Detecting Diabetic Rtinopathy using Deep Learning

November 18, 2019

Presented By:

Subhajit Barh

Department of Mathematics

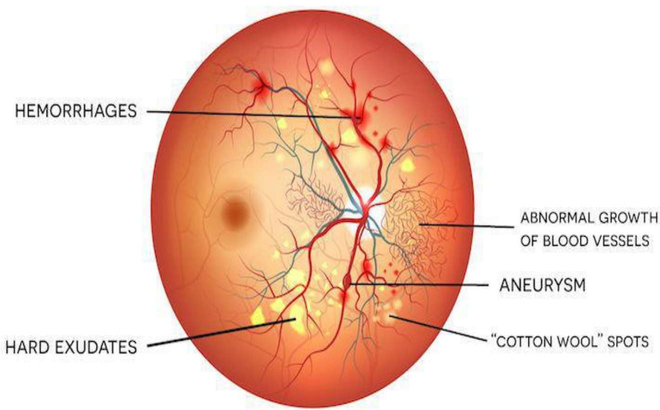
(Computer Science and Data Processing)

Indian Institute of Technology Kharagpur

Contents in Brief

What is Diabetic Retinopathy?

Diabetic retinopathy (DR), also known as diabetic eye disease, is a medical condition in which damage occurs to the retina due to diabetes mellitus. It is a leading cause of blindness. Diabetic retinopathy affects up to 80 percent of those who have had diabetes aged 20 years or more.



Types Of DR

Depending on severity we can devide Diabetic Retinopathy in 5 categories

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

Manual Detection

Detecting Daiabetic Retinopathy is currently done by a method called Dilated eye exam . In this method the doctor administers drops in patient's eye and take pictures of the retina. After reviewing the image doctor have to manually conclude about the presence or advance of the disease . The doctor or experts generally looks for abnormalities in the blood vessels, optic nerve, retina,or formation of new blood vessels,retinal detachment,scar tissue etc .

Manual Detection

Detecting Daiabetic Retinopathy is currently done by a method called Dilated eye exam . In this method the doctor administers drops in patient's eye and take pictures of the retina. After reviewing the image doctor have to manually conclude about the presence or advance of the disease . The doctor or experts generally looks for abnormalities in the blood vessels, optic nerve, retina,or formation of new blood vessels,retinal detachment,scar tissue etc .

Automated Detection

An automated tool for grading severity of diabetic retinopathy would be very useful for accerelating detection and treatment. We will use a Deep learning convolutional neural network model to detect Diabetic Retinopathy with minimum human intervention .

We will use Convolutional Neural Network along with transfer Learning to detect and categorize the presence of diabetic Retinopathy

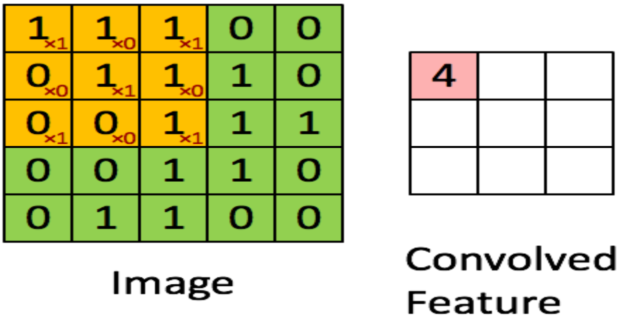
CNN

CNN(convolutional Neural Network) is an neural network architecture which works really well with Images. Each Convolution Layer Consists of mainly three sub layers.

1. Convolution layer
2. Pooling Layer
3. Fully Connected Layer

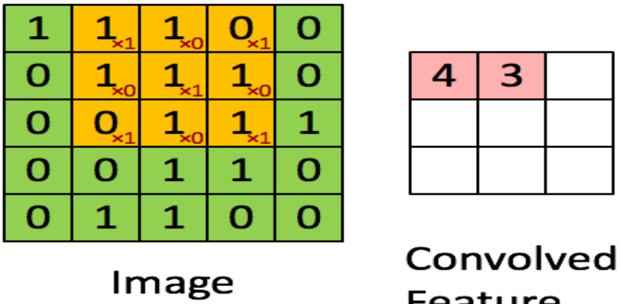
Convolution Layer

In convolution layer the filter(a smaller dimentional matrix) moves to the right with a certain Stride Value till it parses the complete width. Moving on, it hops down to the beginning (left) of the image with the same Stride Value and repeats the process until the entire image is traversed.In each step it computes the dot product with the original matrix and thus constructs the output matrix.



