# Convolution Neural Network (CNN)

# **CIFAR Dataset**

**Package used in python**

Keras.datasets

cifar10 - To load the Cifar 10 Dataset

keras.utils

to\_categorical - Convert the class vector to binary matrix

keras.models

Sequential - layer of Neurons

keras.layers

Conv2D - creates a convolution kernel

MaxPooling2D - To perform maxpooling on layer

Dropout - Drop the neurons randomly ignored to avoid Over fitting

Dense - Connect all the layers

Flatten - Flatten the Data to a Single Array

BatchNormalization - Normalize the activation's of previous layer at each batch

Matplotlib Data Visualization

**Loading the data**

Loaded the CIFAR 10 dataset in Python from Keras Dataset. And the data is spilt into Train X , Train Y and Test X , Test Y

**EDA**

Train Y and testY - Converted class vector integers to binary class matrix with to\_categorical

trainX and testX - Converted into float32 type data

trainX and testX - Data normalized by divided with 255

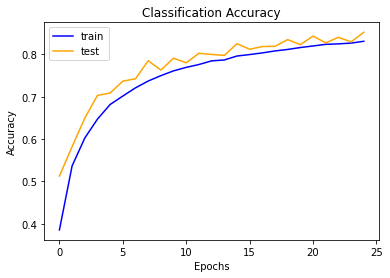
**Data Partitioning**

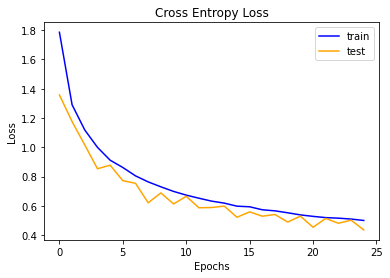
Data was split into Training and testing. Training contains 50000 image data and Testing contains 10000 image data.

**Modeling and Result**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model with Conv2D | MaxPooling** | | | | |
| Epoches | Activation | Optimier | Accuracy | Time Taken for 1 Epoches |
| 10 | ReLu | SGD | 66.45 | 110 ~ 120 sec |
| 20 | ReLu | SGD | 66.65 | 110 ~ 120 sec |
| 100 | ReLu | SGD | 67.07 | 110 ~ 120 sec |
|  |  |  |  |  |
| **Model with Conv2D | MaxPooling , Conv2D | MaxPooling** | | | | |
| 10 | ReLu | SGD | 68.44 | 200 to 210 sec |
|  |  |  |  |  |
| **Model with Conv2D | MaxPooling** | | | | |
| Epoches | Activation | Optimier | Accuracy | Time Taken for 1 Epoches |
| 10 | ReLu | Adam | 66.4 | 120 ~ 130 sec |
|  |  |  |  |  |
| **Model with Conv2D | MaxPooling , Conv2D | MaxPooling** | | | | |
| 10 | ReLu | Adam | 73.41 | 160 ~ 130 sec |
| 50 | ReLu | Adam | 71.91 | 160 ~ 130 sec |
|  |  |  |  |  |
| **Model with Conv2D | MaxPooling** | | | | |
| Epoches | Activation | Optimier | Accuracy | Time Taken for 1 Epoches |
| 10 | ReLu | Adam | 66.45 | 110 ~ 120 sec |
| 20 | ReLu | Adam | 66.65 | 110 ~ 120 sec |
| 100 | ReLu | Adam | 67.07 | 110 ~ 120 sec |
|  |  |  |  |  |
| **Model with Conv2D | MaxPooling| Dropout** | | | | |
| 50 | ReLu | Adam | 82.9 | 280 ~ 320 sec |
|  |  |  |  |  |
| **Model with Conv2D | MaxPooling| BatchNormalization|Dropout** | | | | |
| 25 | ReLu | Adam | 84.91 | 430 ~ 450 sec |
|  |  |  |  |  |
| **Model with Conv2D | MaxPooling| BatchNormalization|Dropout| Augumention** | | | | |
| 40 | ReLu | Adam | 85.7 | 450 ~ 480 sec |

**Accuracy and Loss Plot**





2)

**Convnet Filter**

A Convnet Layer is the simple filter to an input and the output of the convnet layer is called feature map or activation map.

**Maxpool Filter**

Max pooling reduces the spatial size of a layer keeping just the maximum values.

3)

1. 10 Filters so 10 Activation Maps.
2. n\*m\*l+bias\*no of layers

5\*5\*3+1+10 = **760** Parameter

1. (W−F+2P)/S+1

= (64-5\*2\*2)/1+1

= 64

Output will be 64x64x10

1. The Following steps or methods can used to over come the overfitting model

I) **Early Stopping**

Stopping the epoches before the model starts over fitting.

Ii) **Dropout**

Randomly dropping the neuron to reduce the data.

Iii) **Data Augmentation**

This method can be used Increase the dataset by cropping, flipping, rotating the existing dataset.

Vi) **Batch Normalization**

Batch Normalization reduce the internal co-variate shift and instability in distributions of layer activation in networks can reduce the effect of overfitting.