Image analysis based detection of blindness

Presented by

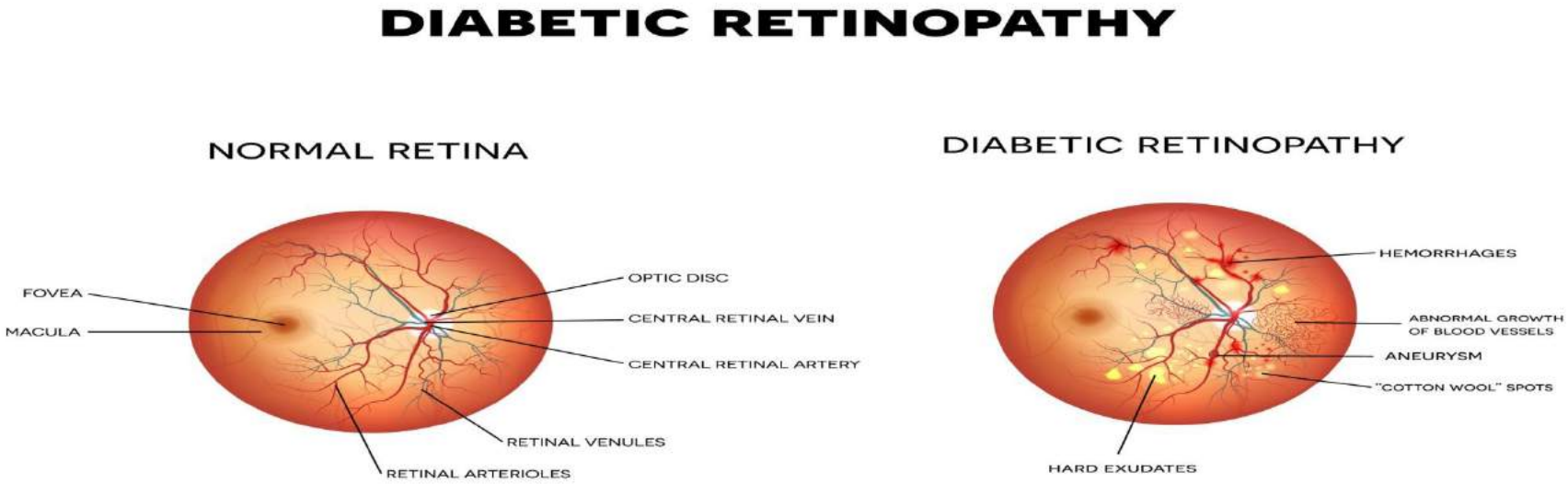
Amar Singh Patel

CSE, 8th sem

Outline

- Objective
- Introduction
- Project Plan
- Model Architecture
- Result
- Conclusion And Future Scope
- References

OBJECTIVE



1) This project had two main goals. The first was to build a convolutional neural network (CNN) from the ground up. A very elaborate approach tested the effects of modulating various parameters on classification accuracy for each layer added in the CNN architecture.

2) The second goal was to demonstrate the power of transfer learning in the context of computer vision tasks. Transfer learning is a valuable tool in deep learning that allows individuals to access the knowledge learned from very complex models trained on millions of labeled images.

INTRODUCTION

Diabetic retinopathy is a complication of diabetes and a leading cause of blindness.

- It occurs when diabetes damages the tiny blood vessels inside the retina, the light-sensitive tissue at the back of the eye.

Damaged blood vessels indicate retinal disease.

- Blood clots indicate diabetic retinopathy.
- Narrow blood vessels indicate Central Retinal Artery Occlusion.

Observation of blood vessels in retinal images

- – Shows presence of disease
- – Helps prevent vision loss by early detection

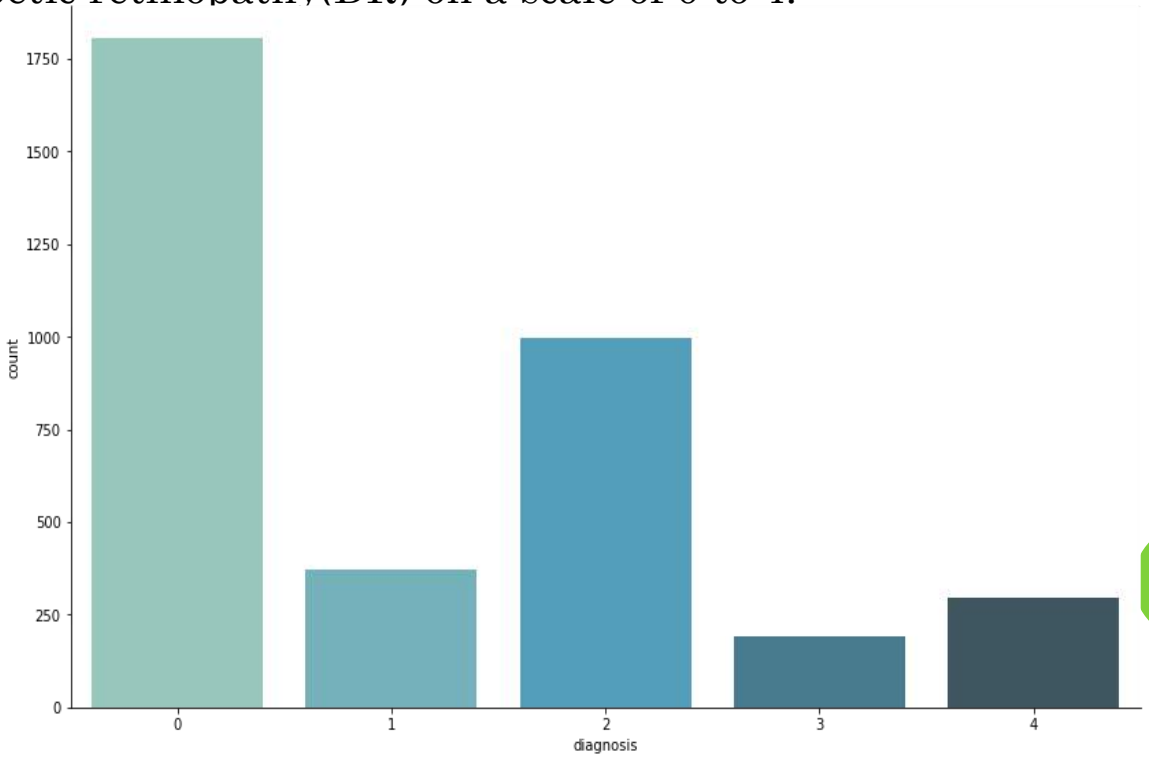
Project Plan

Data Collection (https://www.kaggle.com/c/aptos2019-blindness-detection/data)

- Training set= 3,662 labeled, high resolution color images
- Test set= 1,928 unlabeled images

Within each set images were classified into 5 groups. A clinician has rated each image for the severity of diabetic retinopathy(DR) on a scale of 0 to 4:

- 0 - No DR
- 1 - Mild
- 2 - Moderate
- 3 - Severe
- 4 - Proliferative DR



PROJECT PLAN

Smaple Image:

