INTER IIT QUALIFIERS
STAR CLUSTER IDENTIFICATION

RITHWIK KUKUNURI

PANKAJ VATWANI

SAMMED S KAGI

MANAS BEDMUTHA

INDEX:

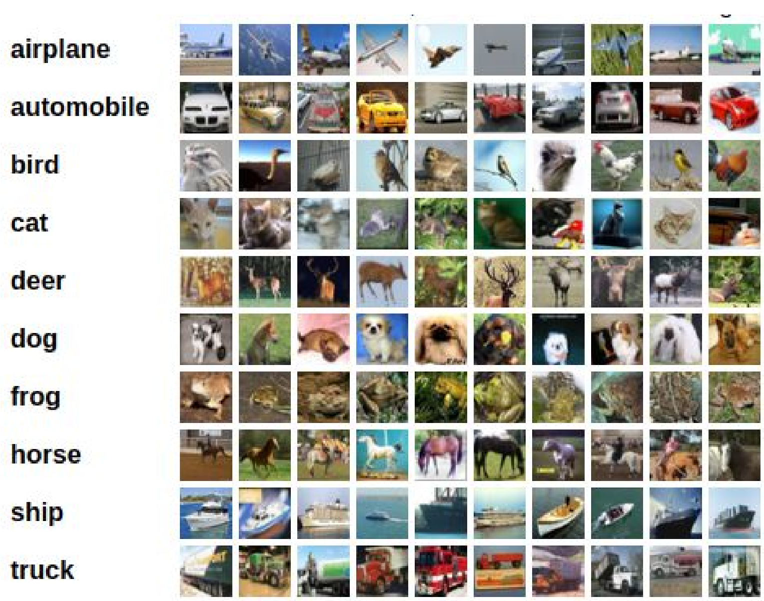
Task:

Classification of the CIFAR - 10 dataset by Studying, Analysing and Interpreting the data in terms of Confusion Matrix

Introduction

CIFAR-10 is an established computer-vision dataset used for object recognition. It is a subset of the 80 million tiny images dataset and consists of 60,000 32x32 color images containing one of 10 object classes, with 6000 images per class.

There are 50,000 train images and 10,000 test images. The, completely mutually exclusive, classes can be found in the below image.



Approach:

We start with exploring Conventional Methods to study and analyse the data and get insights about it.

Analysing the Data -

This is done by means of Exploration of the Data images. We plot random images to see the clarity of the images, and the variety of images under a class.

After this we found out that the images have a range of varied scenes that need to be classified hence the coloration and structure both have equal importance.

We went through a step by step approach as we went on increasing complexity of networks based on factors till training time and accuracy tradeoffs.

Conventional Methods -

The first method tried was Feature Reduction based on PCA . When we tried the PCA based model combined with Dense Layers, the model was not able to converge well. So we had to discard PCA based methods.

Next, we tried the KNN Classifier. We found that it took very long for training even for few epochs. Owing to lack of time, we sought our focus to other methods.

Switch to Deep Learning -

Since we weren't getting very high accuracies or efficiency in training times, we switched to the Deep Learning Methods.

Method I: Keras Example Code

We reran Keras' implementation sample code on the CIFAR 10 data set. As claimed we could reproduce results to a significant error margin. (Link)

Method II: Exploring the State of The Art

Based on this Github Repo we found a compilation of major state of the art methods applied. By a quick review of the methods, we tried implementations of a few of them,

Method III: Individual Implementations

Parallel to existing methods, we experimented with variety of Parameters and Hyperparameters to improve upon the accuracy score.

<u>Since we believe the Final Problem Set will not have much public codes on the internet; our major focus lied on this step.</u>

Understanding Existing Architectures -

We went through a few research papers and understood the unique CNN's used by each of them and tried to implement them.

Architecture Used:

We have read a few research papers and we selected a few innovations from all of them to combine and generate a new architecture.

We have been using

- Dropout Layers
- Adam Optimizer, NAdam Optimizer
- Batch Normalization
- Exponential Linear Unit

Innovations:

Using a combined model based on Deep Learning as well as conventional methods, gives us the best out of both worlds.

Final Model - 1(Link):

We trained our final model by following a blog post. The Model consisted of Deep Neural Network with Exponential Linear Unit as Activations and Convolutions followed by Batch Normalization. In the end we flattened the results and connected with a few Dense layers to get the predicted label. The Resulting accuracy as 84.55%.

Final Model - 1 + SVM (Link):

Then we removed the Final Softmax and Dense layer. We computed the output Dense values for the input data and then passed them through a SVM for classifying the images. The resulting accuracy was 85.09%.

Final Model - 2: (Link)

The Model consisted of Deep Neural Network with Rectified Linear Unit as Activations and Convolutions followed by Batch Normalization.Max Pooling layers are also used in the network.We also used dropouts in our network to avoid overfitting. In the end we flattened the results and connected a few Dense layers to get the predicted label. The Resulting accuracy around 84%.

Results:

Model/Approach	Accuracy (approx)
PCA + NN	32.50
Keras Example Code	79.65
Final Model - 1	84.55
Final Model - 1 + SVM	85.09
Final Model - 2	83.39

Confusion Matrix:

Final Model - 1:

```
[883  14  14   5  15   1   3  10  35  20]
[  7 944   0   2   2   1   0   0   7  37]
[ 53   5 744  26  84  20  28  28   6   6]
[ 21   6  44 621  66 126  39  52  11  14]
[  7   4  15   9 890  12   6  51   6   0]
[ 12   7  30  87  37 749  12  57   1   8]
[  8   5  23  28  42   9 873   7   3   2]
[  6   1   5  11  22  13   2 938   0   2]
[ 34  21   3   2   3   1   3   4 914  15]
[ 19  56   1   4   1   0   1   5  14 899]
```

Final Model - 1 + SVM:

```
[879  10  24  11  13   1   2   6  35  19]
[  6 938   0   3   2   1   0   0   8  42]
[ 42   4 778  33  60  21  31  20   6   5]
[ 14   4  47 690  40 119  35  32   8  11]
[ 10   4  24  26 864  20   9  38   5   0]
[ 10   7  29 110  30 759  12  36   0   7]
[  8   5  27  43  28   8 873   3   3   2]
[  5   1  10  20  27  26   1 908   0   2]
[ 32  15   3   5   3   1   3   3 916  19]
[ 20  49   2   5   1   0   1   4  14 904]
```

Final Model - 2:

```
[[865  12  15  13  10   3   6   4  49  23]
 [  6 923   0   5   5   2   2   2  18  37]
 [ 67   2 730  39  63  35  39  13   8   4]
 [ 20   2  34 703  65  94  39  21  11  11]
 [ 14   1  30  37 859   6  20  28   5   0]
 [  7   2  29 162  42 720  12  23   1   2]
 [  7   1  22  37  33  13 876   3   6   2]
 [ 12   1  15  30  55  20   0 856   6   5]
 [ 25   4   8   4   1   4   3   4 935  12]
 [ 17  67   2   8   4   3   2   4  21 872]]
```

--**********---**