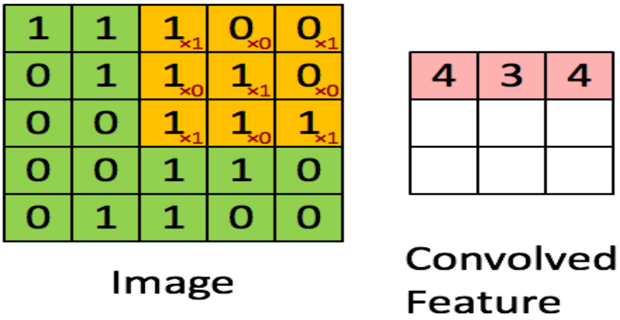
Convolution Layer

In convolution layer the filter(a smaller dimentional matrix) moves to the right with a certain Stride Value till it parses the complete width. Moving on, it hops down to the beginning (left) of the image with the same Stride Value and repeats the process until the entire image is traversed.In each step it computes the dot product with the original matrix and thus constructs the output matrix.



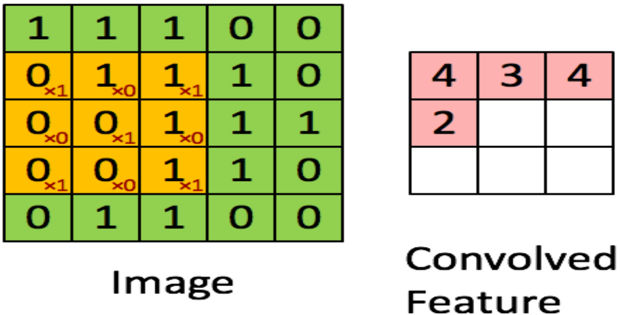
Convolution Layer

In convolution layer the filter(a smaller dimentional matrix) moves to the right with a certain Stride Value till it parses the complete width. Moving on, it hops down to the beginning (left) of the image with the same Stride Value and repeats the process until the entire image is traversed.In each step it computes the dot product with the original matrix and thus constructs the output matrix.



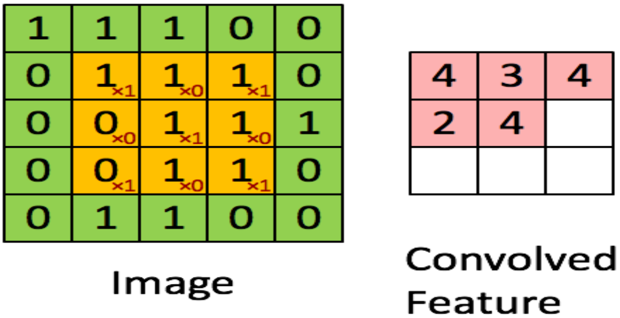
Convolution Layer

In convolution layer the filter(a smaller dimentional matrix) moves to the right with a certain Stride Value till it parses the complete width. Moving on, it hops down to the beginning (left) of the image with the same Stride Value and repeats the process until the entire image is traversed.In each step it computes the dot product with the original matrix and thus constructs the output matrix.



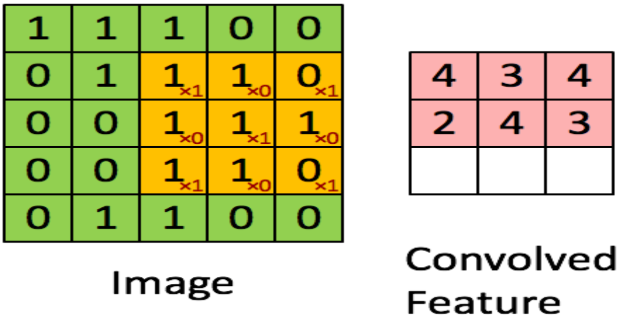
Convolution Layer

In convolution layer the filter(a smaller dimentional matrix) moves to the right with a certain Stride Value till it parses the complete width. Moving on, it hops down to the beginning (left) of the image with the same Stride Value and repeats the process until the entire image is traversed.In each step it computes the dot product with the original matrix and thus constructs the output matrix.



