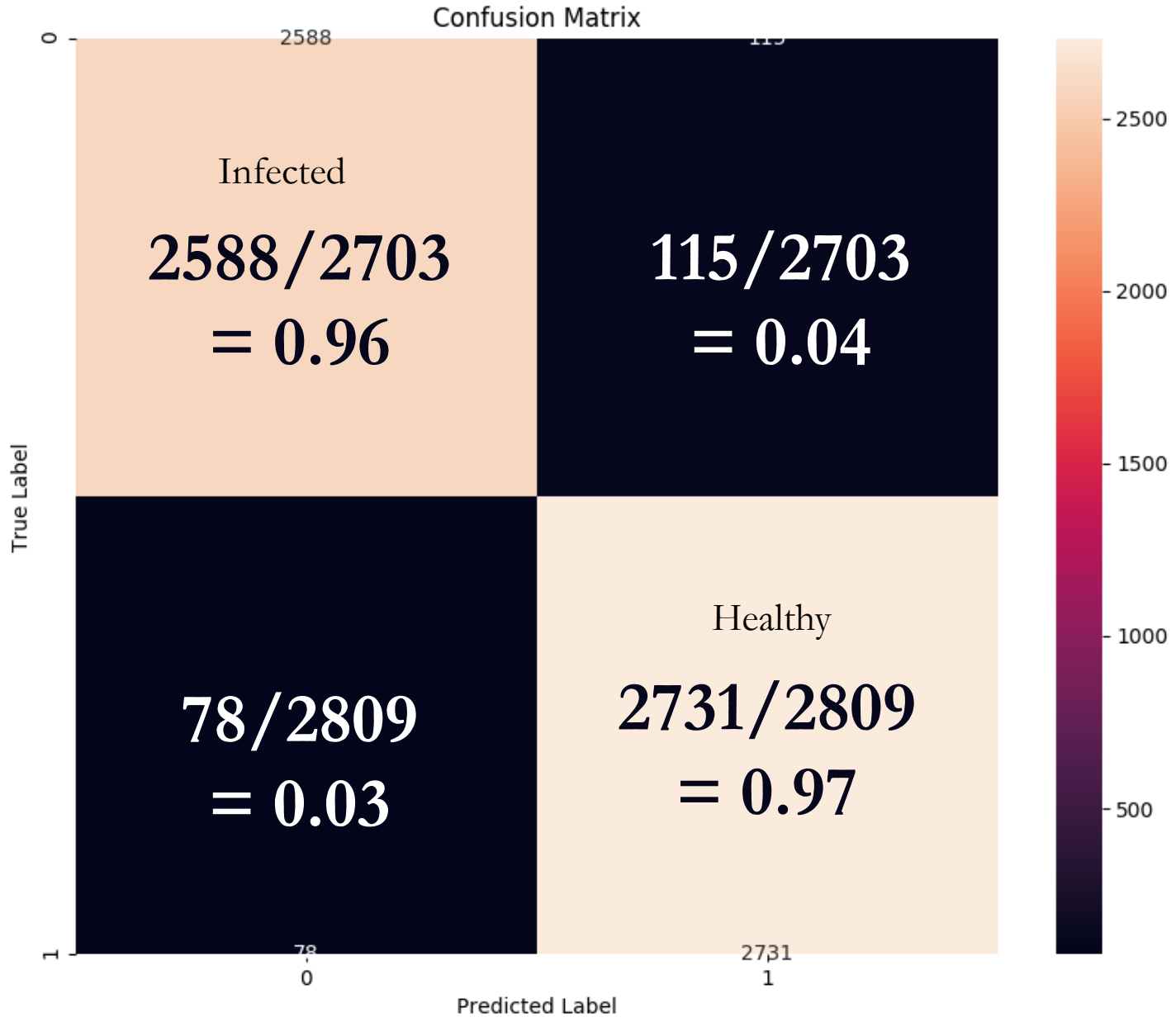
ROC curve

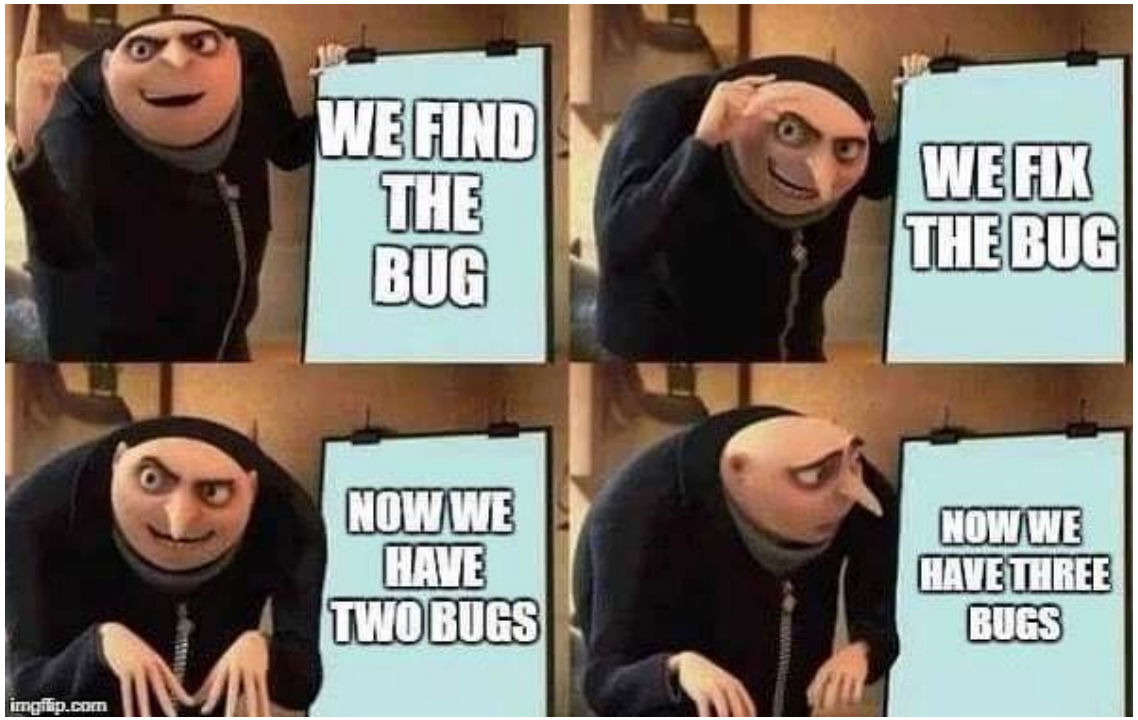

ROC Curve
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
ROC curve (area = 0.9899)

CNN CONFUSION MATRIX

Infected = 0

Healthy = 1





Challenges

- Needed to learn how to process raw images in python!
- Images of all different sizes (and quite large ones)
  - Padding?
  - Resizing?
  - Truncate dataset?
- Relatively large dataset
- Runtime!

Why do we care about this?

ONE COMPACT MODEL FOR END-TO-END MALARIA CLASSIFICATION

TIME SAVED -> MONEY SAVED -> LIVES SAVED

If we had more time...

1. Try the model on another disease dataset (training + testing)
2. Try the model on Malaria dataset from another population
3. Try the model on data collected from more than 200 people