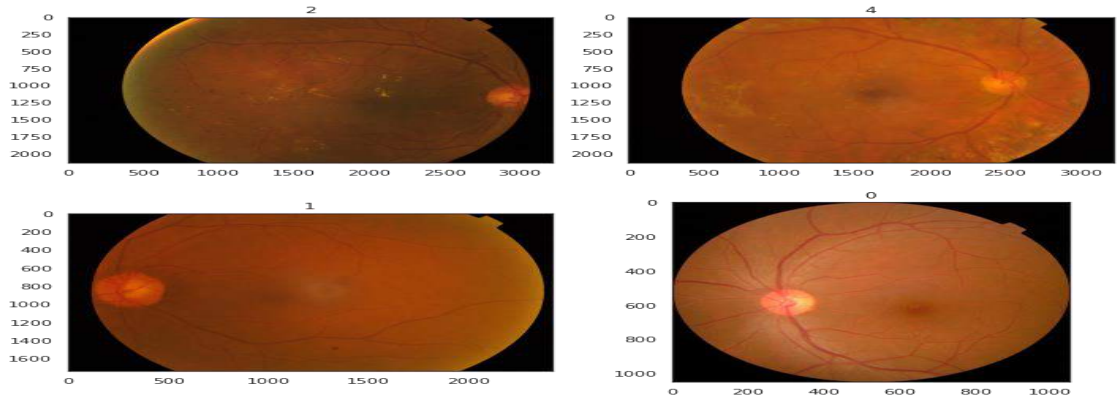


Image processing and analysis

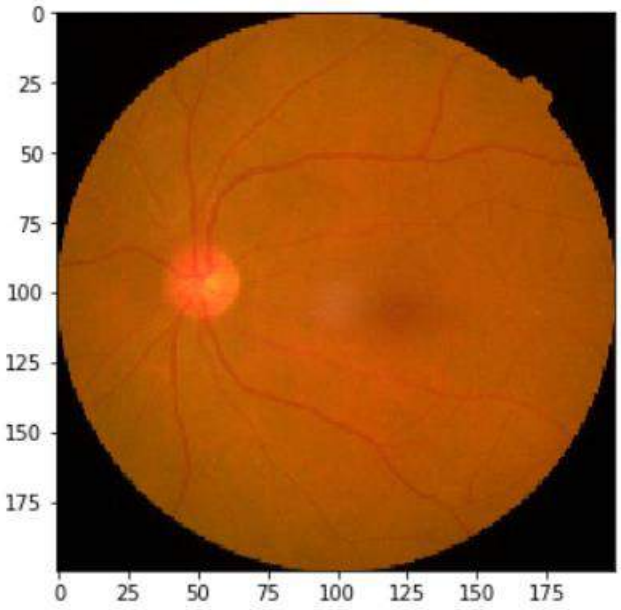
A) Image resize:

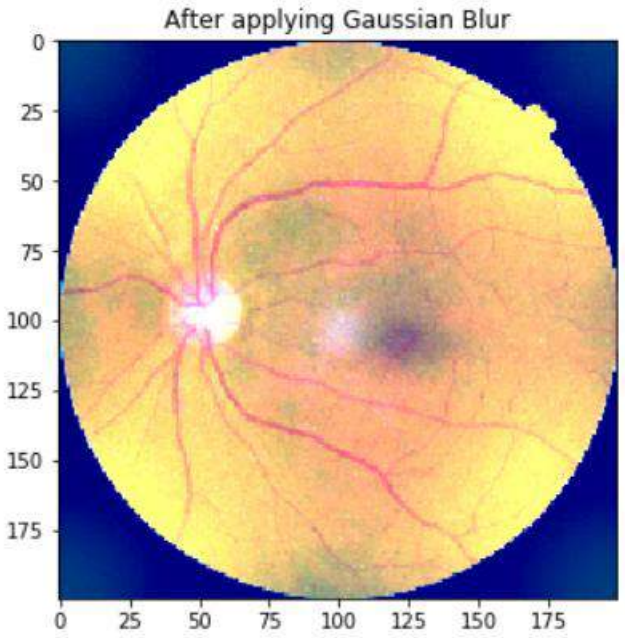
- When performing this task, it is important to keep in mind that there is a balance between computation speed and information loss. To be compatible with the pretrained models used for transfer learning, an image size of 224 x 224 was selected.

Fundus image:

- Fundus photography involves capturing a photograph of the back of the eye i.e. fundus. Specialized fundus cameras that consist of an intricate microscope attached to a flash enabled camera are used in fundus photography

PROJECT PLAN





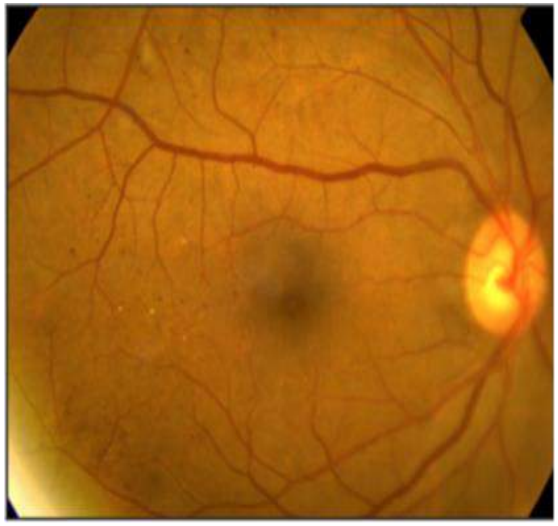
Data Preprocessing — Images

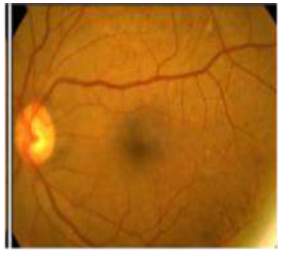
- To adjust for the images and make more clearer images so as to enable the model to learn features more effectively, we will carry out some image processing techniques using **OpenCV** library in python (**cv2**).
- We can apply **Gaussian blur** to bring out distinctive features in the images. In Gaussian Blur operation, the image is convolved with a Gaussian filter which is a low-pass filter that removes the high-frequency components .