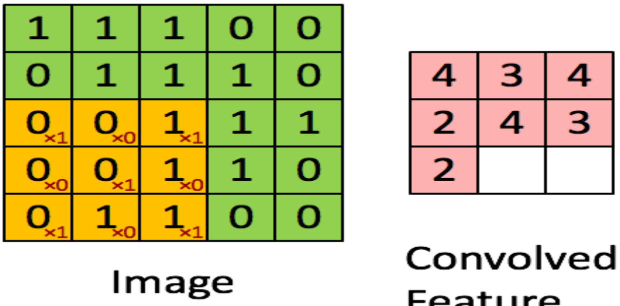
Convolution Layer

In convolution layer the filter(a smaller dimentional matrix) moves to the right with a certain Stride Value till it parses the complete width. Moving on, it hops down to the beginning (left) of the image with the same Stride Value and repeats the process until the entire image is traversed.In each step it computes the dot product with the original matrix and thus constructs the output matrix.



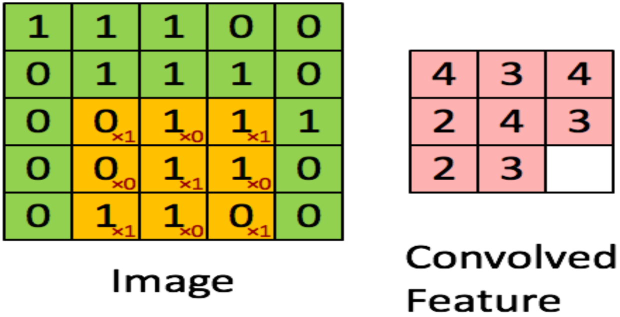
Convolution Layer

In convolution layer the filter(a smaller dimentional matrix) moves to the right with a certain Stride Value till it parses the complete width. Moving on, it hops down to the beginning (left) of the image with the same Stride Value and repeats the process until the entire image is traversed.In each step it computes the dot product with the original matrix and thus constructs the output matrix.



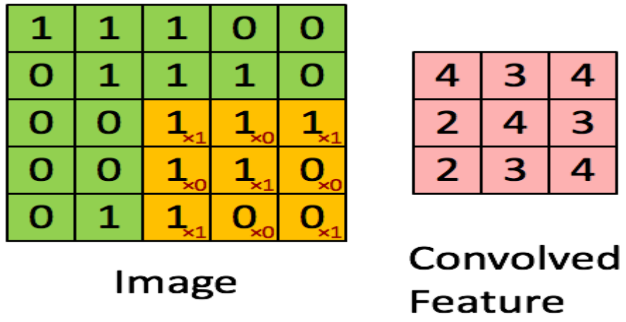
Convolution Layer

In convolution layer the filter(a smaller dimentional matrix) moves to the right with a certain Stride Value till it parses the complete width. Moving on, it hops down to the beginning (left) of the image with the same Stride Value and repeats the process until the entire image is traversed.In each step it computes the dot product with the original matrix and thus constructs the output matrix.



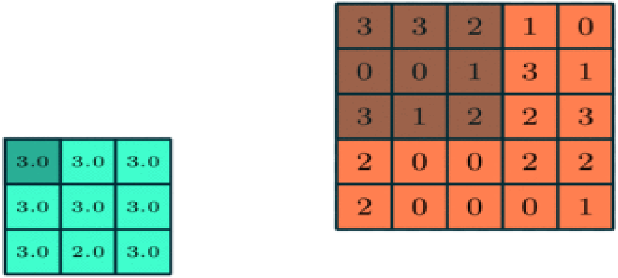
Convolution Layer

In convolution layer the filter(a smaller dimentional matrix) moves to the right with a certain Stride Value till it parses the complete width. Moving on, it hops down to the beginning (left) of the image with the same Stride Value and repeats the process until the entire image is traversed.In each step it computes the dot product with the original matrix and thus constructs the output matrix.



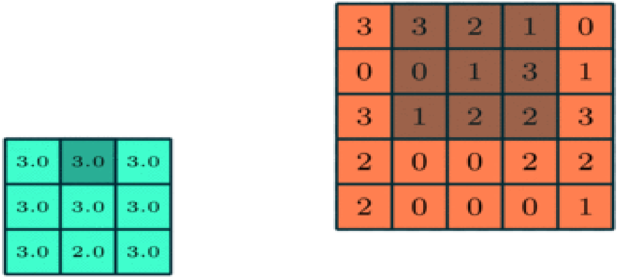
Similar to the Convolutional Layer, the Pooling layer is responsible for reducing the spatial size of the Convolved Feature

