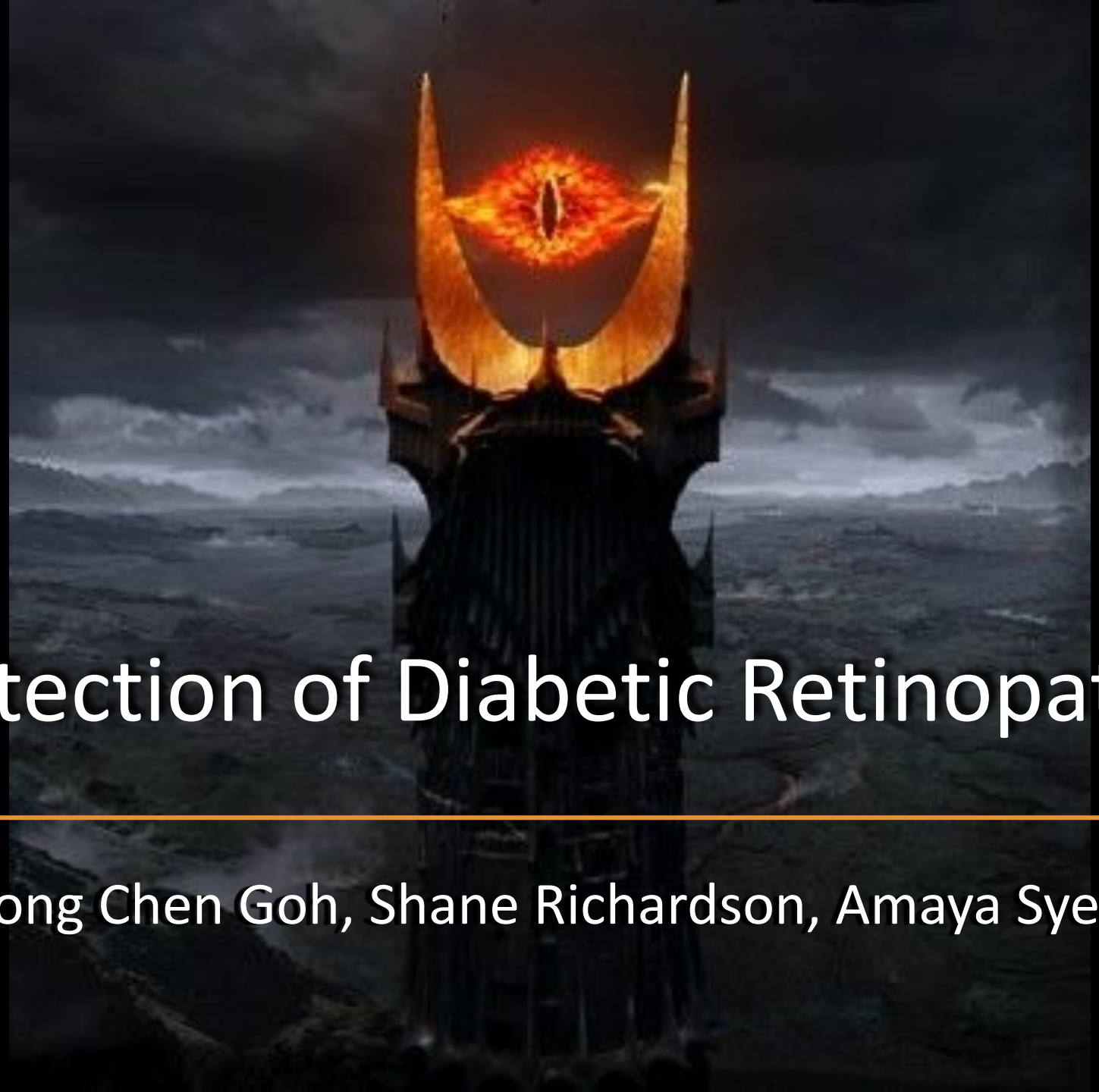


Detection of Diabetic Retinopathy

Yong Chen Goh, Shane Richardson, Amaya Syed

Overview

- Introduction.
- Preparation, training, and testing:
  - Pre-processing (writing a custom data class, image resizing, normalization, and conversion to tensor).
  - Design, training, and testing of our neural network.
- Analysis of results.
- Adversarial attack (only FGSM so far).

