10-601b: Project Report
Image classification on CIFAR-10 dataset

Anonymous Author(s)
Affiliation
Address
`email`

Abstract

Image classification based on image features is a challenging job in Machine Learning. In this project, we explore various methods to classify the images into different categories using machine learning techniques by designing classifiers and tuning the parameters to give better accuracy. The report contains the results obtained using k-nearest neighbours, k-means and support-vector machines. Significant amount of effort has been put to improve the efficiency of these classifiers by tuning the parameters of the classifiers as well as experimenting with various feature sets.

1 Introduction

Image classification is an interesting research domain in both computer vision and machine learning. The state-of-the-art image classifiers have an accuracy upto 94%, which are claimed to be more than humans. To achieve such a remarkable accuracy, one needs both intellectual effort as well as engineering effort. In this project, we try to address both these aspects by implementing the state-of-the art classifiers and fine-tuning them to increase the accuracy.

Machine learning techniques can be used for the task of classification. the basic classifiers are expected to give descent performance when compared to a dummy/random classifier. The more advanced classifiers would need more training and sophisticated mechanism to understand and tune the parameters of the model. More training could sometimes leads to over-fitting the training data, resulting in false predictions during test time. Hence, a trade-off is made between the achieving accuracy and fitting the training data. Yet more complicated non-linear classifiers needs much more training time. In this project, we explore some of the basic classifiers followed by few advanced classifiers like support vector machines (SVM).

The report is structured as follows. The first section talks about the initial data analysis done on the provided dataset. The next section talks about implementation using the basic classifiers like k-means and k-nearest neighbours. Finally, support vector machines are discussed in details followed by results.

1.1 Related work

The best results on CIFAR dataset is given by deep neural networks. The list of the latest are summarized in `http://zybler.blogspot.com/2011/02/table-of-results-for-cifar-10-dataset.html`. From the trends of the accuracy we can see that, the basic classifiers can achieve an accuracy of 72.28%. To achieve better results, we would need to implement one of the variations of deep network.

054
055
056
057
058
059
060
061
062
063
064
065
066
067
068
069
070
071
072
073
074
075
076
077
078
079
080
081
082
083
084
085
086
087
088
089
090
091
092
093
094
095
096
097
098
099
100
101
102
103
104
105
106
107

2 Exploratory data analysis

Before starting with the machine learning techniques, we tried to understand the dataset we were provided. We looked that the **distribution of training labels** provided to us. This was to ensure the training samples are not skewed towards a certain label. We found that 'airplane' class had the least number of training samples (about 470) and 'frog' class had the maximum (about 520). The difference is minimal and should not affect training.

We looked at the **colour distribution** in the overall training dataset and the colour distribution in individual classes. The images seem to follow a normal distribution around different means and have different variances for the red, green and blue components. This was expected and should not affect training of our algorithms at all.

Finally, we constructed the **average image of each class**. This was to look for specific patterns that might exist in the dataset. Any sharp patterns in the average image would skew training performance. However, we did not find any such patterns with the given dataset.
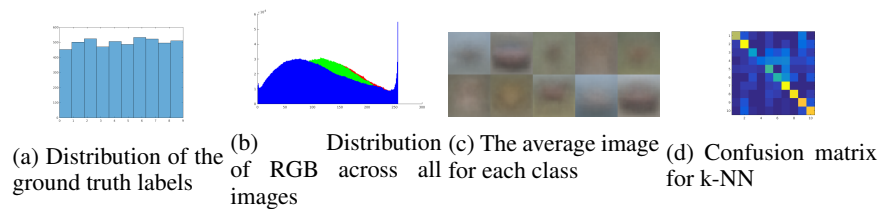

(a) Distribution of the ground truth labels (b) Distribution of RGB across images (c) The average image all for each class (d) Confusion matrix for k-NN

Figure 1

3 Classification with K-means and K-Nearest Neighbours

The very first approach we tried was using the simple k-means and k-nearest neighbours approaches. These are simple to implement and help understand the dataset provided to use better. Details of problems with these techniques are provided in detail below.

3.1 K-Means

The first attempt was to use the raw image data and K-means clustering algorithm to find clusters in the pixel RGB values. This approach took a long to train and produced bad results. With a constant classifier (one that always returns a particular class), we would have about 10% accuracy. We achieved about 15.3% accuracy with the naive K-means classifier.

The next attempt was to cluster responses from a filter bank. We used two sets of filter banks - regular filters and **Gabor filters**. The training time for this increased further but produced a very tiny increase in accuracy (about 16.7%). Given the training time required for K-means and the accuracy, we decided to try K-nearest neighbours. The voting algorithm would be robust to noise in the training inputs.

3.2 K-Nearest Neighbours

We used the K-nearest neighbours on three different feature spaces. The first was raw images. The accuracy was almost as good as the K-means classifier. Second, we tried passing images through the filter bank and Gabor filters used in the previous section. The results were better than K-means (29%). This increase in accuracy can be attributed to the classifier being robust when the input training samples are noisy. We used $K = 51$ to achieve this result.

To improve accuracy further, we calculated the Histogram of Gradients features on the image and used those for classification. This improved the accuracy to 53%. The improvement was because certain classes in the image can be represented accurately with orientation histograms (cars, for example).