There are two types of Pooling: Max Pooling and Average Pooling. Max Pooling returns the maximum value from the portion of the image covered by the Kernel. On the other hand, Average Pooling returns the average of all the values from the portion of the image covered by the Kernel



Pooling Layer

Similar to the Convolutional Layer, the Pooling layer is responsible for reducing the spatial size of the Convolved Feature
There are two types of Pooling: Max Pooling and Average Pooling. Max Pooling returns the maximum value from the portion of the image covered by the Kernel. On the other hand, Average Pooling returns the average of all the values from the portion of the image covered by the Kernel



Pooling Layer

Similar to the Convolutional Layer, the Pooling layer is responsible for reducing the spatial size of the Convolved Feature

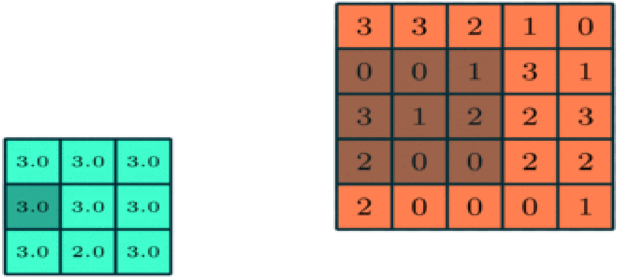
There are two types of Pooling: Max Pooling and Average Pooling. Max Pooling returns the maximum value from the portion of the image covered by the Kernel. On the other hand, Average Pooling returns the average of all the values from the portion of the image covered by the Kernel

3.0	3.0	3.0
3.0	3.0	3.0
3.0	2.0	3.0

3	3	2	1	0
0	0	1	3	1
3	1	2	2	3
2	0	0	2	2
2	0	0	0	1

Pooling Layer

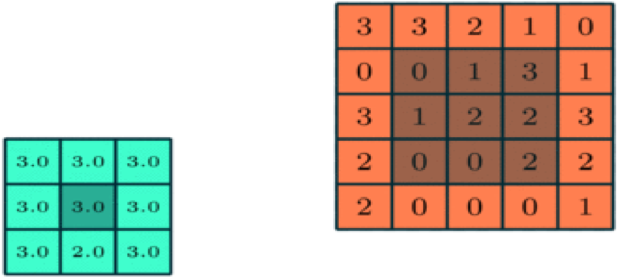
Similar to the Convolutional Layer, the Pooling layer is responsible for reducing the spatial size of the Convolved Feature
There are two types of Pooling: Max Pooling and Average Pooling. Max Pooling returns the maximum value from the portion of the image covered by the Kernel. On the other hand, Average Pooling returns the average of all the values from the portion of the image covered by the Kernel



Pooling Layer

Similar to the Convolutional Layer, the Pooling layer is responsible for reducing the spatial size of the Convolved Feature

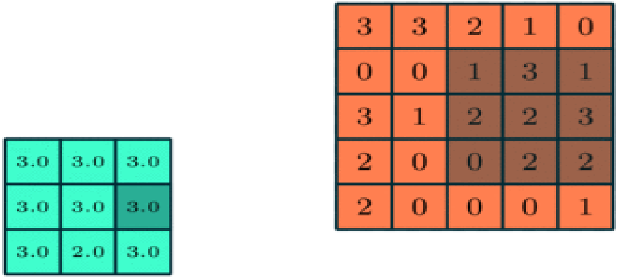
There are two types of Pooling: Max Pooling and Average Pooling. Max Pooling returns the maximum value from the portion of the image covered by the Kernel. On the other hand, Average Pooling returns the average of all the values from the portion of the image covered by the Kernel



Pooling Layer

Similar to the Convolutional Layer, the Pooling layer is responsible for reducing the spatial size of the Convolved Feature

There are two types of Pooling: Max Pooling and Average Pooling. Max Pooling returns the maximum value from the portion of the image covered by the Kernel. On the other hand, Average Pooling returns the average of all the values from the portion of the image covered by the Kernel



Similar to the Convolutional Layer, the Pooling layer is responsible for reducing the spatial size of the Convolved Feature

There are two types of Pooling: Max Pooling and Average Pooling. Max Pooling returns the maximum value from the portion of the image covered by the Kernel. On the other hand, Average Pooling returns the average of all the values from the portion of the image covered by the Kernel