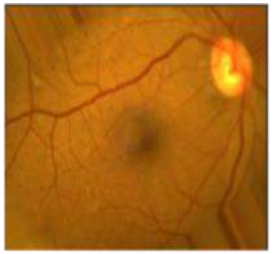
PROJECT PLAN

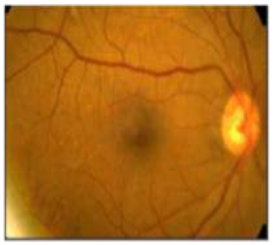
Image augmentations

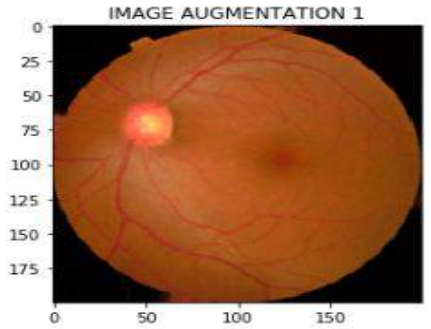
This is one of the most used procedure to generate robustness in the data by creating additional images from the dataset to make it generalize well on new data with rotation flips, cropping, padding etc. using the keras **ImageDataGenerator** class

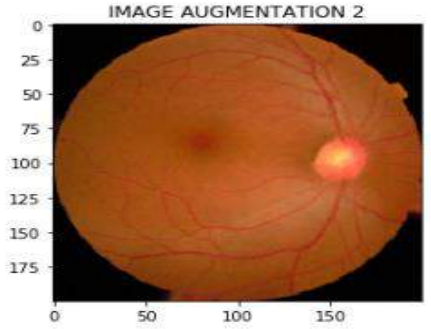


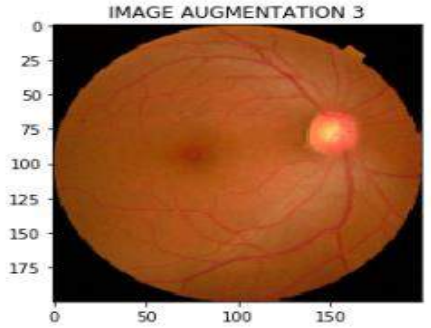










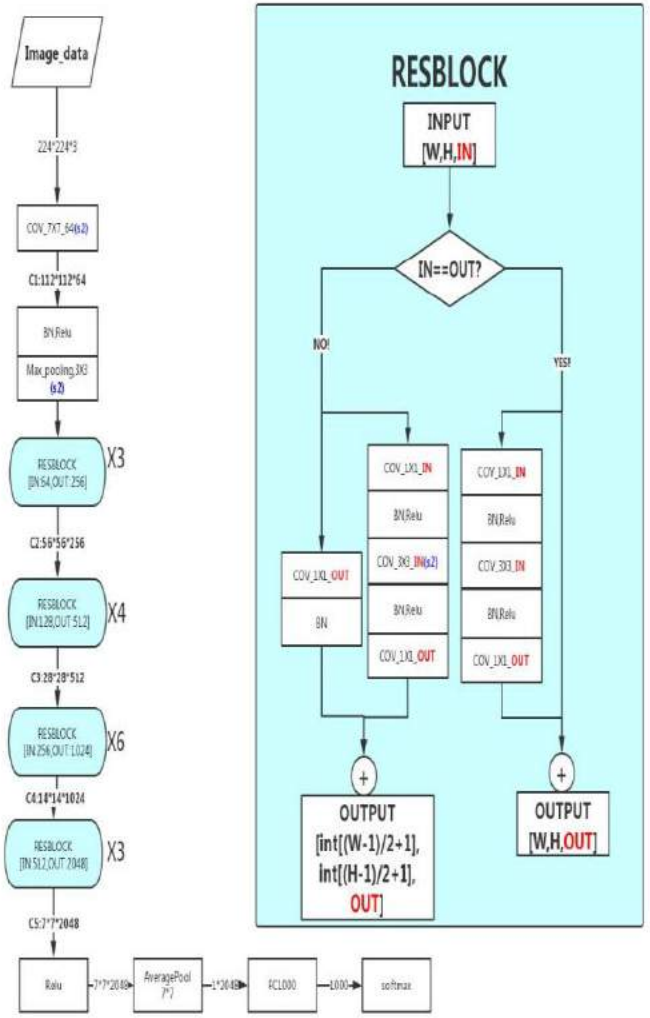


Model Architecture

Model Architecture Used

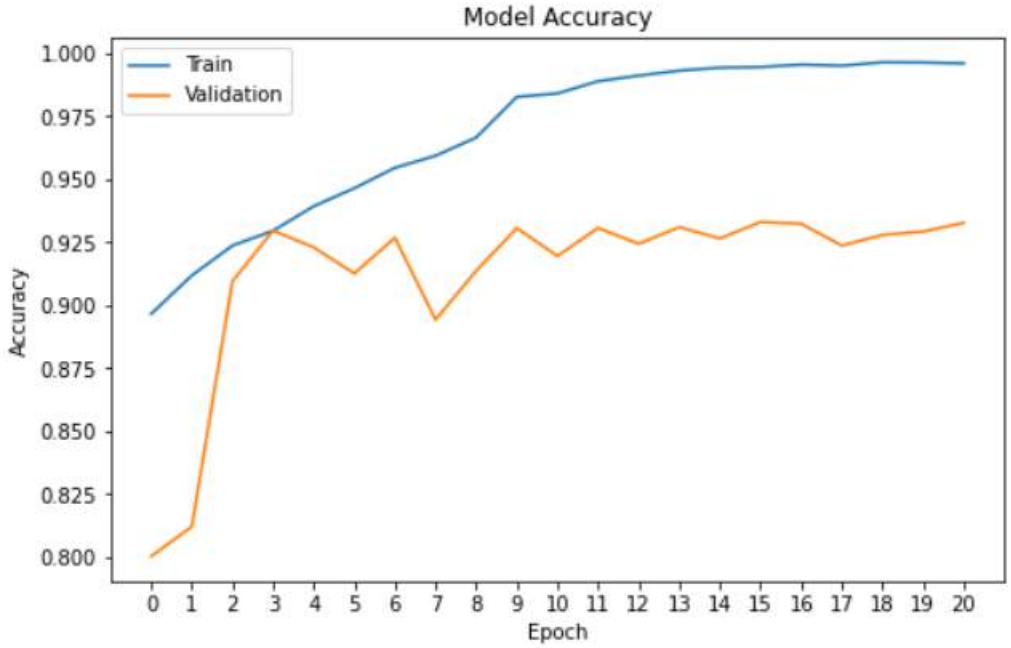
The common Transfer Learning method used when approaching small datasets (with no similarity to ImageNet dataset) is to first use an existing ImageNet weights as Initializers (freezing the first few layers) and then re-training the model.

- We could use similar implementation. A simple **ResNet50** architecture would give good results when used this way



Model Architecture

- We can train the above model for 2–5 epochs (only last 5 layers are trainable which are basically the layers after **ResNet50**).
- Then, we can make all layers Trainable and train the entire model.



- As we can see, only within 20 Epochs, we get good Accuracy Score — Close to
- 95% on Validation dataset.

Model Architecture

1. Convolutional Layer

- The first layer is convolutional layer, this layer performs heavy computation which make further job easy. This layerworks as an input layer taking 128x128x3(i.e. 128 pixels width and height, and 3 because images have depth 3, the color channels) as the input size of the image. Filters of 3x3 matrix will slide over all the spatial locations. The convolutional layer comprises a set of independent filters. Each filter is independently convolved with the image resulting in 32 feature maps.

2. Max-pooling Layer

- In Max-pooling layer highest weighted feature is extracted, this is achieved by converting above 3x3 matrix in more compressed matrix. The above 3x3 matrix is the converted into 2x2 matrix which involves only the highest weighted feature that is present in 3x3 matrix.

3.Flatten Layer

- Flatten layer converts the matrix of image into one single dimension array which acts as an input to the dense layer.

Model Architecture

4. Dropout Layer

- The dropout layer performs inexpensive and powerful operation that highly improve generalization abilities of the neural network. This method involves randomly removing and restoring neurons during the training with a probability determined by the hyperparameter called dropout rate.

