CIFAR-10 Data Object Classification using CNN

1. Introduction

In neural networks, Convolutional neural network (ConvNets or CNNs) is one of the main categories to do images recognition, images classifications. Objects detections, recognition faces etc., are some of the areas where CNNs are widely used.

CNN image classifications takes an input image, process it and classify it under certain categories (Eg., Dog, Cat, Tiger, Lion). Computers sees an input image as array of pixels and it depends on the image resolution. Based on the image resolution, it will see h x w x d(h = Height, w = Width, d = Dimension). Eg., An image of 6 x 6 x 3 array of matrix of RGB (3 refers to RGB values) and an image of 4 x 4 x 1 array of matrix of grayscale image.

****Objective: To Build an Image Classifier to classify images of CIFAR-10 Data****

2. Problem Definition and Algorithm

2.1 Task Definition

Building an Image Classifier to classify images of CIFAR-10 Data .The dataset stands for the Canadian Institute For Advanced Research (CIFAR). CIFAR Data set is widely used for machine learning and computer vision applications.

2.2 Algorithm Definition

CNNs are regularized versions of multilayer perceptrons. Multilayer perceptrons usually mean fully connected networks, that is, each neuron in one layer is connected to all neurons in the next layer. The "fully-connectedness" of these networks makes them prone to overfitting data. Typical ways of regularization include adding some form of

magnitude measurement of weights to the loss function. CNNs take a different approach towards regularization: they take advantage of the hierarchical pattern in data and assemble more complex patterns using smaller and simpler patterns. Therefore, on the scale of connectedness and complexity, CNNs are on the lower extreme.

3. Experimental Evaluation

3.1 Methodology

First of all I have loaded the entire cifar-10 dataset into four different divisions namely, X_train, X_test, Y_train, Y_test. Then after that I have done the data visualization using pandas and numpy library. Then I bought down the X_train and X_test values in the range of 1.

After all these stuffs, I built my CNN model, the model is a Sequential model made using one input layer of 32 neurons and followed by another 32 neuron layer. Then I added two 64 neurons layers. All the layers were having kernel size (3,3),dropout layers of value 0.2, 0.3 respectively and activation function as 'relu' and then I added a flatten layer and following that I added a dense layer of 128 neurons with 'relu' activation.

Following this layer I added a BatchNormalization layer and then a DropOut layer of value 0.4.

Then finally I added the last Dense layer of 10 neurons (Since we have 10 classes) with activation function as 'softmax'.

Model is compiled using the following parameters:

```
model.compile(optimizer='adam',loss='sparse_categorical_crossentropy',metrics=['accuracy'])
```

And Model is trained using following parameter:

```
history = model.fit(X_train,y_train,batch_size=128,epochs=10,verbose= 1,validation_data=(X_test,y_test))
```

3.2 Results

The results came out as follows:

First the summary of the model:

Total params: 272,298

Trainable params: 272,042

Non-trainable params: 256

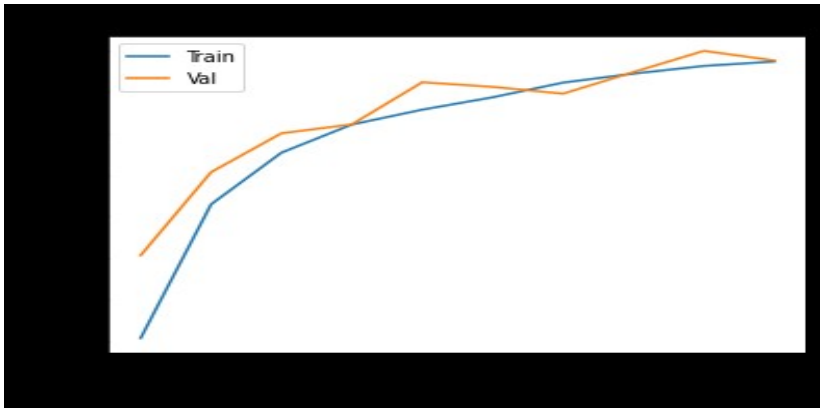
The Model was trained for 10 epochs and with batch size 128.

After evaluating the model we got an val_accuracy of 73.77%

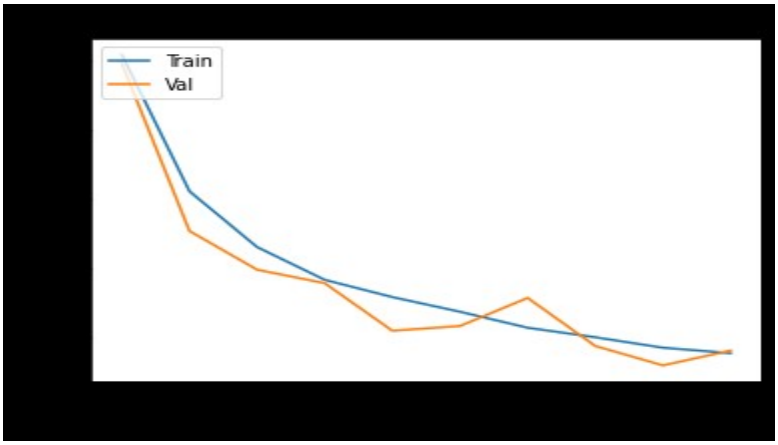
313/313 [==============================] - 8s 27ms/step - loss: 0.7608 - accuracy: 0.7377
[0.7608122825622559, 0.7376999855041504]

I plotted graphs for comparing the Model accuracy and Model loss and it came out as follows:

Model Accuracy Graph:



Model Loss Graph:



4. Conclusion

For all the classes available in the cifar-10 dataset I made a dictionery of the classes as follows:

```
class_name = {
  0 :'airplane',
  1: 'automobile',
  2: 'bird',
  3: 'cat',
  4: 'deer',
  5: 'dog',
  6: 'frog',
  7: 'horse',
  8: 'ship',
  9: 'truck'}
```

Now when we have to predict any image we write the following codes and get the result:

Here we see that X_test[45] has this image:

Let's see what our model predicts:

```
print(class_name.get(np.argmax(pred[45])))
```

output: `truck`

Our Model Predicted correctly,

Hence Our model is working properly.