3.0	3.0	3.0
3.0	3.0	3.0
3.0	2.0	3.0

3	3	2	1	0
0	0	1	3	1
3	1	2	2	3
2	0	0	2	2
2	0	0	0	1

Pooling Layer

Similar to the Convolutional Layer, the Pooling layer is responsible for reducing the spatial size of the Convolved Feature

There are two types of Pooling: Max Pooling and Average Pooling. Max Pooling returns the maximum value from the portion of the image covered by the Kernel. On the other hand, Average Pooling returns the average of all the values from the portion of the image covered by the Kernel

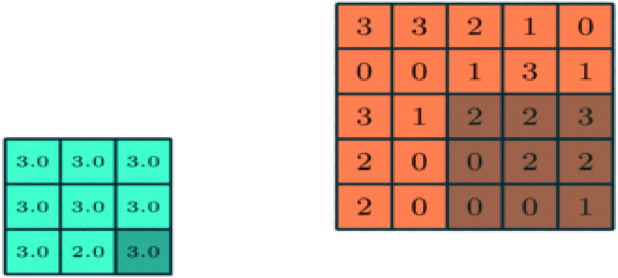
3.0	3.0	3.0
3.0	3.0	3.0
3.0	2.0	3.0

3	3	2	1	0
0	0	1	3	1
3	1	2	2	3
2	0	0	2	2
2	0	0	0	1

Pooling Layer

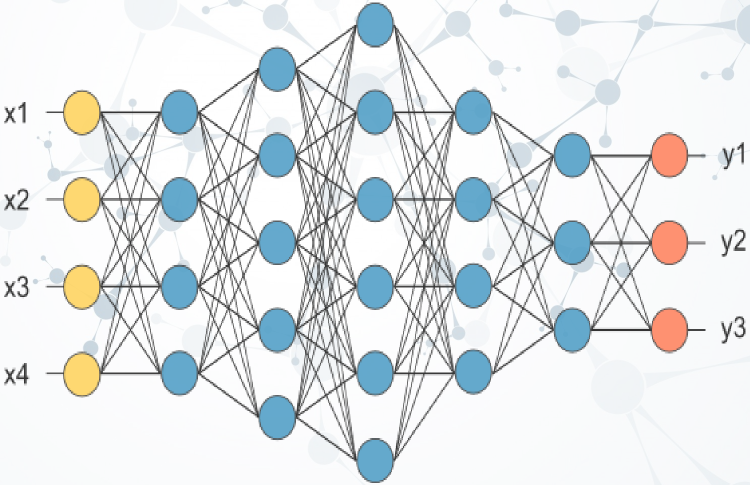
Similar to the Convolutional Layer, the Pooling layer is responsible for reducing the spatial size of the Convolved Feature

There are two types of Pooling: Max Pooling and Average Pooling. Max Pooling returns the maximum value from the portion of the image covered by the Kernel. On the other hand, Average Pooling returns the average of all the values from the portion of the image covered by the Kernel



Fully Connected Layer

The fully connected layer is nothing but a ordinary perception based network. Here we flattened our matrix into vector and feed it into a fully connected layer like any other neural network.



All those filters have weights. Those weights and percepton weights in FC layers must be learned using back-propagation . In each iteration we update those weights using our optimizer.Here we used Gradient Descent optimizer.But in future we can use optimizers like adam or adaguard etc.

Gradient Descent

Gradient descent is a iterative optimization algorithm for finding the minimum of a function. To find a local minimum of a function using gradient descent, one takes steps proportional to the negative of the gradient (or approximate gradient) of the function at the current point

Cost function or error function can be an indicatior of how accurate our model is and optimizer uses that to optimize our model. Here we will use MSE loss. Which is defined as

$$J(w,b) = \frac{1}{M}\sum(h_{(w,b)}(x^{(i)}) - y^{(i)})^2 \tag{1}$$

Then we calculate gradients of the cost function as below

$$\nabla_w J = \frac{\partial}{\partial w} J(w,b) \tag{2}$$

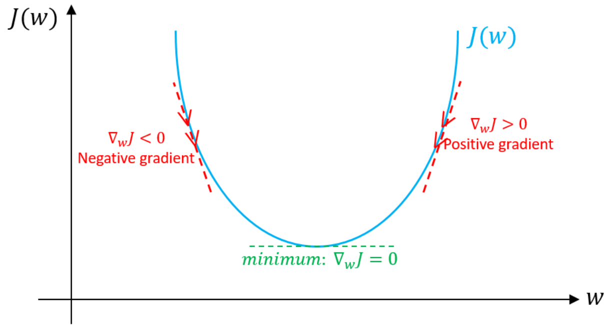
$$\nabla_b J = \frac{\partial}{\partial b} J(w,b) \tag{3}$$