5. Dense Layer

- The fully connected layer has its neurons connected to all neurons in the previous layer. These layers are used as last elements of deep neural classifier, which are feed by the features extracted by the successive convolutional layers.

6. Output layer

- The last layer that produces the output of the network is a softmax layer or sigmoid neuron, depending on the solving task - binary or multiclass classification.

Model Architecture

Activation Functions

1. ReLU:

- Rectified Linear Unit (ReLU) is the most used activation function in the world right now. Since, it is used in almost all the convolutional neural networks or deep learning. As you can see in the above figure, the ReLU is half rectified (from bottom). f(z) is zero when z is less than zero and f(z) is equal to z when z is above or equal to zero.

2. Softmax:

- The softmax function is a more generalized logistic activation function which is used for multiclass classification.

Model Architecture

Selecting activation and optimizing functions

- The first step in the process of building my CNN from scratch was to determine which optimizers and activations functions performed best on the task at hand.
- As a summary, the softmax activation function outperformed the rest, while the optimizers adam and adamax outperformed the others.

	Adam	Adamax
relu	A -0.72 V-0.71	A-0.69 V-0.73
softmax	A-0.96 V-0.72	A-0.90 V-0.70

Model Architecture

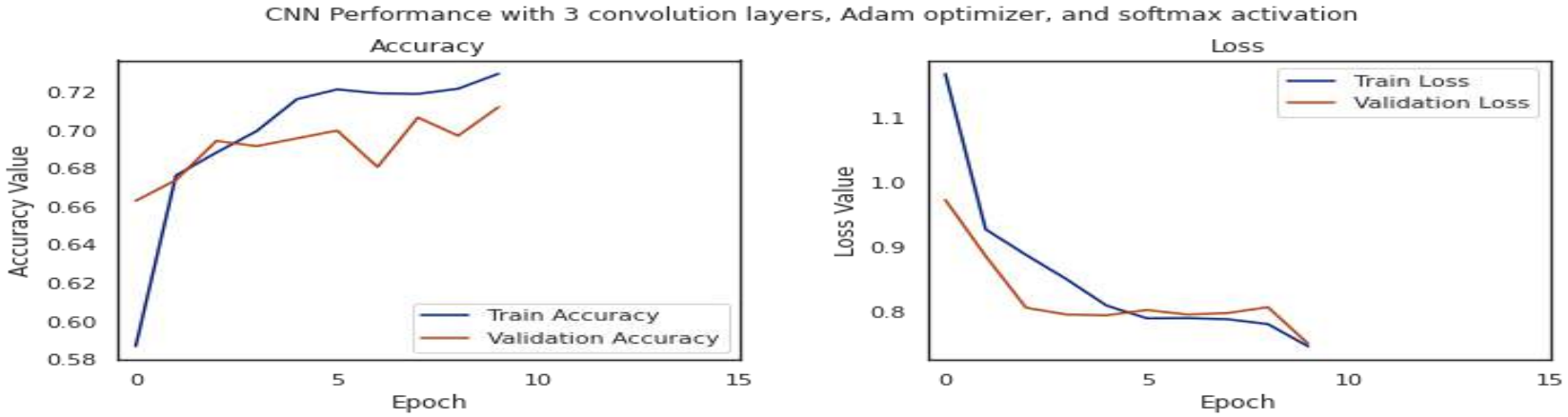
Determining optimal number of convolutional layers

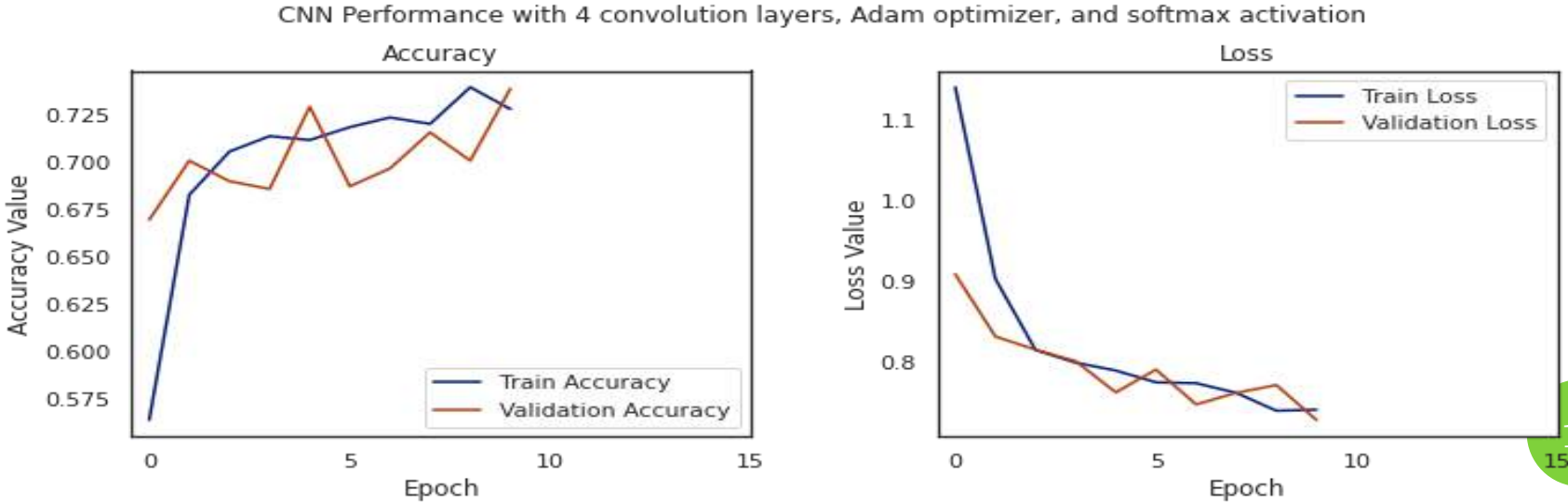
After selecting the top optimizer and activation function, the next step was to determine how many layers to add in the convolutional base. Results from models containing 3 to 6 convolution layers are presented below.

