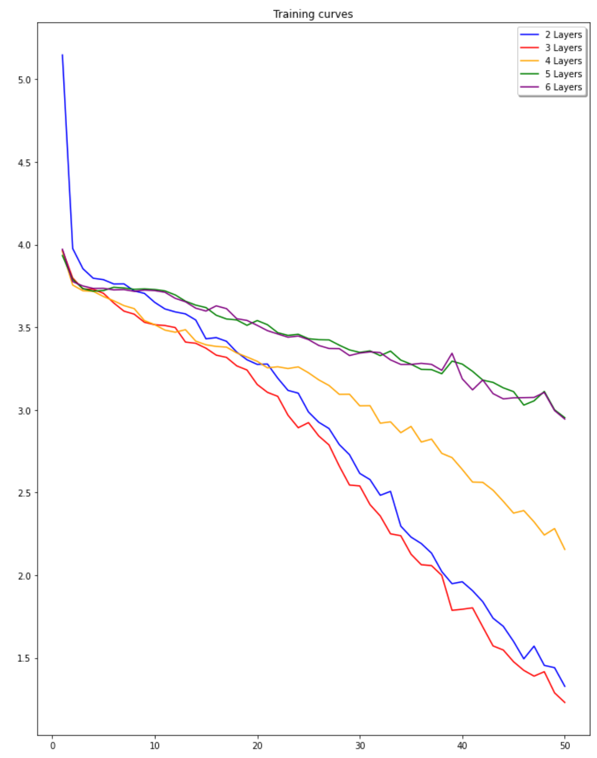
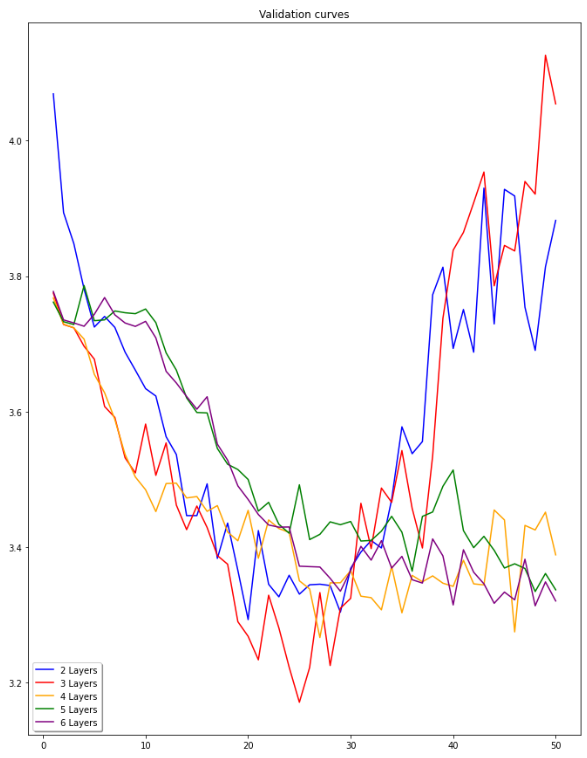
Report Assignment - 5

1) Effect of Number of Layers



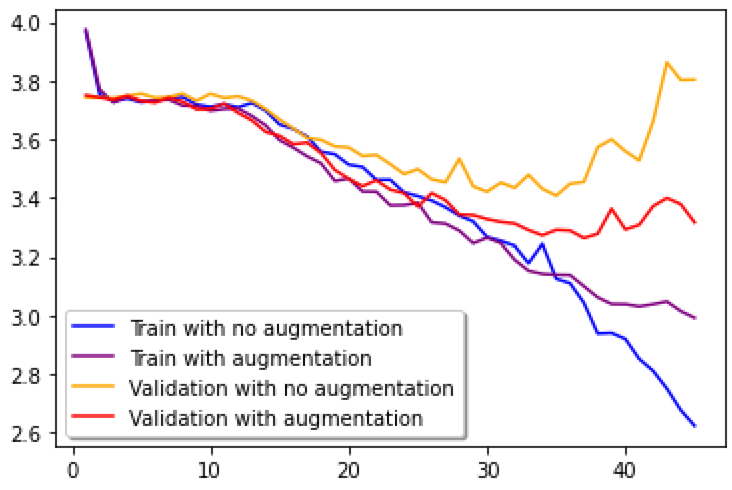


Analysis of the number of layers and reason for choosing a given configuration.