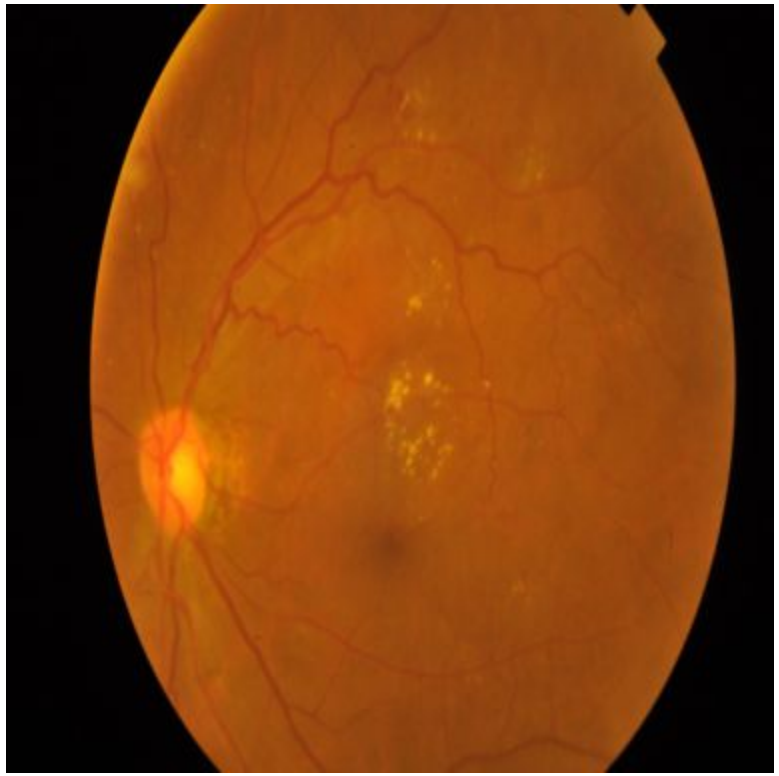
Diabetic Retinopathy

- By 2040 there will be an estimated **642 million people with diabetes**, 80% of whom will suffer for 20 years or more.
- Diabetic Retinopathy is a complication of diabetes that slowly damages the retina and causes blindness if untreated. At least 90% of new cases could be reduced with proper treatment and monitoring of the eyes.
- **Developing an efficient diagnostic algorithm** would be a huge help to clinicians around the world, to unify diagnostic criteria and to prevent patients developing more severe symptoms.