At the end of the 3-layered model, both the accuracy and loss seem to be suggesting that with additional epochs, a greater performance would be expected

No. of Layers	Accuracy	Loss	Validation accuracy	Validation Loss
3	0.74	0.60	0.73	0.66
4	0.72	0.62	0.73	0.60
5	0.73	0.60	0.72	0.67
6	0.70	0.62	0.67	0.60

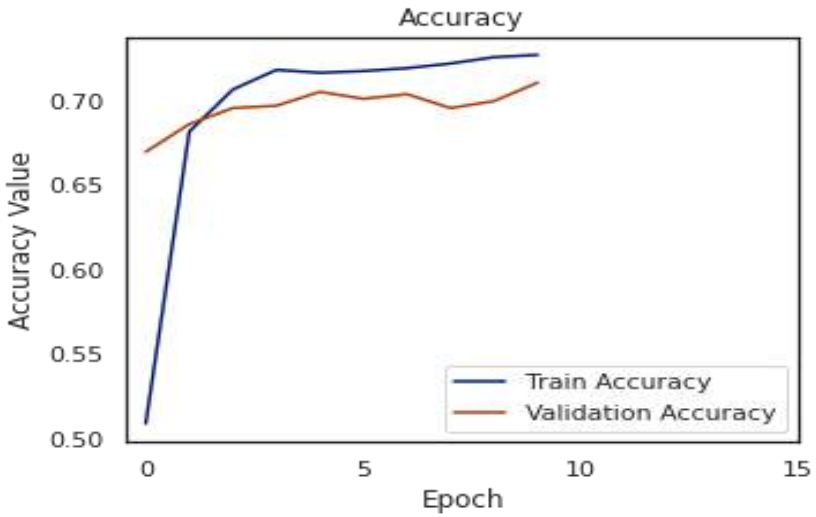
Model Architecture

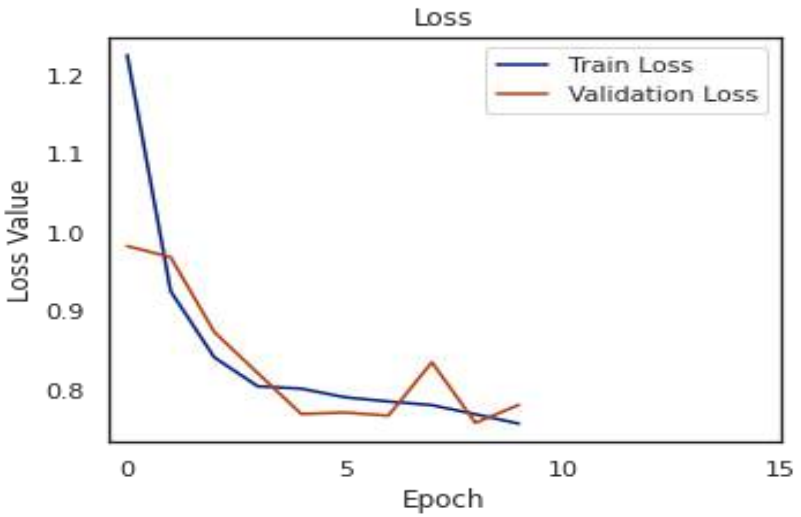




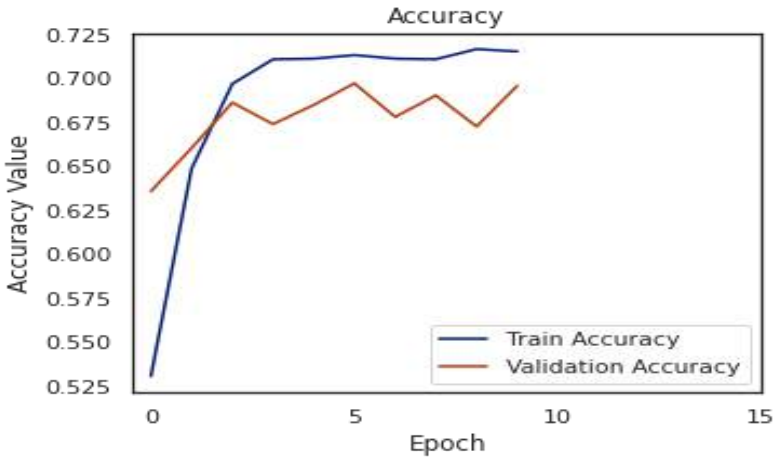
Model Architecture

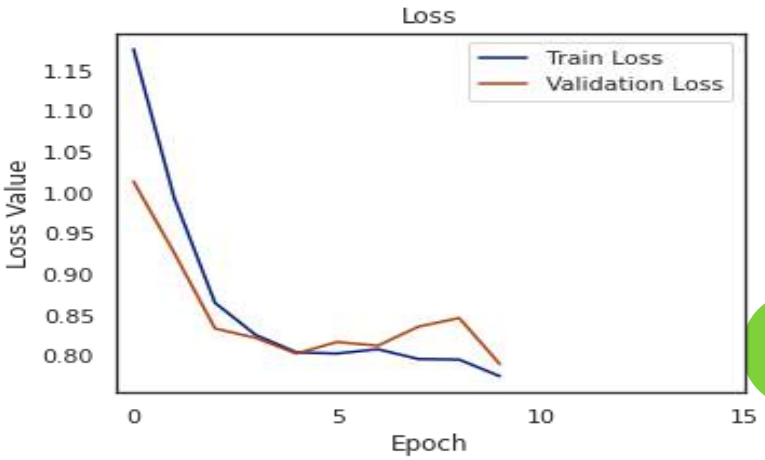
CNN Performance with 5 convolution layers, Adam optimizer, and softmax activation





CNN Performance with 6 convolution layers, Adam optimizer, and softmax activation