- The biggest observation is that more layers prevents overfitting.
- Although networks with lower numbers of layers achieve lower training losses, their validation losses increase drastically after ~35th epoch.
- The networks with 5 and 6 layers perform the best on unseen data.
- This shows that more layers allows the network to learn richer and more general features and thus allows it to generalise over unseen data.

Thus, we will use the network with 6 layers going forward.

2) Effect of Data Augmentation



Analysis of data augmentation.

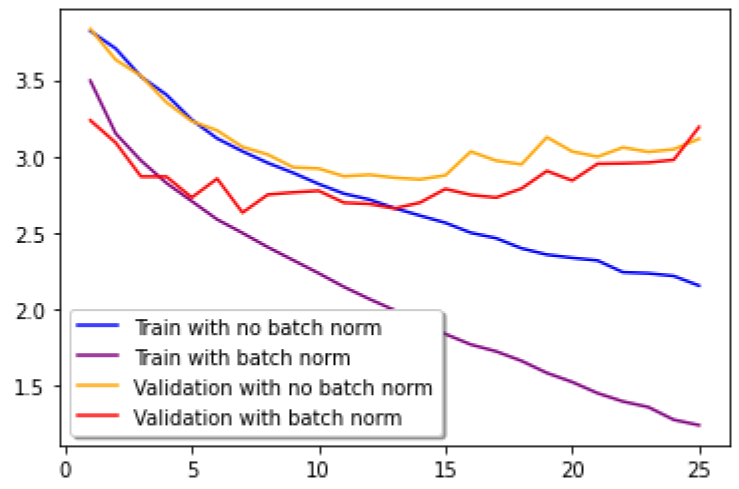
Data Augmentation used-

1. Normalisation
2. Horizontal and vertical flips.
3. Random and Central crop to 180x180.
4. Color jittering
5. Random Rotations.

- The most noticable difference of using data augmentation is that it prevents overfitting.
- If we observe the curves for no augmentation, we can observe that the training curve keep decreasing but the validation curve starting increasing after a certain point. This indicated overfitting.
- This does not happen in the validation curves.
- This is because we are essentially increasing the training data size effectly by a factor of 5. Thus, the network has more data points to learn from.

Thus, we will use data augmentation going forward.