Then we update the weights using the formula as below

$$w' = w - \alpha \nabla_w J \tag{4}$$

$$b' = b - \alpha \nabla_b J \tag{5}$$

Gradient Descent



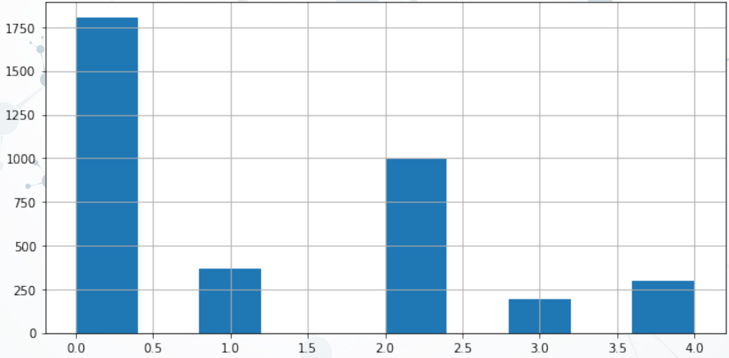
Transfer learning is a process in which we make use of the knowledge gained while solving one problem and applying it to a different but related problem .

Pretrained Models

Here we use a pre-trained big model named resnext101(44 million parameters) which was trained by facebook on Imagenet . We take the resnext101 model chop the last layer down and add our custom regression layer to classify the images.

Data

Asia Pacific Tele-Ophthalmology Society (APTOS) and Aravind Eye Hospital of India published a dataset of retina images through Kaggle.The Data-set can be found on `https://www.kaggle.com/c/aptos2019-blindness-detection/data` With quick exploration we can visualize the data as follows



First we train the model freezing the base layers. we do this for 30 epochs we used

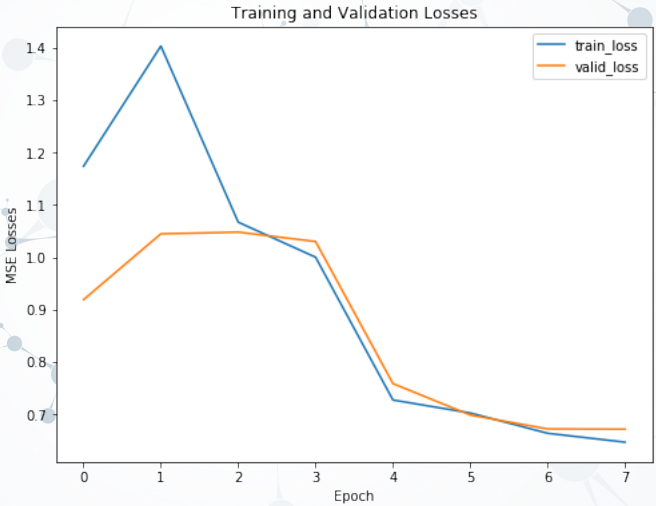
1. Gradient Descent as Optimizer
2. learning rate=0.001
3. cost function is Mean Square Error

Then we fine Tune the whole model for another 30 epochs .Here we unfreeze the base layer

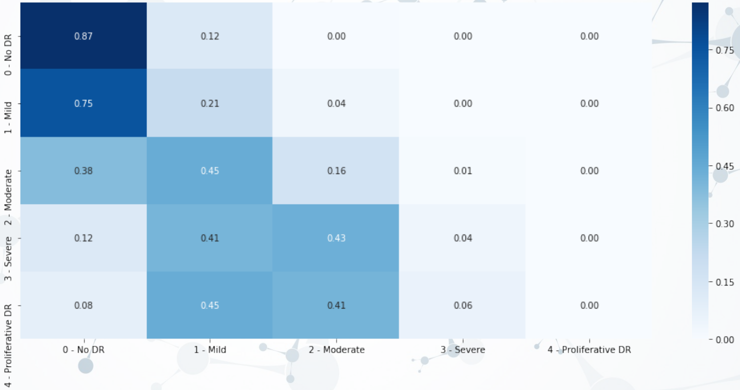
1. Gradient Descent as Optimizer
2. learning rate=0.0001
3. cost function is Mean Square Error