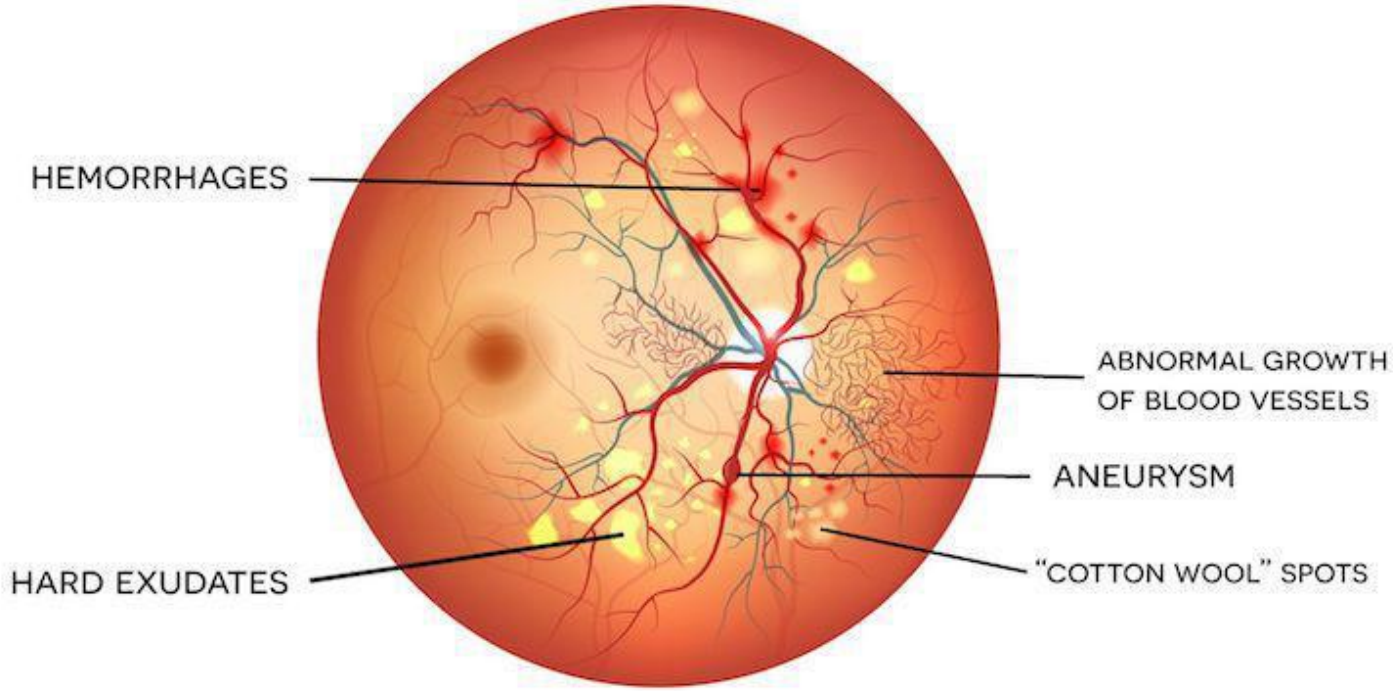
Disease diagnosis and progression

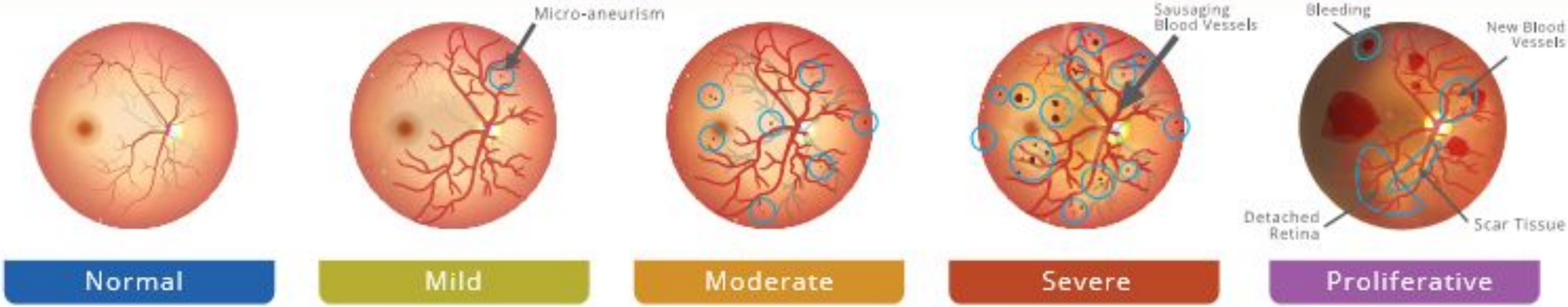
A photograph of the retina is taken and classified into one of 5 categories:

- 0: No sign of disease.
- 1: Incipient signs of disease.
- 2: Disease in progression.
- 3: Disease causing significant loss of sight.
- 4: Loss of sight, either partial or total.

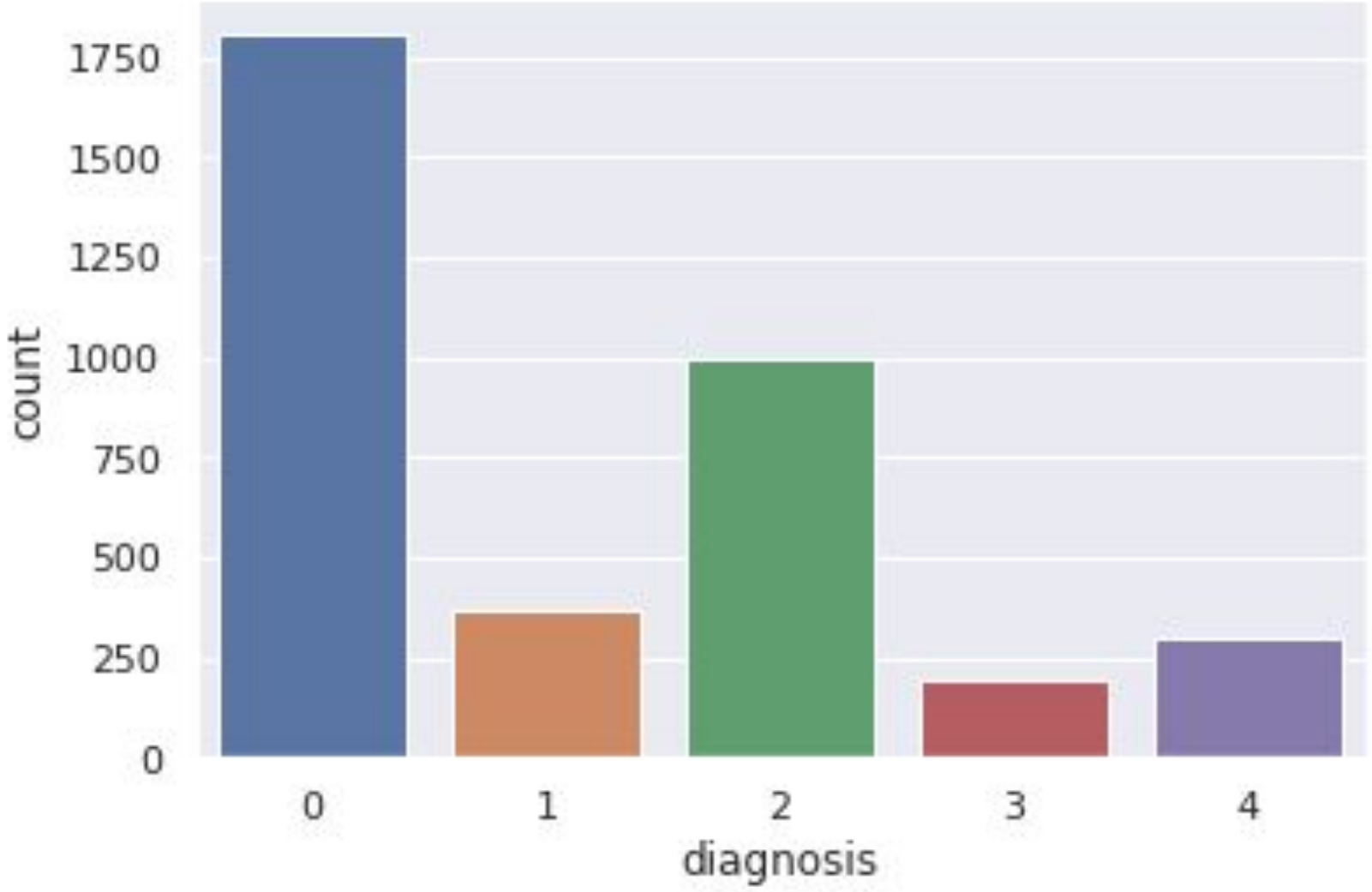


Aptos Blindness detection dataset on Kaggle:https://www.kaggle.com/c/aptos2019-blindness-detection