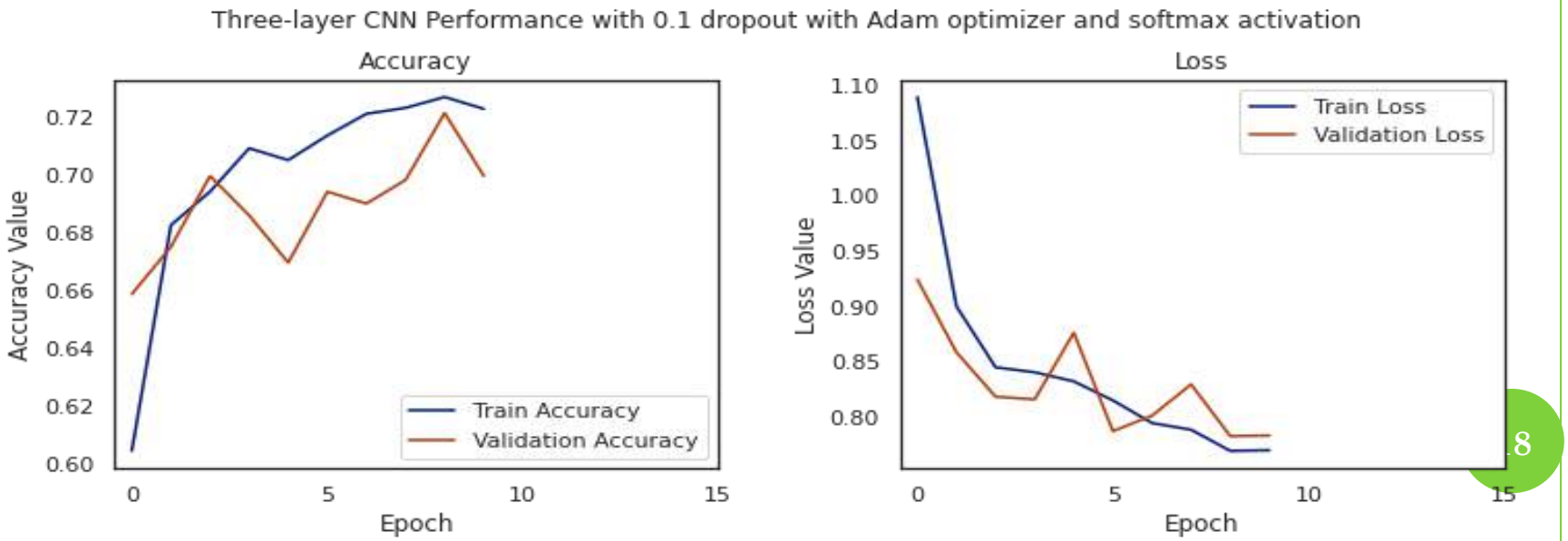




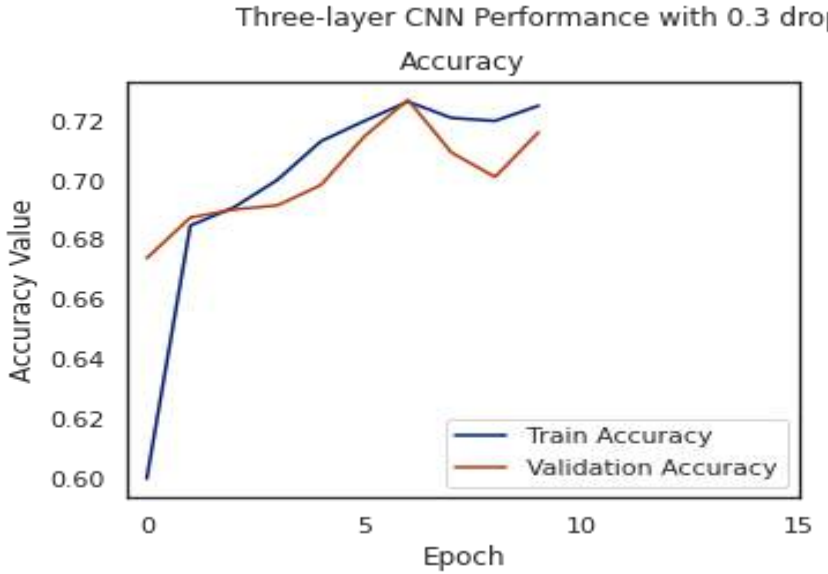
Model Architecture

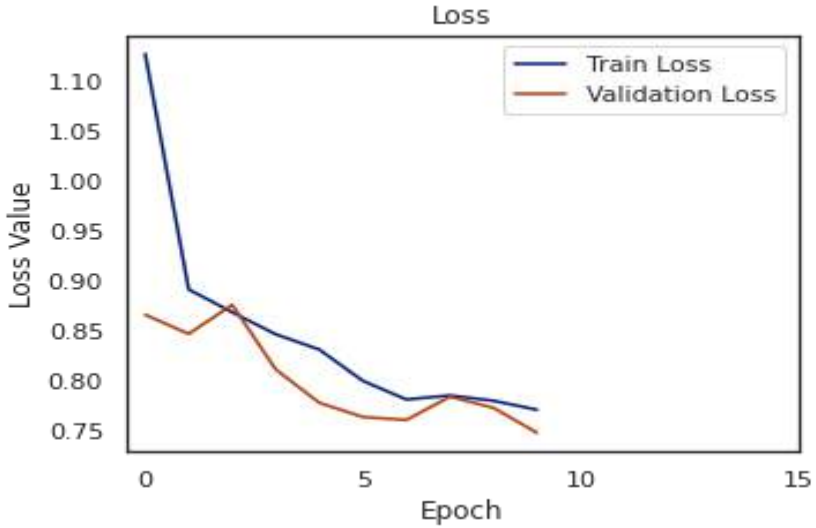
Refining the model using dropout and batch normalization

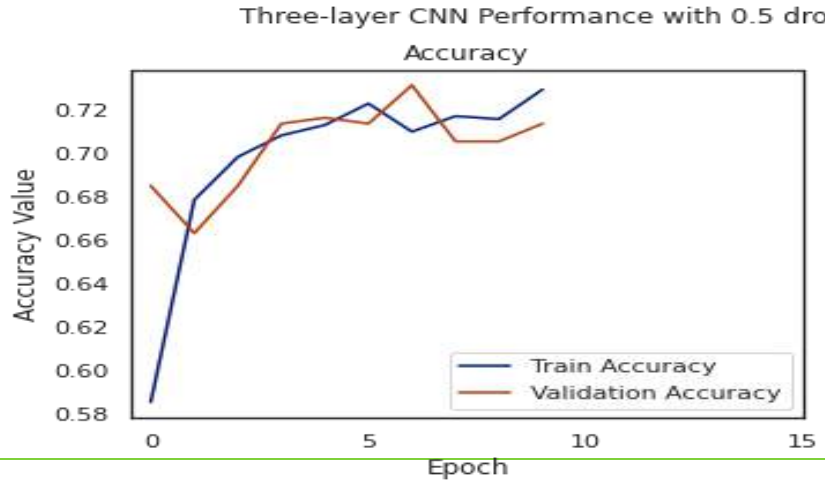
- Up to this point we have maximized the image classification model as much as possible. Although it is not an outstanding model, the fact that it performs equally well on the validation set suggests that it may be generalizable. Although the model does not appear to be overfitting, manipulations of dropout (10%, 30%, 50%) and batch normalization were explored to see if they offered any performance advantage.