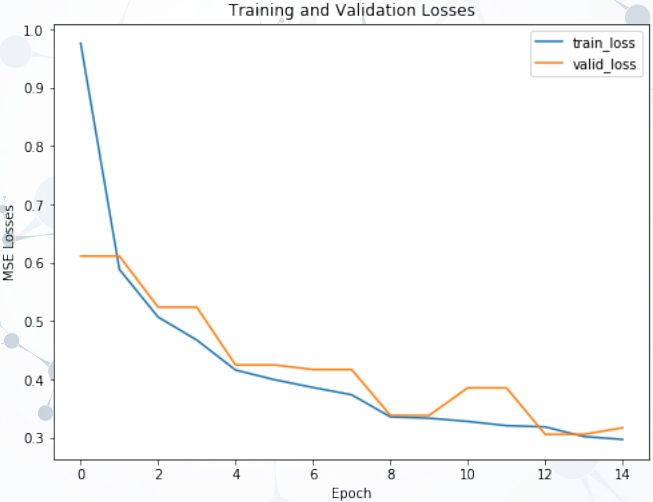
Detecting Diabetic Rtinopathy using Deep Learning

Introduction
Convolutional Neural Network
Transfer Learning
Data
Training and Finetuning
Result
Deficiency
Improvement
References

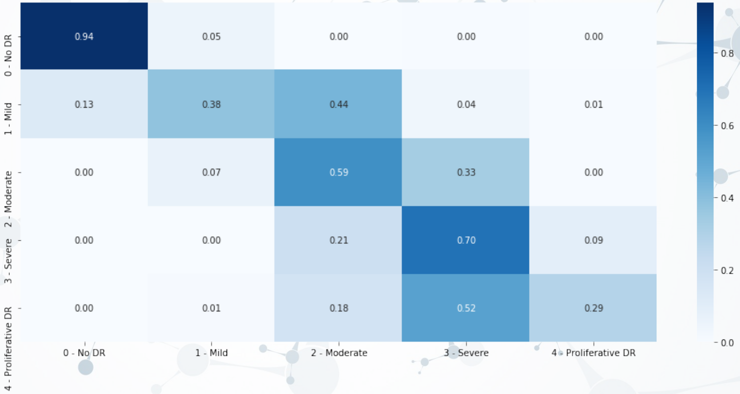


version 7(Confusion Matrix)

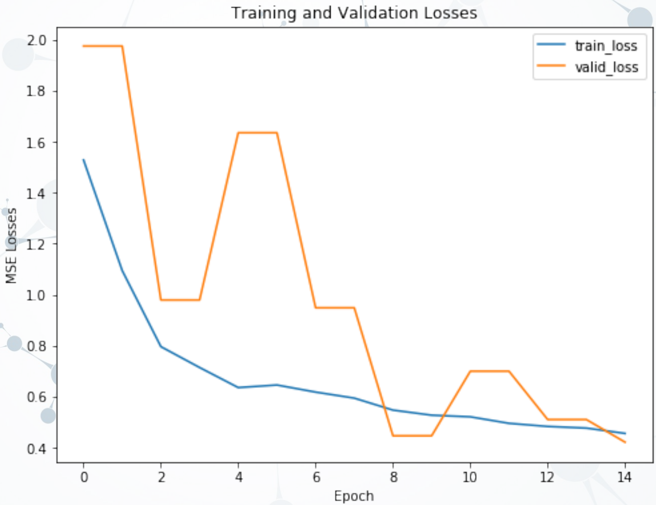



Training and Validation Losses
train_loss
valid_loss
MSE Losses
Epoch
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
2
4
6
8
10
12
14

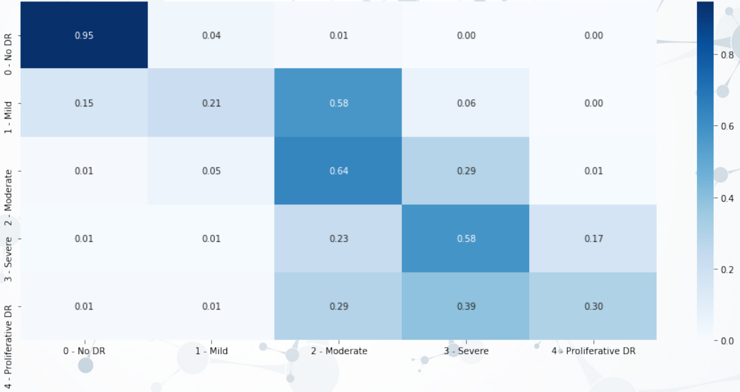
version 49(Confusion Matrix)



version 51(loss graph)