HEMORRHAGES
ABNORMAL GROWTH
OF BLOOD VESSELS
ANEURYSM
HARD EXUDATES
"COTTON WOOL" SPOTS


Micro-aneurism
Sausaging
Blood Vessels
Bleeding
New Blood
Vessels
Detached
Retina
Scar Tissue
Normal
Mild
Moderate
Severe
Proliferative

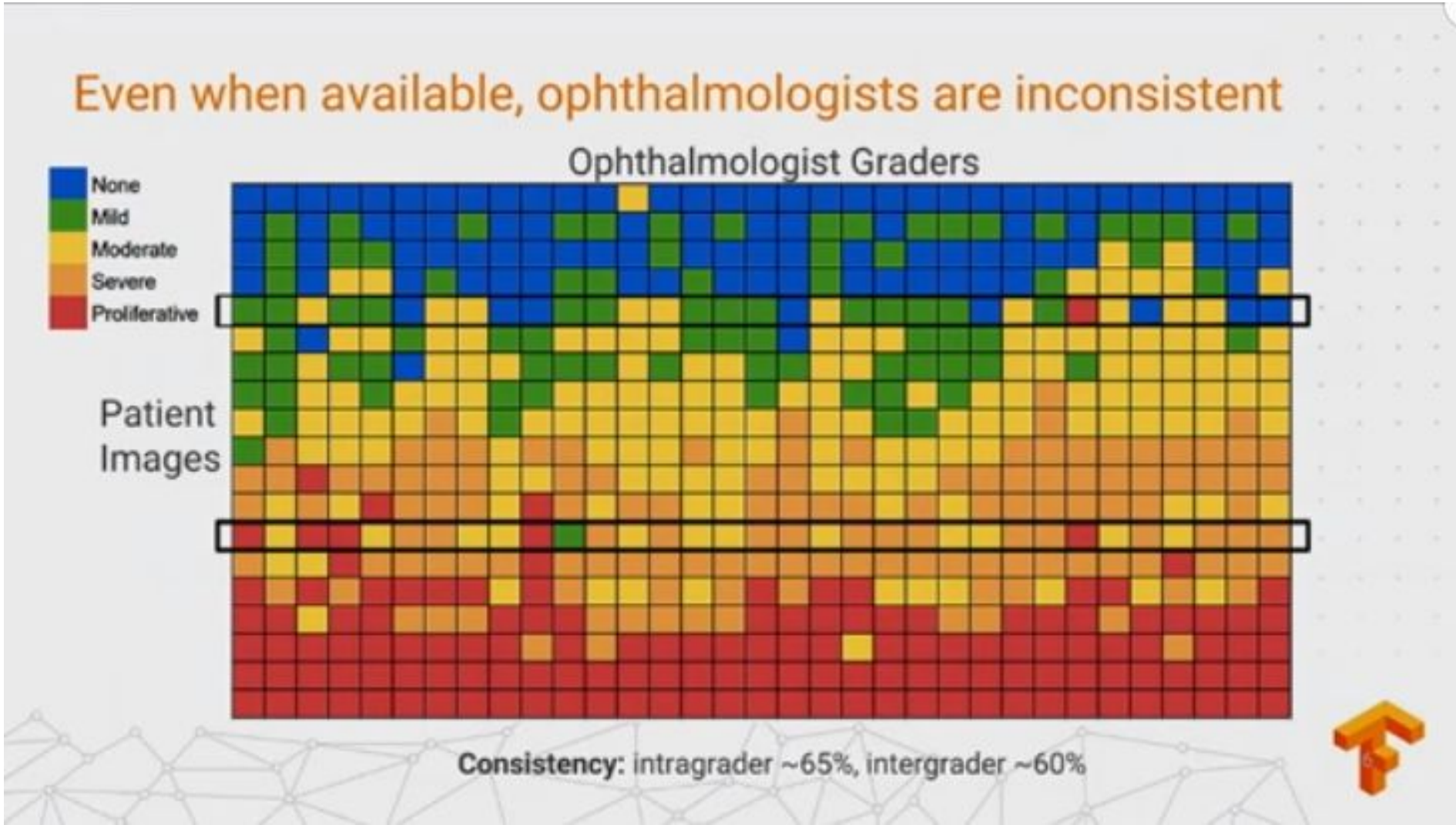
Data distribution



Unbalanced dataset

0	1805
2	999
1	370
4	295
3	193

Inconsistent labelling



Network Structure

Six layers, input [100, 3, 384, 384] ->

- **3 X Convolutional**: Kernel size = 3/3/5, Stride = 1/1/1, features going from [**60 -> 120 -> 240]**
  - Batch Normalization:
  - Relu
  - Maxpooling
- **2 X Linear**: Features going from [**486000 -> 120 -> 60 -> 5]**
  - Relu
- **1 X Softmax + linear output to diagnostic categories [100 x 5]**

