1. Motivation For Using CNNs

CNNs work particularly well when working with data with spatial relationships. The convolutional layers in CNNs allow for dimensionality reduction while preventing information loss. The filters in the convolutional layers break down the large image into smaller sub-images. Without manually performing feature extraction, CNNs allow for faster and more accurate training and predictions for image data.

2. Baseline

For the baseline model, we follow a very basic architecture as shown below.

We used an Adam optimizer and binary cross entropy loss function. An early stopping function was also included to prevent overfitting.

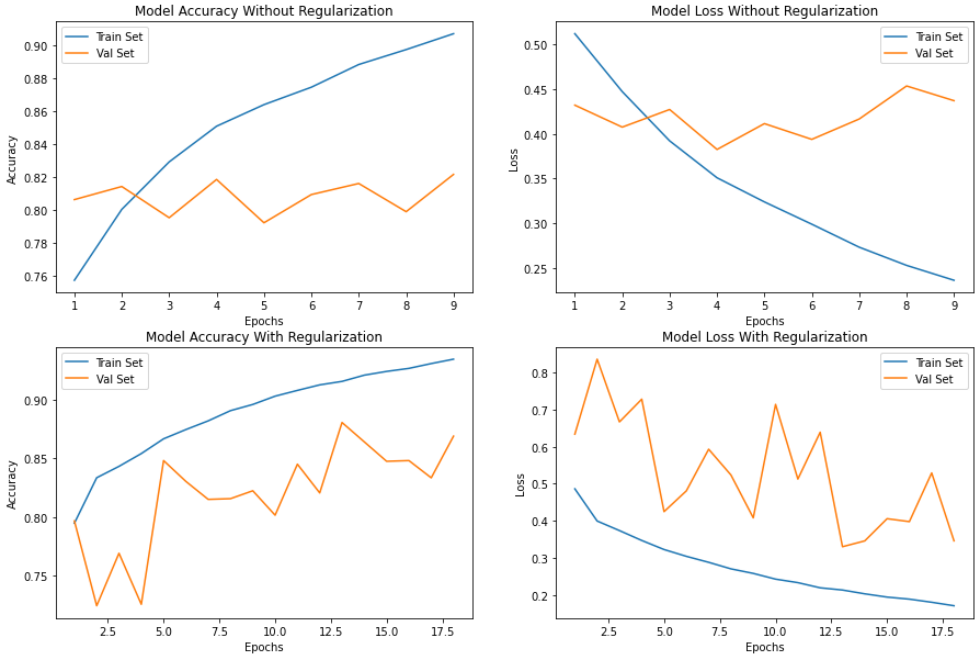
3. Expanding the Neural Network

With our very basic neural network architecture, there are already over 75 million trainable parameters. To reduce the training time, we add in (2,2) maxpool layers after every convolutional layer. As a result, we are able to experiment with more complex models. We tried a 3-layered (convolutional layers) model and a 6-layered model. The number of trainable parameters were further reduced by adding in dropout layers. Aside from that, everything remained the same. As mentioned by Tan et al. (*EfficientNets*), the model performance will increase for images with higher resolutions if a more complex model is used for feature extraction and prediction. As per expectations, the 6-layered neural network was able to capture much more information as compared to the 3-layered model.

4. Regularization

2 methods are commonly used for regularization, namely, L2-norm and Batch Normalization. They should only be used separately since batch normalization will cancel out the effect of the L2-norm if used one after another. Another consideration is to apply the activation function before or after batch normalization. Research has shown that batch normalization before activation help to reduce internal covariate shift. (Ioffe et al., 2015). However, there is still some debate as to whether is better. Hence, we will consider both and compare their performances.

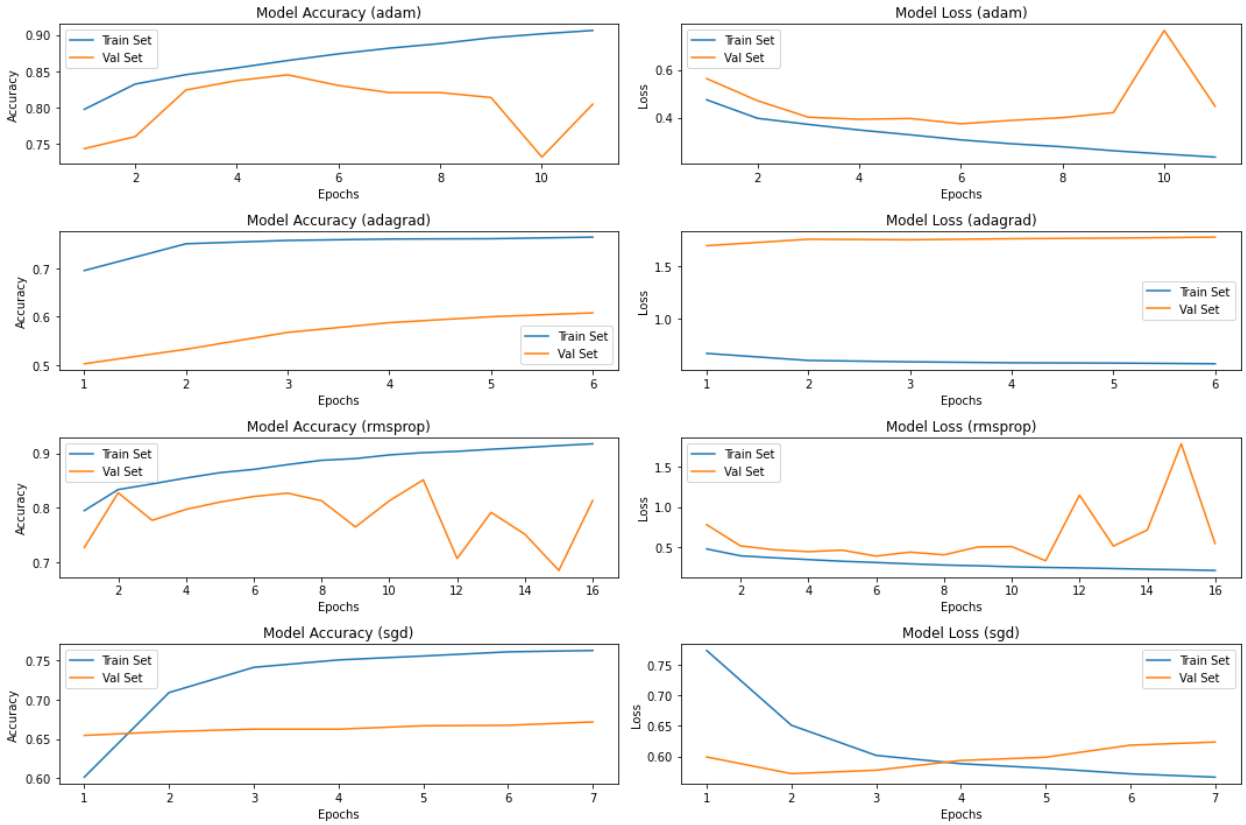
After experimenting with the above setups, we will proceed with applying batch norm before applying the activation function. As shown in the figure below, we can see that regularization is helping to prevent overfitting.



