How Convolution Neural Network works?

Convolution Neural Network

A CNN is also known as a Convolution Neural Network(CNN), consisting of several different network architectures such as convolution, pooling, normalization, and Perceptron. Each one has different characteristics.

Convolution Layer - As the name suggests, this one is the building block of CNN. It consists of several filters of various sizes, and by using a layer, you will find several features from images. For example, Sobel filters help find horizontal and vertical lines from images.
Pooling Layer- It is a part of a concept known as dimensionality reduction. It will reduce the size of the feature map we obtained using the Convolution Layer.

Normalization Layer - It is helpful to speed up and stabilize the learning process.

Perceptron Network – It is a fully connected neural network that will simplify machine learning tasks as we reach the end of the training period.

Here, We have used a 3-layered Convoluted Network followed by a 3-layered fully connected network.

We are attaching reference images for reference.

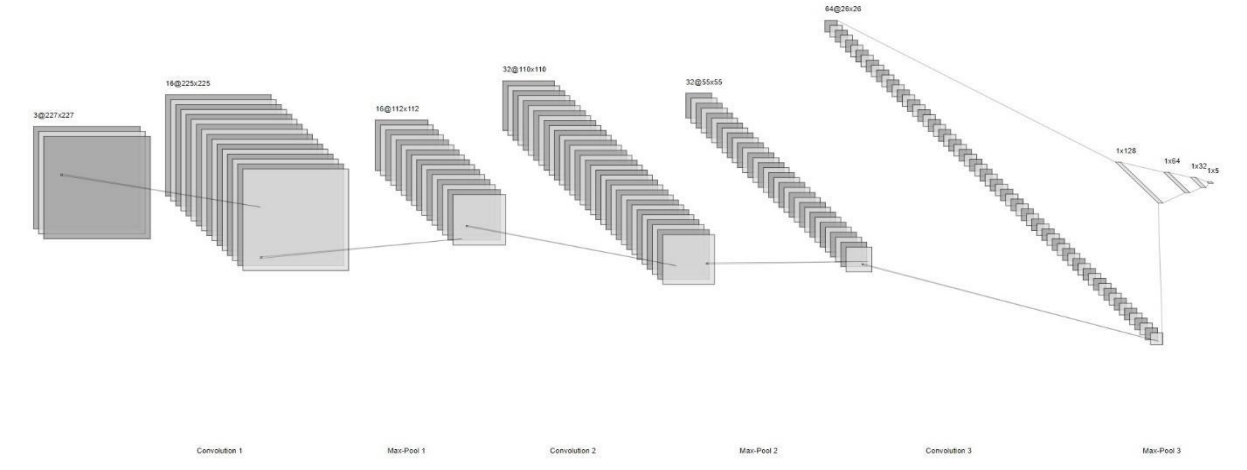

Fig 5. Flow of CNN

The first layer of CNN consists of a Convolution Layer with leaky ReLU(Leaky Rectified Linear Unit). This layer has a filter of size 3, padding, and stride of size 1. Convolution Layer 1 is followed by the

Max pooling layer with a size of 2 and then by the Batch Normalization Layer. The second and third CNN layers follow the same structure as the first CNN, but the only difference between these three is that the first layer has 16 filters, the second has 32 filters, and the third layer has 64 filters.

After the final CNN layer, the perceptron network comes into the picture. First, we will straighten all the neurons, and then three layers will have 128, 64, and 32 neurons with Leaky ReLU activation function. Finally, in the output layer, we have five neurons as we have five distinct classes.

Loss Function

We are working with a multiclass classification problem. There are several loss functions, such as sigmoid loss, SoftMax log loss function, and cross-entropy loss function. We have chosen a cross-entropy loss function.

Optimizer

Pytorch provides us with several optimization functions. The purpose of the optimization function is to change weight during training methods. We can have Adam, RMSProp, NAdam, SGD. We have used SGD with several hyperparameters, such as a learning rate of 0.04 and weight decay of 0.001. By mentioned configuration, we use 10 epochs to train our model, and you can see the training output in the attached image.
We have trained the model for ten epochs using the mentioned configuration, and we can see the training output in the attached image.