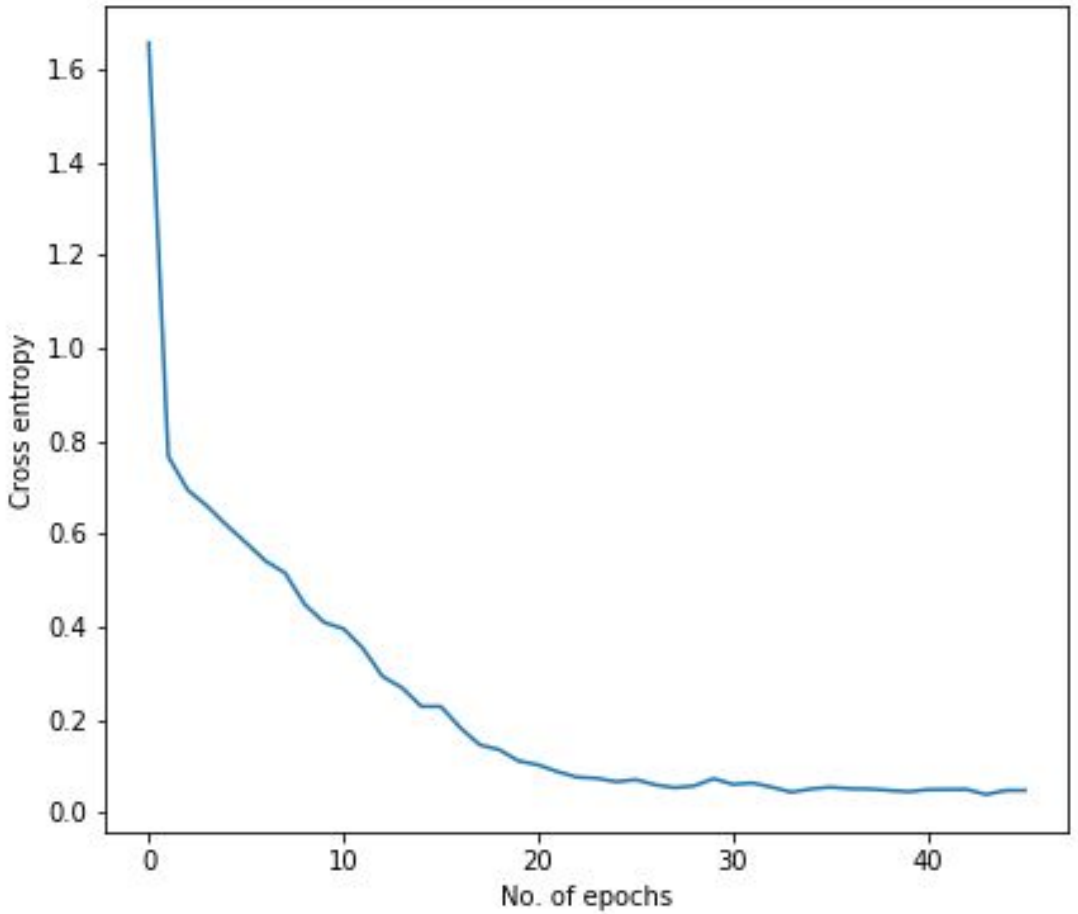
Training loop

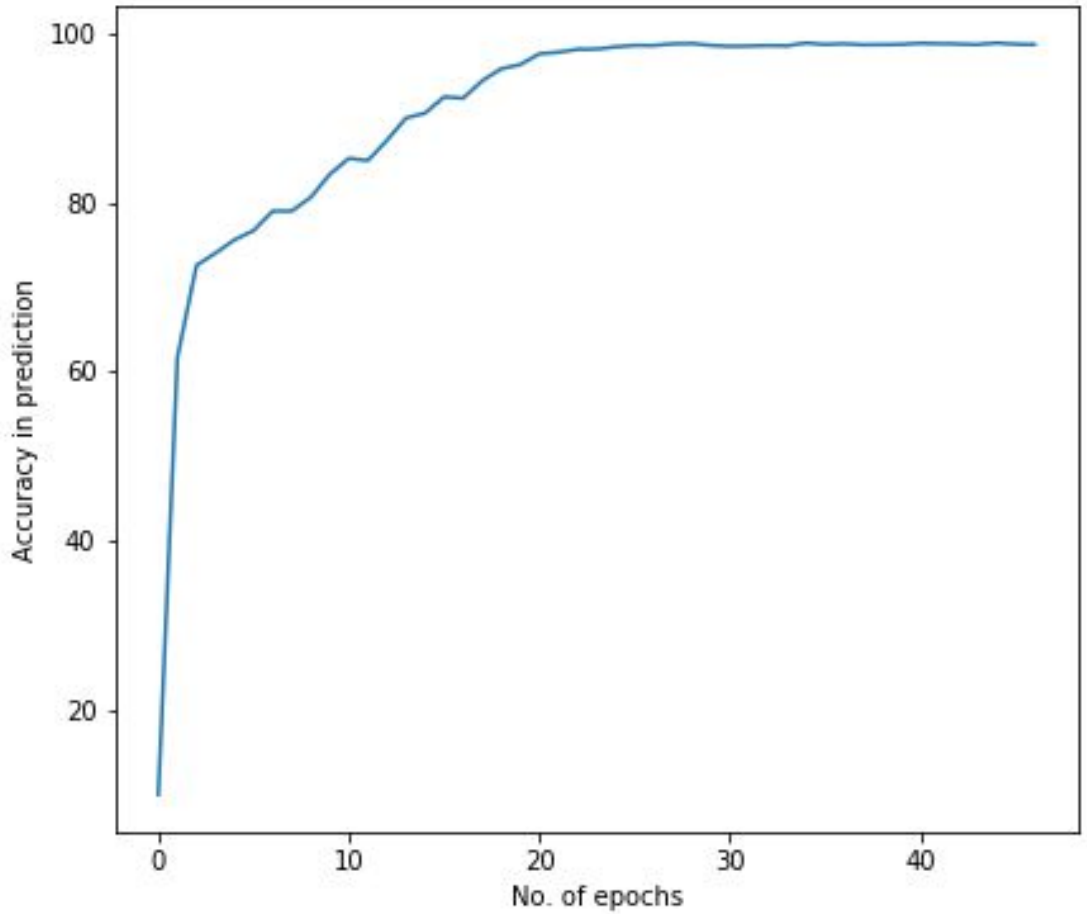
- 45 epochs, with a batch size of 100.
- Optimiser: Adam - adaptive learning rate - 0.0001.
- Loss function: Cross Entropy.
- Batch normalization: Increased accuracy from 72% to 94% in testing when added between the convolutional layers.
- Dropout probability: Implemented for the linear layers, but ultimately rejected because of consistently poor results.