3) Effect of Batch Normalisation

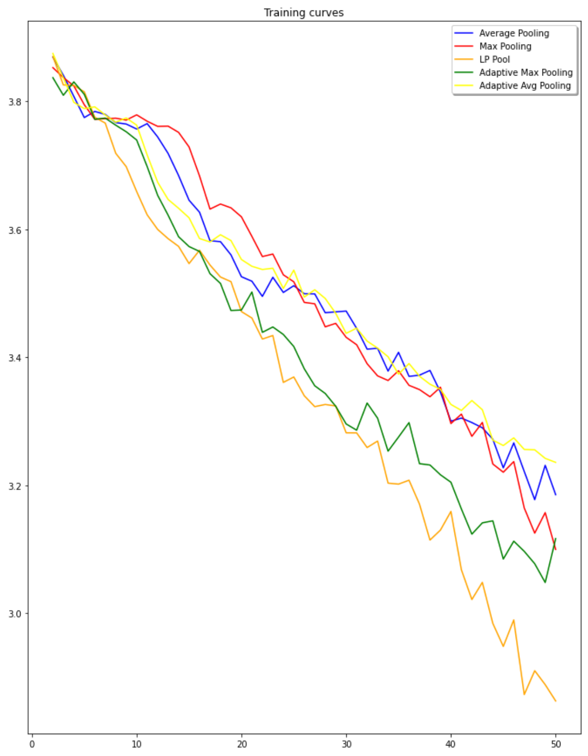


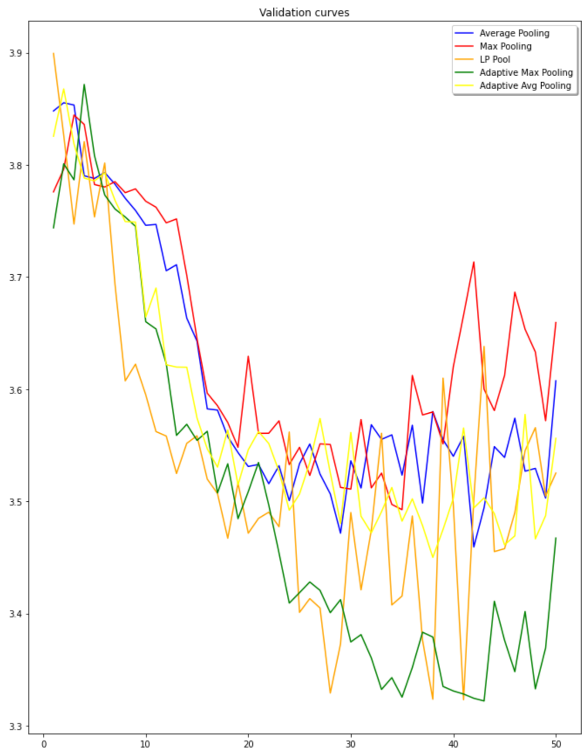
Batch normalization normalizes the output of a previous layer by subtracting the batch mean and dividing by the batch standard deviation.

- We can see that batch normalisation prodoces better results in both training and validation. But why? This is because it combats the issue of the well known internal covariance shift. To put it briefly, when a function learns a mapping from x to y, it should be invariant to any shift in distribution of x. Otherwise, it will not be able to generalise.

Thus, I will be using batch normalisation.

4) Comparing Different Pooling strategies

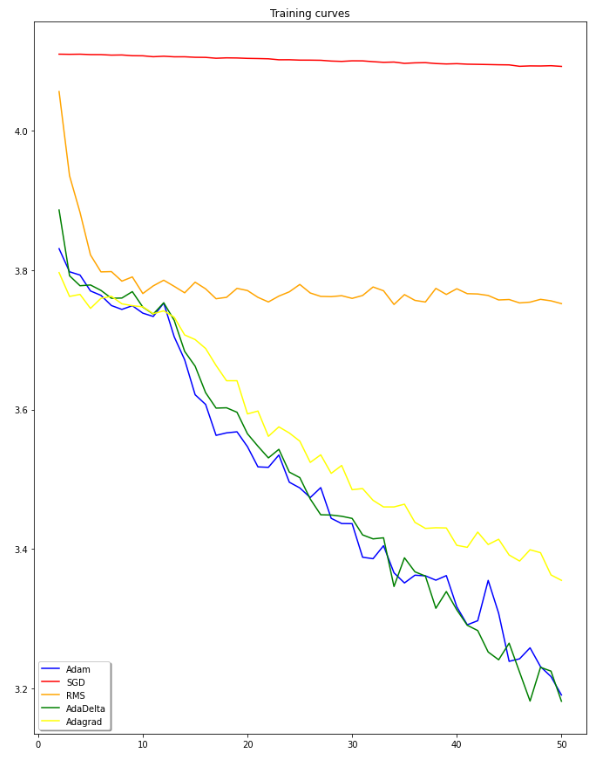


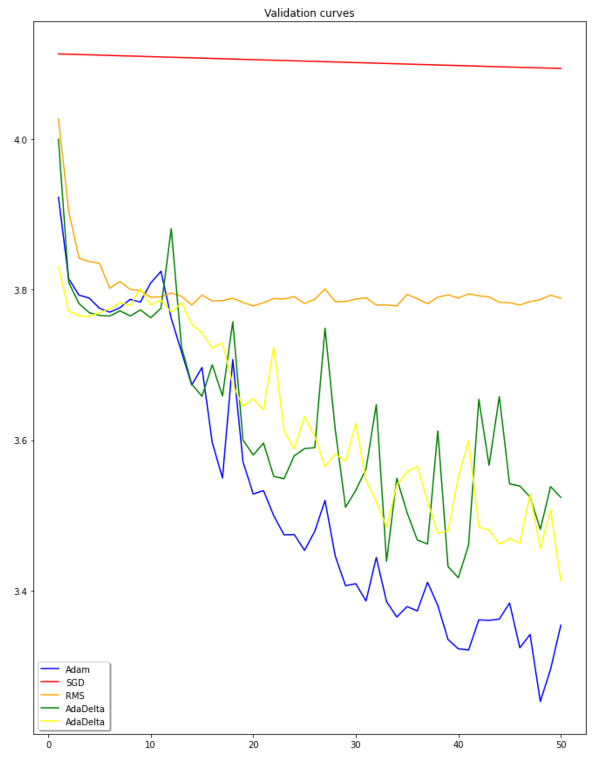


- We can see that **adaptive max pooling** performed the best for validation, this is because Adaptive in nature as the mixing proportion can adapt according to the features present within the pooling region. A similar explanation can be given for **adaptive avg pooling**
- **Max pooling** surprisingly does not perform well. 1 possible reason could be The discerning features disappear when the majority of the elements in the pooling area are present with high magnitudes. It is deterministic in nature.
- **Average pooling** may not be the best option as seen in the graphs because of reduced contrast if low magnitudes are taken into consideration.

