Convolutional Neural Network – CIFAR-10

The report describes the neural network architecture created for CIFAR-10

Various permutation and combination of models are tried and tested by changing various parameters of the network to understand the effect of parameters and hyperparameters on the performance of the network.

The best models are selected, and the details are described in individual reports attached.

1. Hyperparameter tuning of Optimizers:

Optimizers are used in back propagation to reduce the loss function to global minima. They update the parameters of the neural network and effectively improve the training of the network.

Various optimizers were used to train the CIFAR-10 neural network model by keeping the other parameters such as architecture(layers), epochs, batch, regularization constant to study the effect of each optimizer and select the best optimizer for our network.

*Layer1 – 32 ,32, maxpooling (2*2)*

*Layer 2- 64,64, maxpolling (2*2)*

Flatten layer – 512, dense-10

Drop rate – 0.25

Batch size – 32

Epochs – 40

The below is the detailed analysis report having training loss, testing loss, training accuracy and testing accuracy of each model with different optimizers.

The optimizer outperformed all other optimizer models giving the testing accuracy of

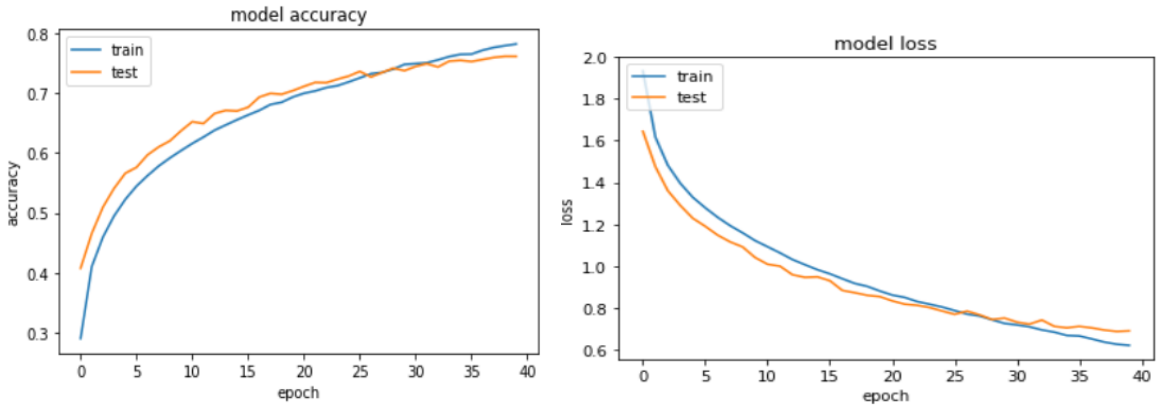
This model is the best model according to me because there is no overfitting /underfitting of the data and the testing loss is less as compared to other models as it can be seen from the below graphs.

CNN-Optimizer Tuning.xlsx

The model which performs best is the model with Adam optimizer:

Training Accuracy - 0.7827

Testing Accuracy - 0.7667



2. Hyperparameter Tuning of Network Architecture

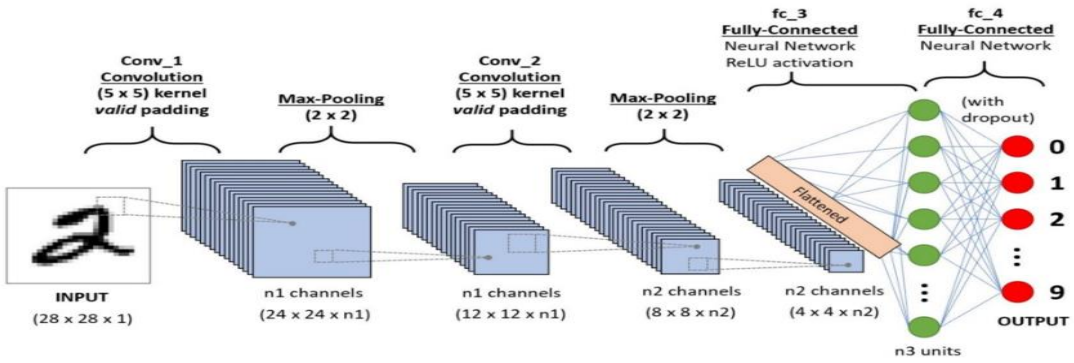
Neural network consists of many layers known as hidden layers which are responsible for learning the data which forms the architecture of the model. Deciding the accurate hidden layers plays a very key role in training the network which avoids it from overfitting and underfitting problems.

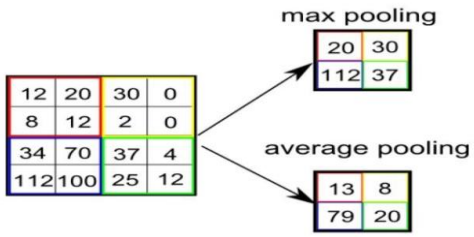
Convolutional layer: Image is convolved around with the given filter size to reduce the image dimensions if the padding = 'valid' .

Maxpooling layer: the input image is convolved around with the mentioned filters and the maximum feature is selected from each part of the image which further reduces the dimensions

Flatten layer: acts as a fully connected layer

Dense layer: output layer (in our case -10)





Various models are created having different hidden layers structure to relate the effect of overfitting, underfitting and accuracy difference between each other.