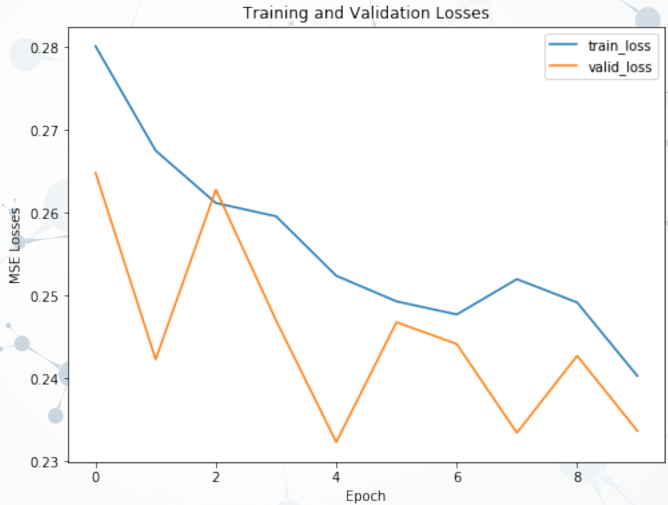
version 51(Confusion Matrix)



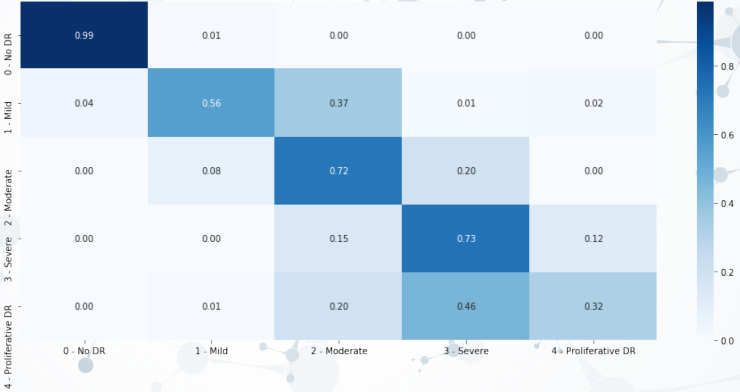
version 23(Fine tuned loss graph)


0.98
0.02
0.00
0.00
0.00
0.03
0.54
0.40
0.03
0.00
0.00
0.07
0.78
0.15
0.00
0.00
0.00
0.17
0.77
0.06
0.00
0.01
0.14
0.61
0.24
0 - No DR
1 - Mild
2 - Moderate
3 - Severe
4 - Proliferative DR
0.0
0.2
0.4
0.6
0.8

version 26(Fine tuned loss graph)



version 26(Fine tuned Confusion Matrix)



1) This model is very slow .Without GPU it takes more than 5 minutes to predict single image

Deficiency

1) This model is very slow .Without GPU it takes more than 5 minutes to predict single image
2) This model is very big .It is almost 2 GB in size.Which makes it impossible for Deployment

Deficiency

1) This model is very slow .Without GPU it takes more than 5 minutes to predict single image
2) This model is very big .It is almost 2 GB in size.Which makes it impossible for Deployment

1) for pre-trained model we can use something like efficient-net which is significantly smaller

Room for Future Improvements

1) for pre-trained model we can use something like efficient-net which is significantly smaller
2) We can use Heat-Map of model and overlay it on the top of existing picture to help human experts to conclude

Room for Future Improvements

1) for pre-trained model we can use something like efficient-net which is significantly smaller
2) We can use Heat-Map of model and overlay it on the top of existing picture to help human experts to conclude

References

1. *Preprocessing the Image* https://www.kaggle.com/ratthachat/aptos-eye-preprocessing-in-diabetic-retinopat
2. *Dataset* https://www.kaggle.com/c/aptos2019-blindness-detection
3. *Gradient Descent* https://towardsdatascience.com/understanding-the-mathematics-behind-gradient
4. *Transfer Learning* https://towardsdatascience.com/what-is-transfer-learning-8b1a0fa42b4
5. *ReNext* https://github.com/facebookresearch/ResNeXt