Thus, for my model, I will use adaptive max pooling.

5) Effect of different optimizers



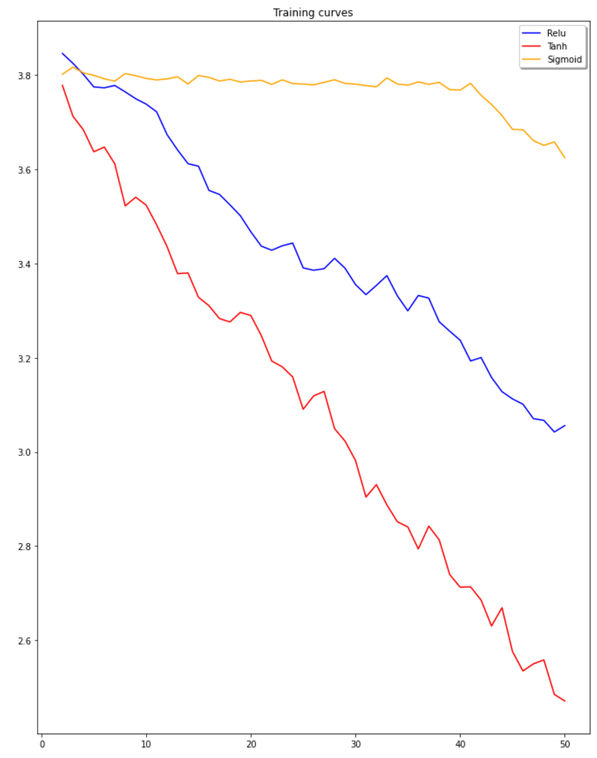


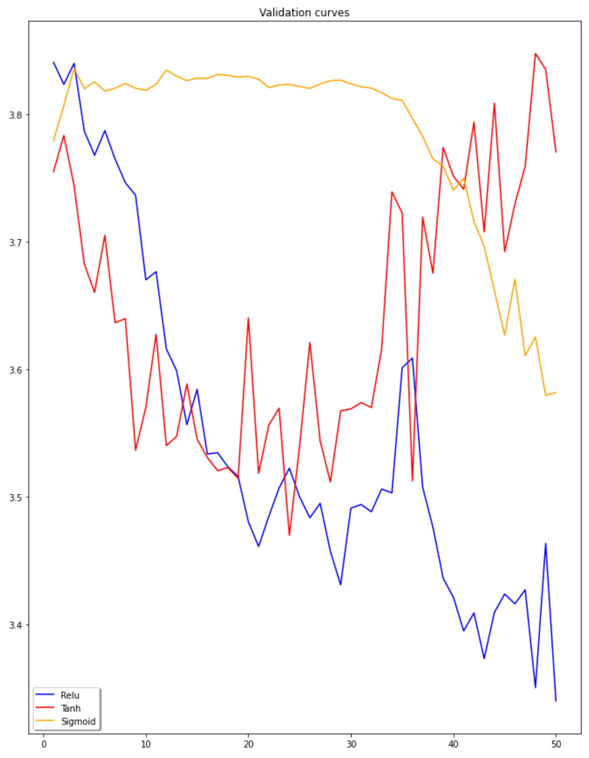
For all the optimizers, I have used the default learning rates.

1. SGD- This optimizer uses a fixed learning rate which does not change and updates its learning rate based on the features. This is why it decreases very slowly for both training and validation.
2. Adagrad- It adapts the learning rate to the parameters, performing smaller updates (i.e. low learning rates) for parameters associated with frequently occurring features, and larger updates (i.e. high learning rates) for parameters associated with infrequent features. Its main weakness is the accumulation of gradients.
3. Adadelta- It is an extension of Adagrad that seeks to reduce its aggressive, monotonically decreasing learning rate. Instead of accumulating all past squared gradients, Adadelta restricts the window of accumulated past gradients to some fixed window size.
4. Adam- Adam combines both Adagrad and RMS to better improve results.

