**Convolutional Neural Network**

Convolutional Neural Network is a sequence of layers, and every layer of a ConvNet transforms one volume of activations to another through a differentiable function. A simple ConvNet for CIFAR-10 classification could have the architecture [INPUT - CONV - RELU - POOL - FC]. In more detail:

* INPUT [32x32x3] will hold the raw pixel values of the image, in this case an image of width 32, height 32, and with three color channels R,G,B.
* CONV layer will compute the output of neurons that are connected to local regions in the input, each computing a dot product between their weights and a small region they are connected to in the input volume. This may result in volume such as [32x32x12] if we decided to use 12 filters.
* RELU layer will apply an elementwise activation function, such as the max(0,x)max(0,x) thresholding at zero. This leaves the size of the volume unchanged ([32x32x12]).
* POOL layer will perform a downsampling operation along the spatial dimensions (width, height), resulting in volume such as [16x16x12].
* FC (i.e. fully-connected) layer will compute the class scores, resulting in volume of size [1x1x10], where each of the 10 numbers correspond to a class score, such as among the 10 categories of CIFAR-10. As with ordinary Neural Networks and as the name implies, each neuron in this layer will be connected to all the numbers in the previous volume.

A config file was created to change the parameters automatically rather than hardcoding the constants. The config file was saved as a JSON file and was read by the code to fit the model.

Following were the constants which were changed using the config file.

Batch Size: Which varied from 32 – 128.

Number of Classes: Which remained the same as 10 since CIFAR10 Dataset is of 10 classes.

Epochs: Usually varied from 20 – 200

Data Augmentation: which is True

Optimizer: Adam, RMSprop

Learning Rate: 0.01, 0.001

Loss: Categorical Crossentropy, Binary Crossentropy

AWS ID and Access Key

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Models | Epochs | Batch Size | Number of layers | Learning rate | Activation Functions | Dropout rates | Train Data | Test data | Train Accuracy | Test Accuracy |
| CNN1 | 200 | 32 | 6 | 0.0001 | relu,relu,relu,relu,relu,softmax | 0.25,0.25.0.5 | 50000 | 10000 | 0.79 | 0.74 |
| CNN2 | 100 | 32 | 6 | 0.0001 | relu,relu,relu,relu,relu,softmax | 0.25,0.25.0.5 | 50000 | 10000 | 0.76 | 0.73 |
| CNN3 | 75 | 32 | 6 | 0.0001 | relu,relu,relu,relu,relu,softmax | 0.25,0.25.0.5 | 50000 | 10000 | 0.75 | 0.72 |
| CNN4 | 100 | 32 | 9 | 0.0001 | relu,relu,relu,relu,relu,relu,relu,relu,softmax | 0.25,0.25.0.5 | 50000 | 10000 | 0.74 | 0.78 |
| CNN5 | 100 | 64 | 9 | 0.0001 | relu,relu,relu,relu,relu,relu,relu,relu,softmax | 0.25,0.25.0.5 | 50000 | 10000 | 0.86 | 0.82 |
| CNN6 | 100 | 128 | 9 | 0.0001 | relu,relu,relu,relu,relu,relu,relu,relu,softmax | 0.25,0.25.0.5 | 50000 | 10000 | 0.92 | 0.82 |

The following were different models which were tried and tested to get the best accuracy possible:

The best model which was found was a CNN with 19 layers. This model consists of 6 Conv2D layers, 8 activation Layers (RELU), 1 fully connected layer, 3 Pooling layers and 1 softmax layer.

The Final Model gave an Accuracy of 0.82.