Below parameters are kept constant to determine the model (network) with good testing accuracy

Activation - Adam

Batch Size - 128

Epochs – 50

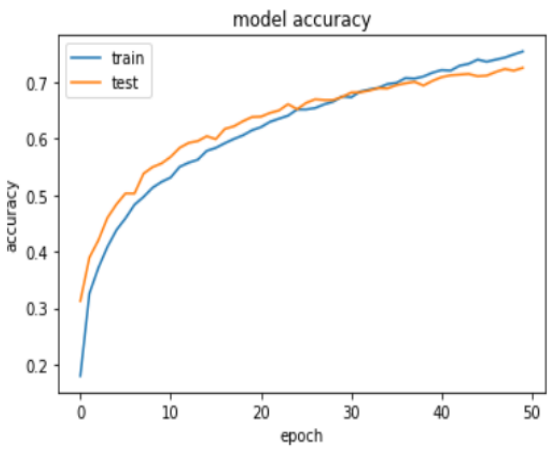
The below sheet details the models with varying networks along with their loss and accuracy.

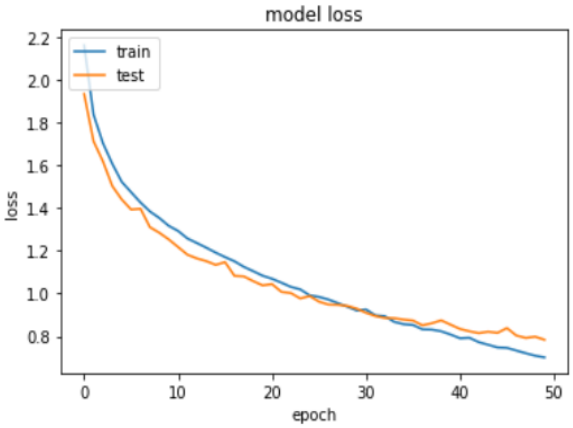
CNN-Network Architecture Tuning.xl

The model seems to be the best model with:

Layer 1 Relu - 256
Layer 2 Relu- 128
Layer 3 Relu-64
Flatten - 1024
Flatten – 512

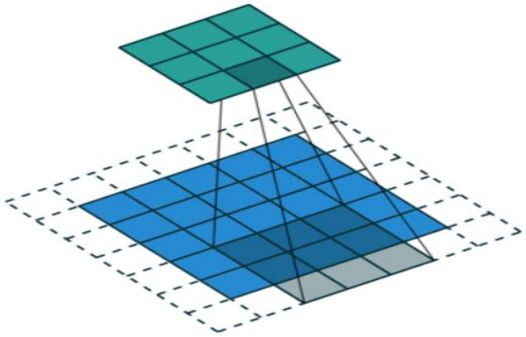
Training Accuracy - 0.755
Testing Accuracy - 0.726
Training Loss - 0.7012
Testing Loss - 0.7837
The other models seem to overfit leading to high testing accuracy then the training accuracy, therefore I chose this model because it does not overfit or underfit.





3. Hyperparameter Tuning of Strides and padding of the layers

The entire image is convolved with the filters.



Strides determine the amount of shift of each filter after convolving on a part of the image

Padding ensures that the dimensions of the image are proper for convolving the filter around the image.

Padding = 'Same' – Dimensions of the image are maintained after each layer

Padding = 'Valid' – no padding on the image

Padding and strides are changed for various models to get the best model testing accuracy

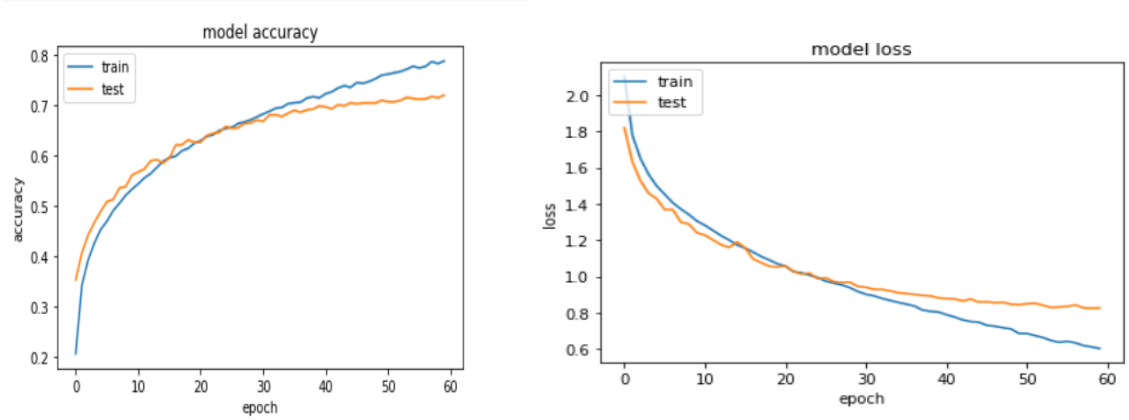
The details of each model are attached in the sheet are attached:

CNN-conv2D Tuning.xlsx

Strides with 4 and 3 with no padding failed because of the inconsistency in the dimensions.

The best model details:

Relu - 32
Relu-32
Relu-64
Relu-64
Flatten – 512
With strides 1 and no padding works as the best model with testing accuracy of 0.7195 and testing loss 0.8255
The model tries to learn without overfitting / underfitting.
Accuracy can be improved by increasing the epochs.



4. Batch Normalization

Covariant shift - causes the shift in the distribution of data while training the network. It causes the layers to learn again and again with the different distributions which leads to the slow learning of the model.

Batch Normalization is the solution to the problem which is applied after each layer before the activation layer of each neuron. It causes the mean to 0 and std deviation to 1 and normalizes the data which ensures that the data distribution lies on similar line.

The effect of batch normalization with respect to no normalization can be seen below:

CNN-Batch Normalization.xlsx

The testing accuracy drastically improves when the batch normalization is applied on the layers.