5. Tuning Optimizers

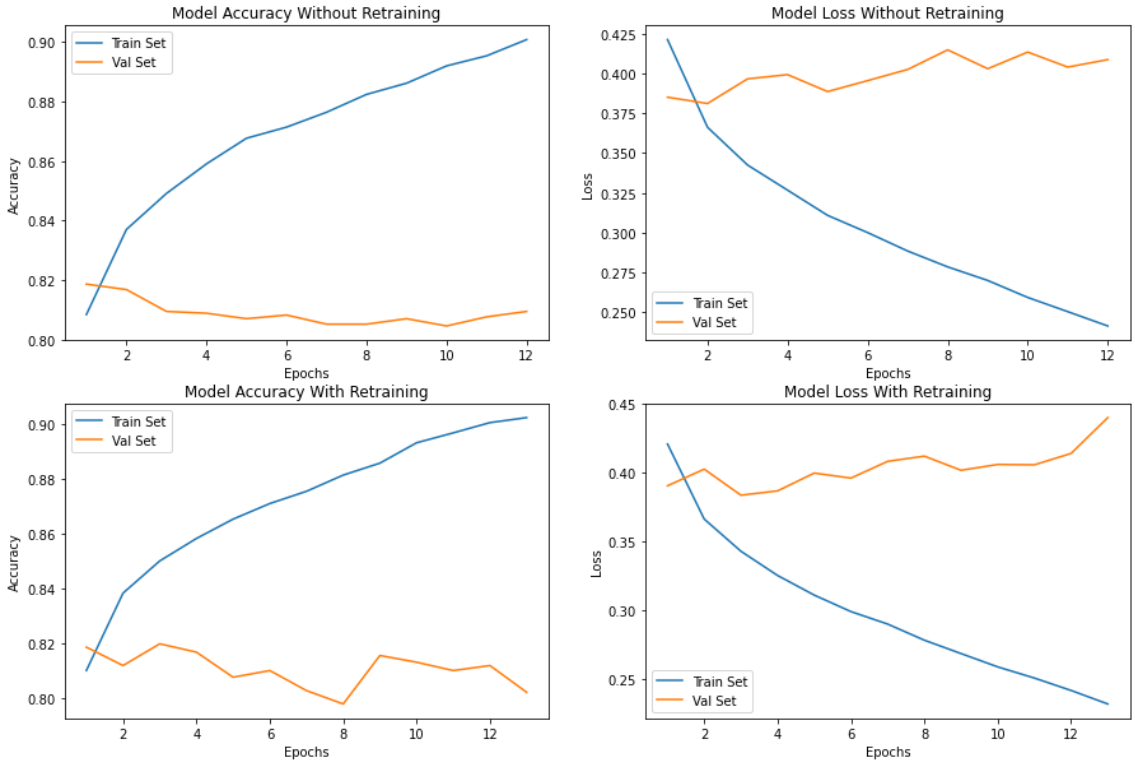
We will run the best 6-Layered model with batch normalization on 4 optimizers. Adam, Adagrad, RMSprop and SGD. Numerous research suggests that Adam is the best performing optimizers in terms of

speed and performance. Our results below shows that Adam and RMSprop are best suited for our model. However, since Adam is able to achieve similar results in a shorter period of time, we will proceed with Adam as our preferred optimizer.



6. Transfer Learning

There are many pre-trained models available. We settled on VGG-16 as it remains as one of the best pre-trained models since 2014. One disadvantage of using VGG-16 was the number of trainable parameters. As such, we experimenting with training only the fully connected layer as well as retraining the entire model including the VGG-16 model. Despite taking much longer per epoch, with or without retraining resulted in comparable performances between the models. The plot below shows the learning curve of both models.



7. Data Augmentation

Data augmentation allows the resultant trained model to be more robust. It is applicable in the field of healthcare as it reduces the cost required to collect the data and diminishes the effect of data scarcity. Data augmentation increases some level of variability in our current data, without the need of collecting new data. From our results, we observe that data augmentation produces comparable performance to without. However, training accuracy is not as high as without augmenting the data. This shows a lesser extent of overfitting.

8. Best Model

With all the above taken into consideration, we decided that the 6-layered CNN with batch normalization before activation, Adam optimizer and data augmentation will be our preferred model going forward. The diagram below shows our complete model.