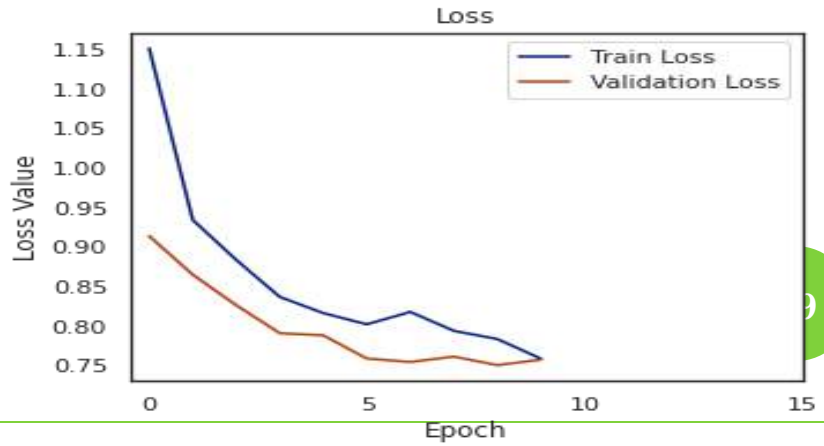


Model Architecture









Model Architecture

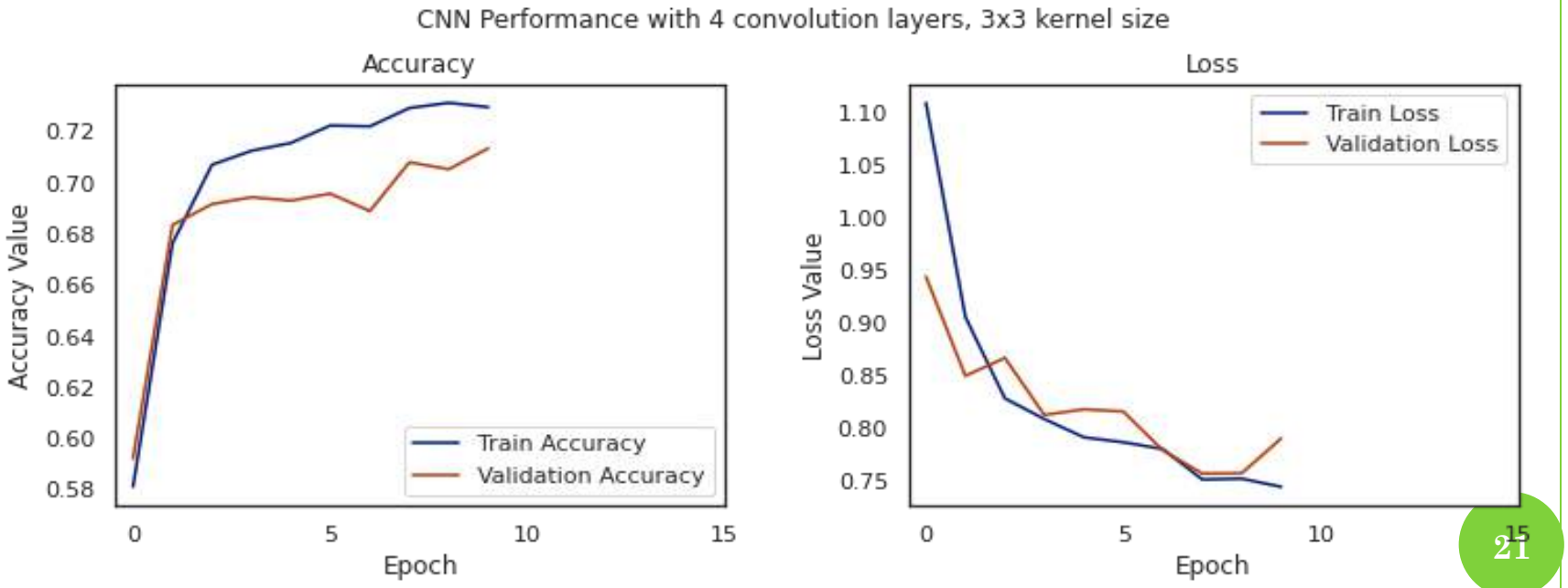
Refining the model using dropout and batch normalization

Dropout	Accuracy	Loss	Validation accuracy	Validation Loss
10%	0.72	0.76	0.69	0.77
30%	0.73	0.69	0.71	0.70
50%	0.73	0.76	0.72	0.76

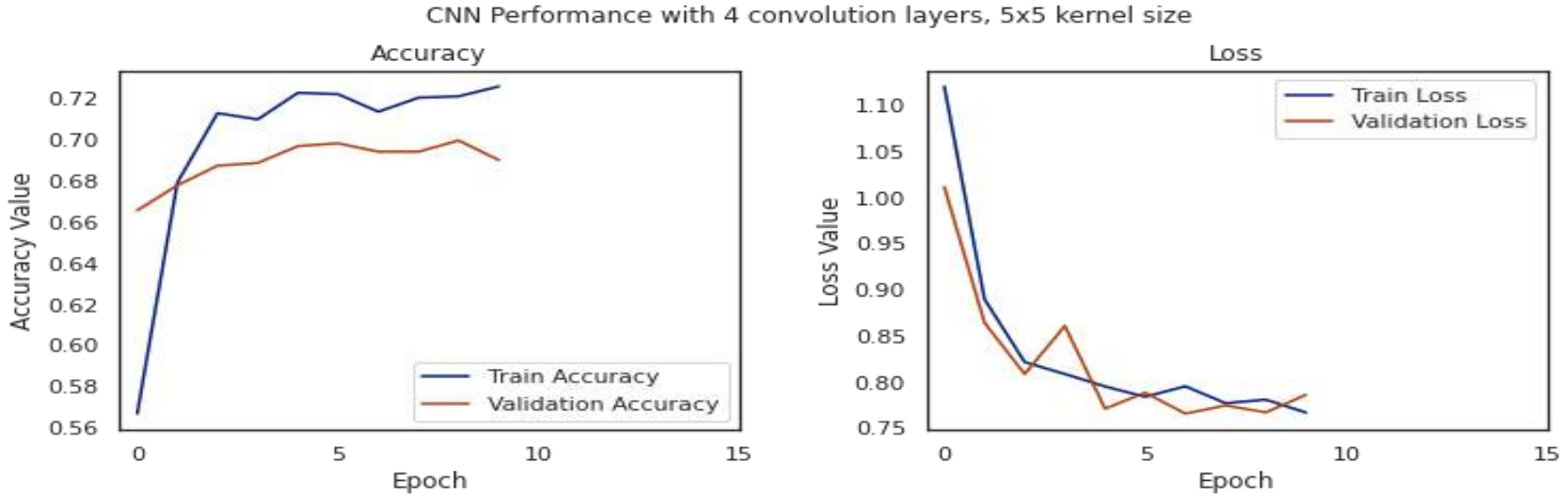
Model Architecture

Evaluating the role of kernel size in model performance

- With a decently performing model, next the influence of kernel size on performance was evaluated. The results from comparing 3 x 3, 5 x 5. Toward the end of the 5 x 5 kernel sizes both accuracy and loss appear to be plateauing.



RESULT



Kernel Size	Accuracy	Loss	Validation accuracy	Validation Loss
3*3	0.73	0.75	0.71	0.80
5*5	0.73	0.69	0.69	0.68

RESULT

Evaluation metric (Quadratic weighted kappa)

- **Quadratic weighted kappa** measures the **agreement between two ratings**. This metric typically varies from 0 (random agreement between raters) to 1 (complete agreement between raters). In the event that there is less agreement between the raters than expected by chance, this metric may go below 0. The quadratic weighted kappa is calculated between the scores assigned by the **human rater** and the **predicted scores**.