4. Use GridSearchCV to find the best parameters
5. Visualize model filters!

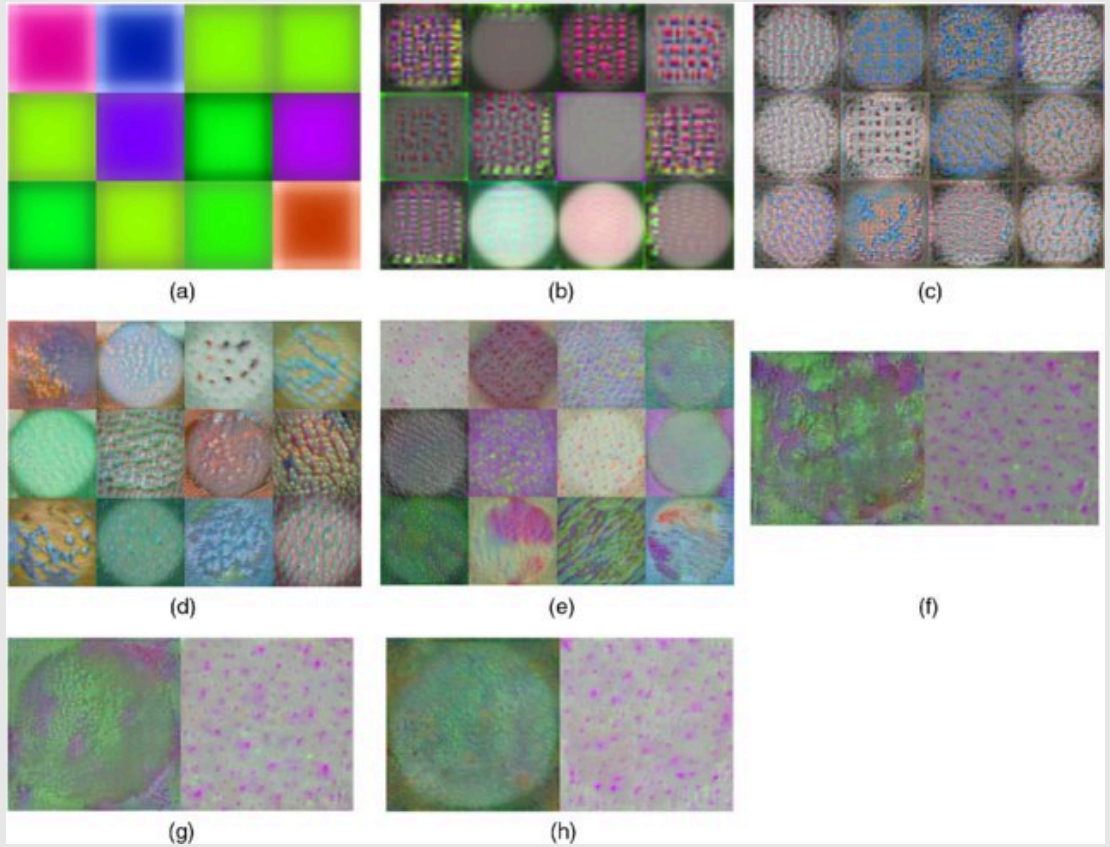


Figure source: Sivaramakrishnan Rajaraman, Journal of Medical Imaging

Sources:

- NIH Malaria Dataset:
  - https://lhncbc.nlm.nih.gov/publication/pub9932
- https://www.researchgate.net/publication/326557171_Understanding_the_learned_behavior_of_customized_convolutional_neural_networks_toward_malaria_parasite_detection_in_thin_blood_smear_images/figures?lo=1
- https://www.pyimagesearch.com/2018/12/03/deep-learning-and-medical-image-analysis-with-keras/?fbclid=IwAR2WrCJK66swdiwQs51TT-dQDR7eGwX3iCniBr0L3aSdRk9MhzLdDcZdO1E
- https://machinelearningmastery.com/image-augmentation-deep-learning-keras/

THANK YOU FOR LISTENING!~ ANY QUESTIONS?