Data augmentation

- The quality of our data is highly variable, with huge variations in resolution, cropping, intensity, luminosity, hue, blurriness, etc.
- We thought pre-processing would be extremely important, but apart from resizing and normalizing our data, bests results were obtained with no pre-processing.
- Data augmentation, however, was the key to successfully minimizing overfitting. We added data with random illuminations, hues, saturations, blurriness, and rotations.

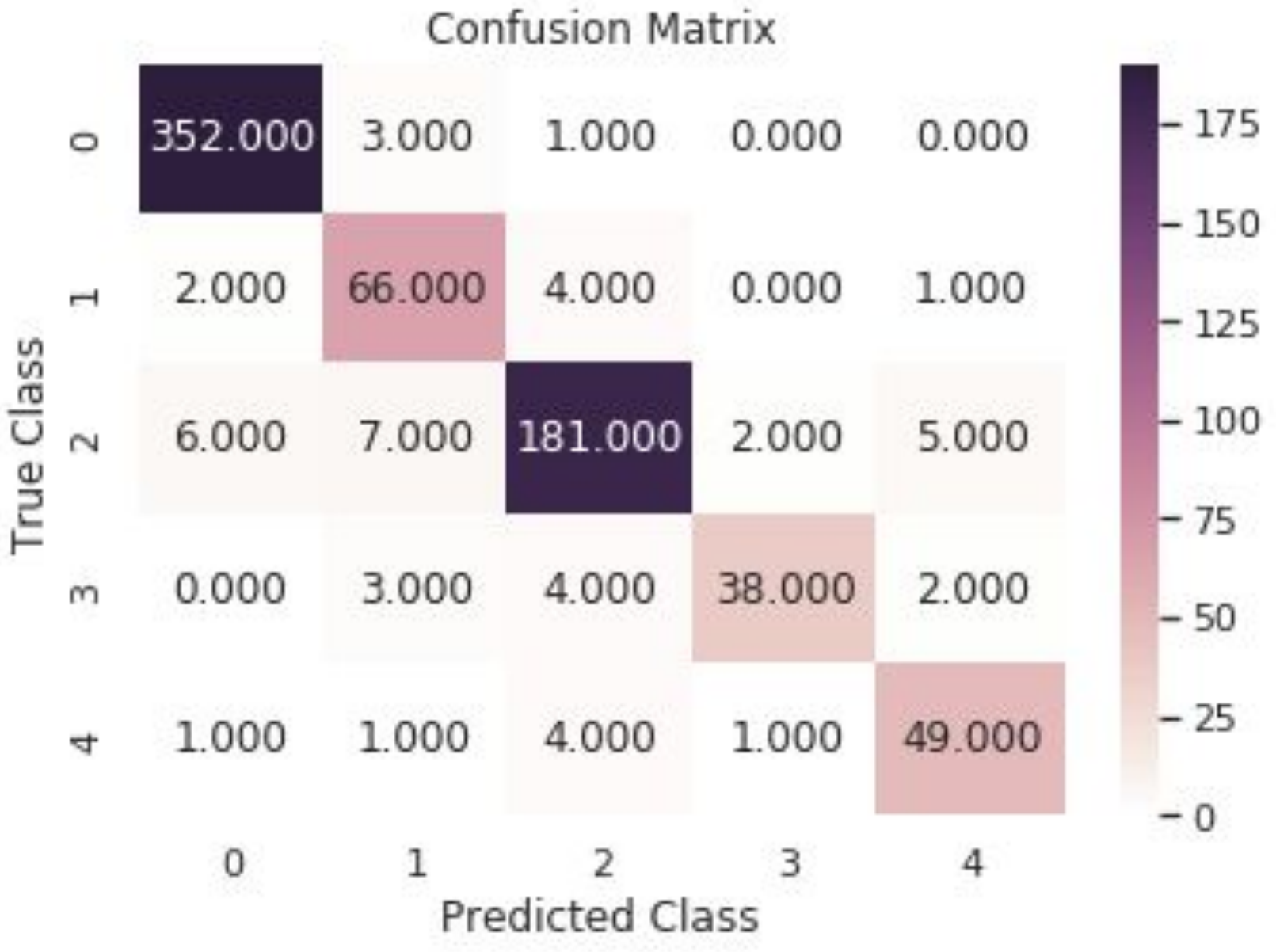
Training results





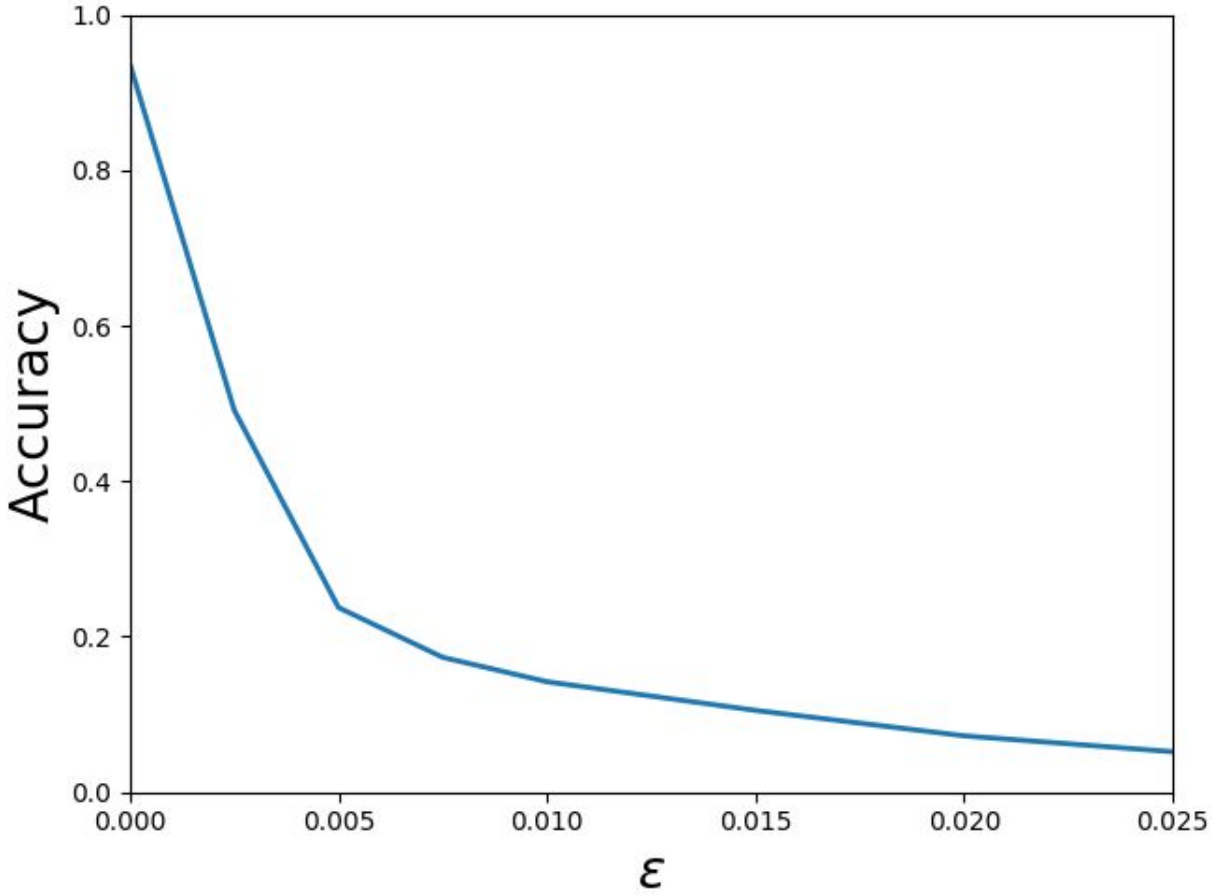
Testing results

We obtained 94% accuracy on our test set, indicating there is still some overfitting.



Adversarial attack - FGSM

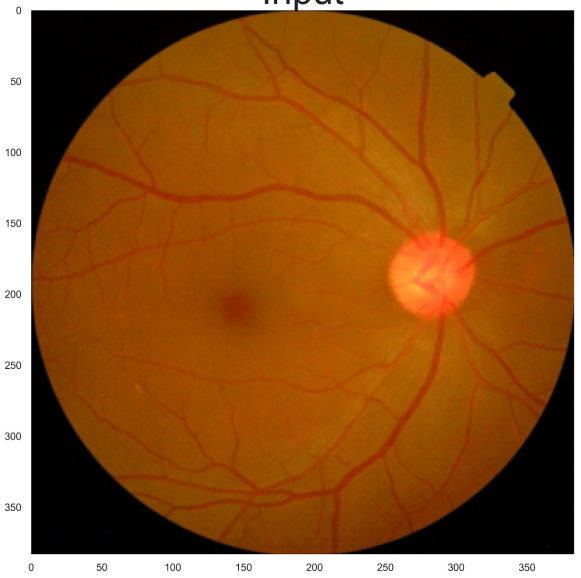
- Accuracy declines rapidly.
- This is almost certainly due in part to the design of our network, but also because of the weak nature of the ‘signal’ our network is looking for.