All the improvements from one algo to the next are clearly visible in both the above graphs. Thus, I will use ADAM for my model.

6) Effect of Different Activation Functions





We can observe sigmoid performs the worst whereas ReLu performs the best.

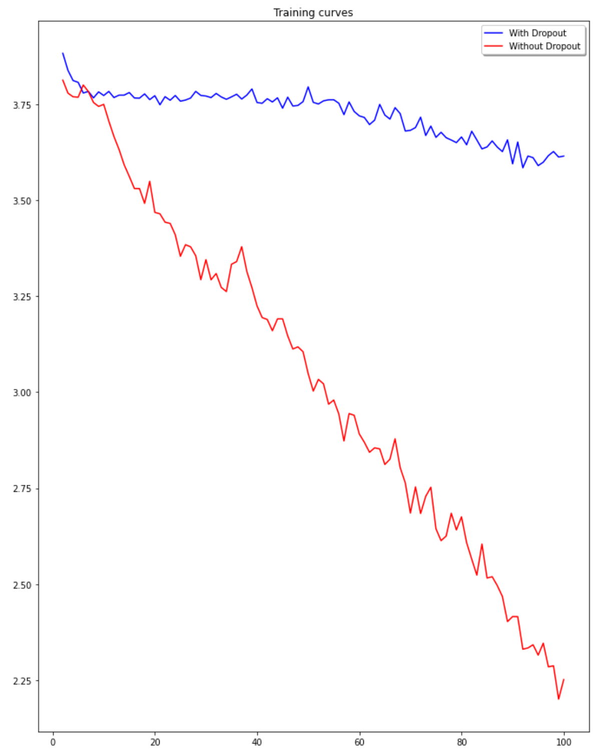
Why does Tanh perform better than sigmoid?

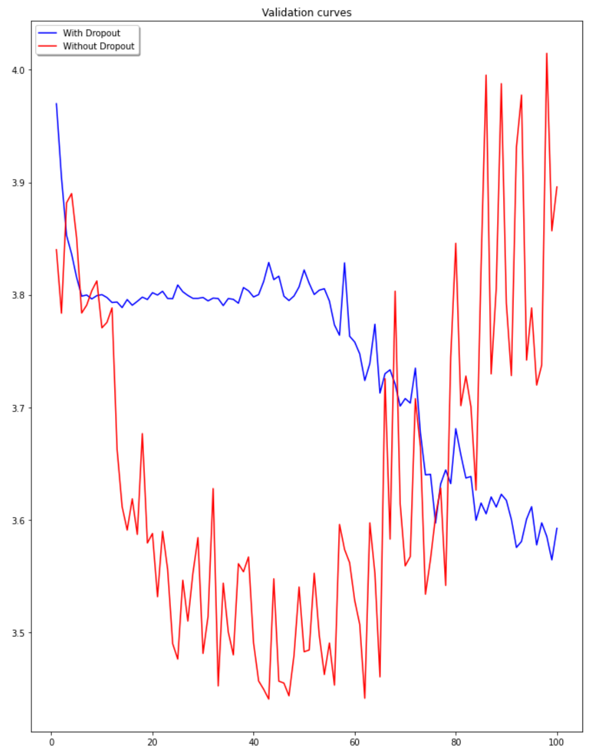
- The mean of Tanh would be close to 0 when compared to sigmoid due to the nature of the derivative. Since my input is normalised around 0, there are also likely to be centered about the origin. Thus, this may be the reason that Tanh outperforms sigmoid.

Why does Relu out perform Tanh and sigmoid?

- We can observe that relu converges faster as well. This is because ReLu is non-saturation of its gradient, which greatly accelerates the convergence of stochastic gradient descent compared to the sigmoid / tanh functions.

7) Effect of Dropout





We can see that drop out avoids overfitting as expected. This is because Dropout prevents overfitting due to a layer's "over-reliance" on a few of its inputs. Because these inputs aren't always present during training (i.e. they are dropped at random), the layer learns to use all of its inputs, improving generalization.

The effect is especially observable after the 75th epoch.

Thus, we will go ahead and use dropout in our training.