ASSIGNMENT NO. 4

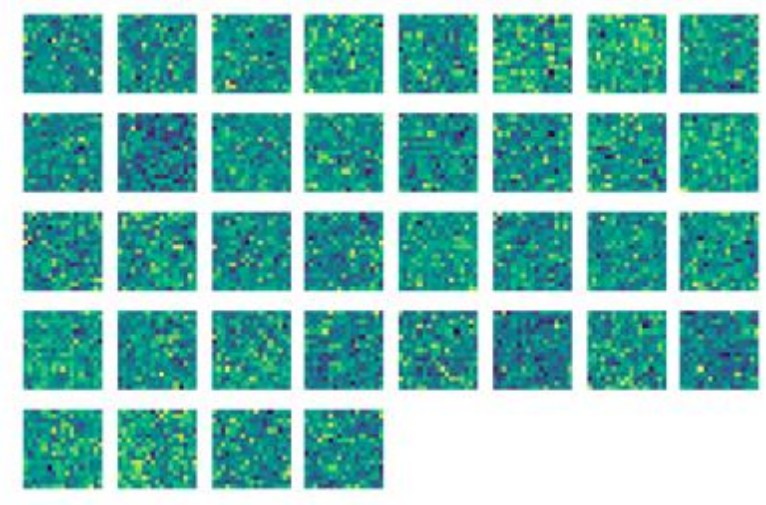
Deep Learning

Submitted by,
Muhammad Mudasser Afzal

Assignment 4 Report

Convolutional neural networks are used for feature extraction from the set of image. CNN is very similar to the ordinary neural network as we made in assignment 2.

In this task firstly we read dataset which is cifar-10. It has three channels. We make our filter using some random numbers with the 36 filters in number which are 17x17 dimensional and will be used to detect 36 number of changes in an image like; gradient filter along x-axis or y-axis. Here is the representation of filters that we have used.



In CNN firstly we have to implement convolution function in which I, implement a single step of convolution, in which you apply the filter to a single position of the input. This will be used to build a convolutional unit, which Takes an input volume, Applies a filter at every position of the input and Outputs another volume (usually of different size). As cifar data is three dimensional so we have to apply the filter to every channel and at the end add them up to make a 2D image

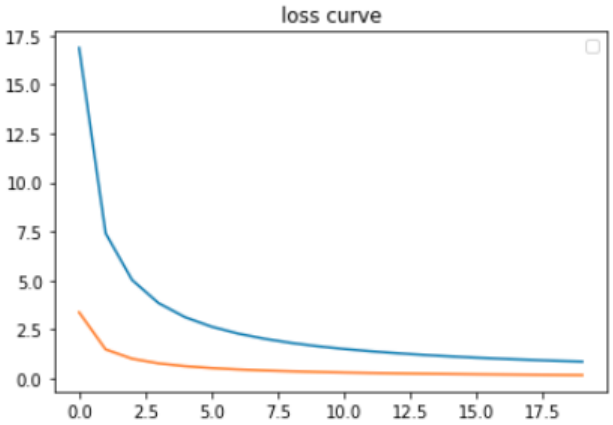
After performing convolution, we normally do pooling operation which can max or average pooling with the main purpose to reduce the dimension of data which help in using less computation power. In both of these steps, we have to first calculate the dimension of the output value and then we execute out width and height loop with the condition on output dimension. This methodology helps us to carry out a controlled operation.

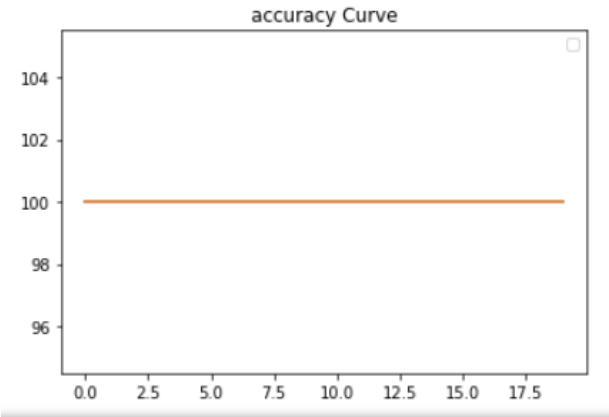
After pooling our task is to do backpropagation on filters to update them because we don't want to add all weights on FCN. We then implement back_convolution and back_pooling function which get back the original image and update the filter vectors.

The number of neurons used on hidden and output layers are 128,64 and 10. The results we get are somehow like this.

Results on 20 epochs and 0.1 learning rate for training.

Train accuracy = 100% and Testing Accuracy = 100%





We did the same for MNIST dataset as well and The number of neurons used on hidden and output layers are 128,64 and 10. The results we get are somehow like this.

Results on 20 epochs and 0.1 learning rate.

Train accuracy = 16% and Testing Accuracy = 8%