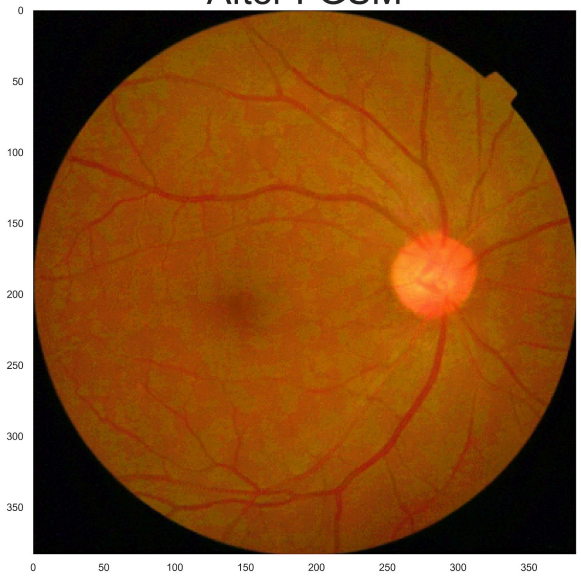
FGSM example

- We've chosen to show an 'extreme' example because it's hard to see much otherwise!
- For this level of perturbation (ε = 0.025), classification accuracy has dropped to ~10%. The noise is visible, but not overpowering, although it is possible to see structure in the difference image.

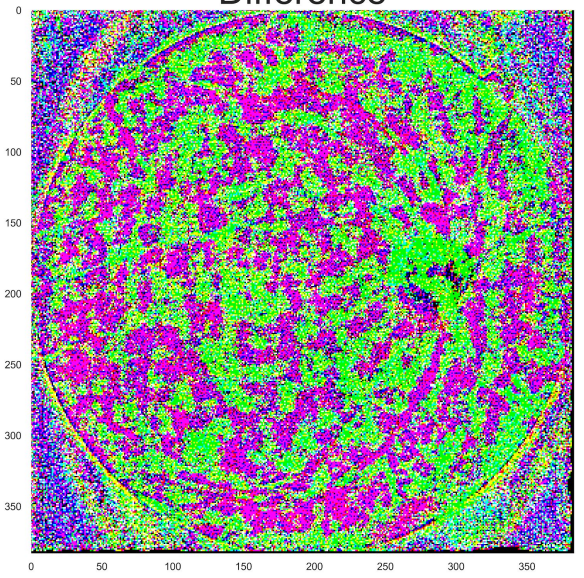
Input



After FGSM



Difference



Conclusions

- With a relatively simple network we achieved classification accuracy of 98.79% on our training set.
- However, despite implementing regularization strategies, the model is still slightly overfitted, as we obtained only 94.60% accuracy on our testing set.
- Classification accuracy goes down very quickly under an FGSM attack. The design of the network does not seem very robust to external perturbations. Yet.

Lots more we could do!

- Experiment with different loss functions.
- Source extra data (from a past Kaggle competition).
- Explore architectures which are more robust to attack.
- Pseudo-labelling to account for class imbalance.
- Selective data augmentation (only boost under-represented classes).
- Implement further adversarial attacks:
    - Universal adversarial perturbation.
    - Single pixel perturbations.

And the Eye of Sauron does indeed see all!

Run through the network and diagnosed as class 0!


Thank you Dave.. Any questions?

References

- https://github.com/rwightman/pytorch-nips2017-attack-example/blob/master/attacks/attack_iterative.py
- https://github.com/StephanZheng/neural-fingerprinting/tree/master/models
- https://www.kaggle.com/c/aptos2019-blindness-detection
- https://www.kaggle.com/ratthachat/aptos-eye-preprocessing-in-diabetic-retinopathy
- https://pytorch.org/tutorials/beginner/data_loading_tutorial.html
- https://pytorch.org/tutorials/beginner/fgsm_tutorial.html
- https://tolkienfacts.wordpress.com/2015/04/22/tolkien-fact-50-the-eye-of-sauron/
- https://towardsdatascience.com/adam-latest-trends-in-deep-learning-optimization-6be9a291375c
- https://towardsdatascience.com/stochastic-gradient-descent-with-momentum-a84097641a5d
- A. Galloway, A. Golubeva, T. Tanay, M. Moussa, G. W. Taylor - Batch Normalization is a Cause of Adversarial Vulnerability - https://openreview.net/forum?id=H1x-3xSKDr
- J. T. Springenberg , A. Dosovitskiy , T. Brox, M. Riedmiller -Striving for simplicity: the all convolutional net - https://arxiv.org/pdf/1412.6806v3.pdf