Quadratic Weighted Kappa

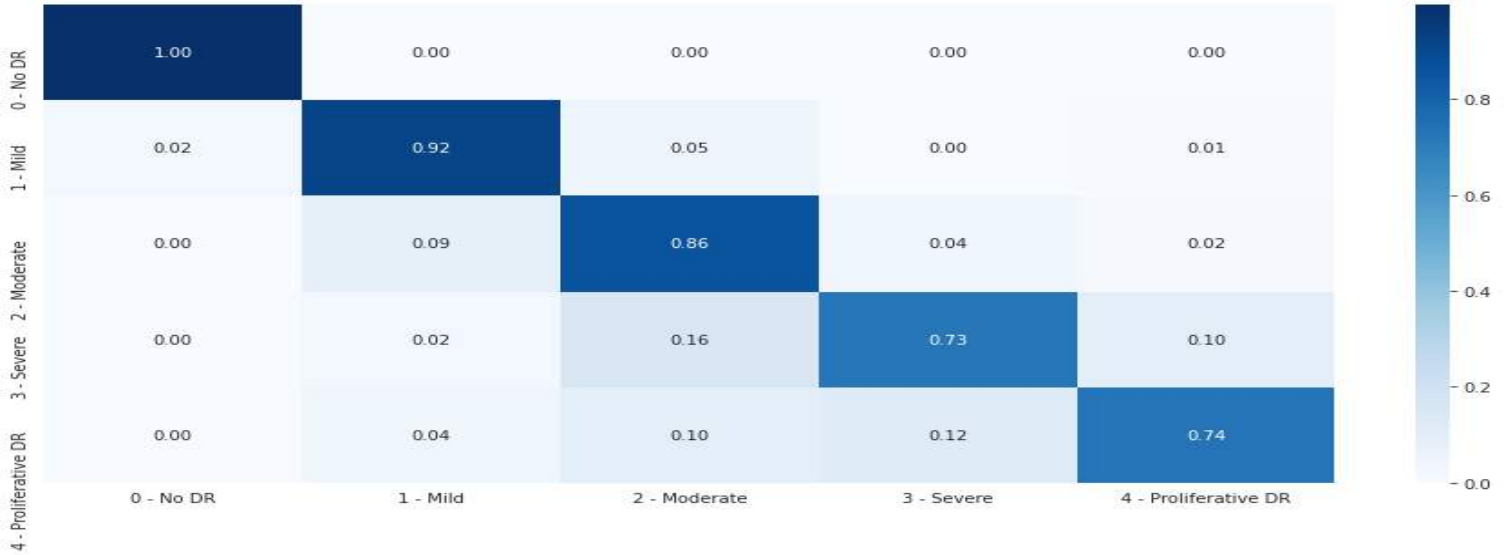
Hide code

In [28]:
```
print("Train Cohen Kappa score: %.3f" % cohen_kappa_score(train_preds, train['diagnosis'].astype('int'), weights='quadratic'))
```

```
Train Cohen Kappa score: 0.952
```

RESULT

Confusion Matrix



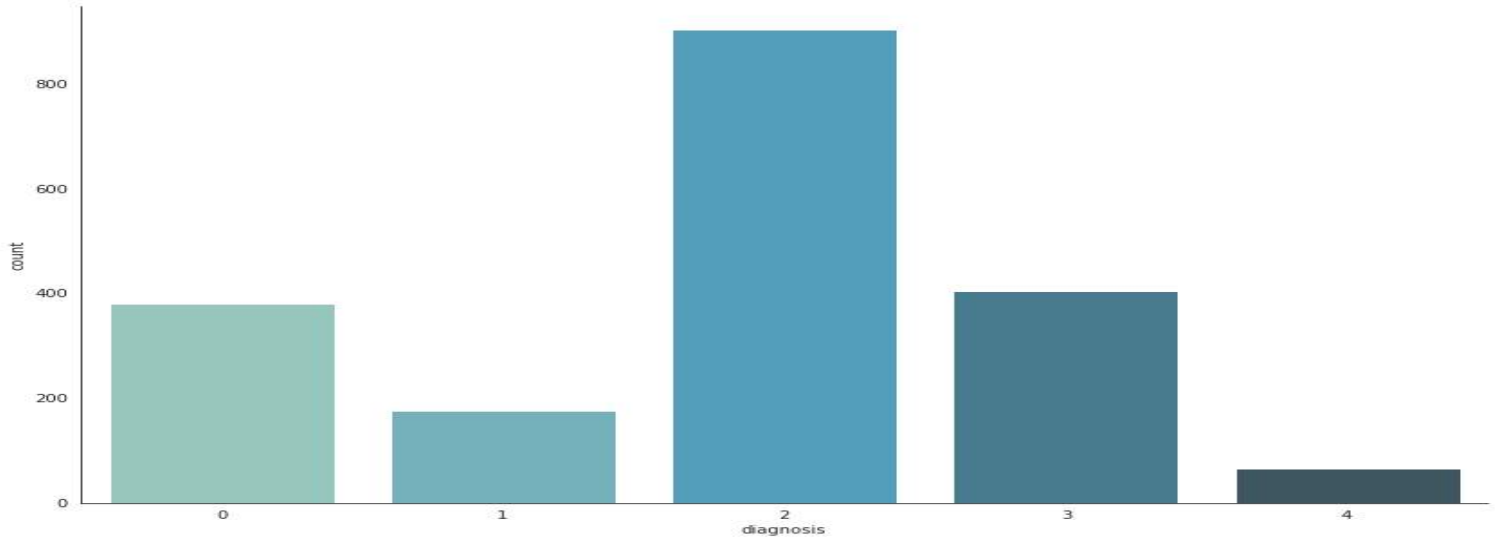
Apply model to test set and output predictions

Out[30]:

	id_code	diagnosis
0	0005cfc8afb6	2
1	003f0afdcd15	3
2	006efc72b638	3
3	00836aaacf06	2
4	009245722fa4	3
5	009c019a7309	2
6	010d915e229a	3
7	0111b949947e	1
8	01499815e469	3
9	0167076e7089	0

RESULT

Predictions class distribution



Conclusion and Future Work

- Training score : **95% (Validation data)**
- This Model gives Quadratic weighted kappa **0.85** on Test data – **85%** (Test data on kaggle.com Private score) : **85% (Test data)**
- Best activation and optimizing functions **(A-0.96,V-0.72)** : **Adam optimizer with softmax**
- Best optimal number of convolutional layers : **Layer 3(A- 0.74 ,L- 0.60)**
- Model with best dropout and normalization : **Dropout 30%(A -0.73 ,L- 0.69)**
- Best kernel size in model performance **: 5*5 (A – 0.73 ,L- 0.69)**

Conclusion and Future Work

Research paper used for the refrence

- Review on the Role of Macular Edema in Retinopathy, Blindness and Automated Diagnosis Methods

  (https://eudl.eu/doi/10.4108/eai.17-3-2021.169034)

- Performance Measures for the Proposed Method:

Accuracy	Specificity
80%	76.7%

Accuracy obtained in my project

- Training score : **95% (Validation data)**

- This Model gives Quadratic weighted kappa : **85% (Test data)**

  0.85 on Test data – **85%**

  (Test data on kaggle.com Private score)

Conclusion and Future Work

Future work:

Experimentation with Ensembling and K Fold Cross Validation

- The research paper also mentions usage of Ensembling across various architectures — **EfficientNetB4, 5EfficientNetB5, SE-ResNeXt50** etc and using **Stratified Cross validation (5 fold)** to improve model performance and generalization ability.
- Using different Optimizer and activation function ex; Adamax,Nadam etc. **(A-0.96,V-0.72)**
- Model dropout and batch normalization can help in accuracy.
- Performing different kernel size in model performance.
- Increasing no. of epochs to get better accuracy.

References

References

- Nwankpa, C. E., Ijomah, W., Gachagan, A., & Marshall, S. (2018). Activation Functions : Comparison of Trends in Practice and Research for Deep Learning. *ArXiv*, 1–20.
- Ruamviboonsuk, P., Krause, J., Chotcomwongse, P., Sayres, R., Raman, R., Widner, K., … Webster, D. R. (2019). Deep learning versus human graders for classifying diabetic retinopathy severity in a nationwide screening program. *Nature Digital Medicine*, *25*(March), 9.
- I would like to thank the authors found in the link below for their insight and wisdom. I learned SO much!
- Jason Brownlee has great tutorials on essentially everything! Check his work our here https://machinelearningmastery.com/