Technique	Training time	Accuracy
K-means (filter bank)	1 hour	16.5%
K-means (Gabor filters)	1-1.5 hours	16.7%
K-Nearest (filter bank)	1 minute	28.6%
K-Nearest (Gabor filters)	1 minute	29.4%
K-Nearest (HOGs)	1 minute	53.1%

Table 1: Results with K-means clustering and K-nearest neighbours

4 Support vector machines

Support vector machines (SVM) are linear classifiers which classify the given feature vectors into two classes. For training the SVM we need input features and the labels from the training set. The dataset contains 32x32x3 size images, which needs to be converted to a feature vector. In this paper we have used histogram of gradients (HOG) as the basic feature. The HOG features divides the image into small block and calculates the histograms of the gradients. The gradients contain information regarding the direction in which the intensity changes the maximum. The HOG feature descriptor quantizes the gradients into 9 directions and returns the histogram along those directions. HOG descriptors holds information regarding the shape of the objects.

To use the SVM as classifier for the CIFAR dataset which has 10 classes, we need to train each 10 separate SVM classifiers. To train any one of the classifiers for a particular class, there would be some positive samples and many negative samples. For 10 classes, one tenth of the data would be positive and nine tenth of the data would be negative sample. This imbalance between positive and the negative samples is overcome by randomly picking up n negative samples (equal to positive samples).

In another experiment, in order to overcome the imbalance between the groups we trained each class with every other class. So class 1 is trained with class2,...class10 using SVM resulting in 9 SVMs for each class. So 10 classes we would have 90 SVMs. During classification the classification is found using the equation 1 for all the 90 SVMs. The SVM separating class1 to class2 is same as class2 to class1, which results in total of 45 training SVMs.

$$y^{new} = W^T X + b \tag{1}$$

In each class all the 9 classifiers vote whether the data belongs to that class or not. For example for class1, the SVMs corresponding to class1 vs class2, ... class10 vote if the class belong to class1 or not. We count the number of vote for class1. Similarly for class2 and so on. The final classification depends on the highest number of votes. The class which has most number of votes using the simple classifiers wins. This voted SVM technique is found very effective and the results have boosted in order of 7-10%.

In addition the range of image is 8-bit (0-255). This might not be a good option for SVMs, as the range of the data is more. In order for the SVMs to be a robust classification its is recommended that the input dataset varies in the range (-1,1). So the HOG features are normalized in this range. This improves the classification efficiency by 3-4%.

To further improve the classification, we have implemented contrast normalization. The images might have been taken in various lighting conditions. The day-light images will have different contrast than low-light images. The intensity of these two images would be different and so in-order to make the images robust to the lighting conditions, we subtract the mean of the intensity from every pixel. This would normalize the contrast among all the images. An example of the contrast normalization is shown in the Figure 2. Contrast normalization inproves the accuracy by 1-2%.

Also, the margin in SVM plays an important role to decide the accuracy of the classifier. This margin decides decides the extent of over-fitting in the model. A margin of $C = 0.1$ yields optimum performance.

SVM technique	Accuracy
Normal SVM	44.3 %
SVM with data Normalization	50.8%
SVM with data and contrast normalization	51.9 %
Voted SVM	54.3%

Table 2: Results of calssification with SVMs

(a) Image without contrast normalization

(b) Image after contrast normalization
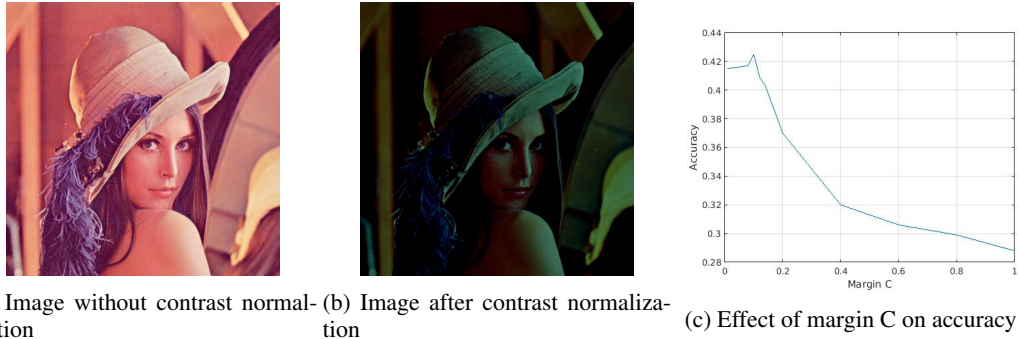
(c) Effect of margin C on accuracy

Figure 2: The effect of contrast normalization and the SVM margin

5 Results

Upon analysing the confusion matrix for K-Nearest neighbours (Figure 1 (d)), we found that the perfectly classified labels belonged to classes with several sharp and well defined edges - cars, airplanes, ships, trucks. The horse and frog classes seem to perform quite well too - however, the bird, cat, deer and dog classes are harder to classify.

From this, we can infer why the classifier fails. At such low resolutions, the HOG features are unable to capture the distinctive features of these classes. Most of them seem to be mammals with 4 legs and thus the confusion. Using HOG with higher resolutions (with smaller cell sizes), we found that accuracy of the classifier decreased to below 50% with no increase in accuracy on the problematic classes. The results of k-means and k-nearest neighbours are summarized in Table 1.

The results obtained with the SVMs are summarized in the Table 2. From the table we can see the improvement in accuracy by implementation of the voted SVM. The accuracy is further improved by using contrast normalization and data normalization as discussed in section 4.

6 Future work

Until now, we have achieved a maximum accuracy of 55%. We have used simple techniques like k-means, k-nearest and more sophisticated techniques like SVM. Our next trial will be with a discriminative sum-product networks. The paper claims to achieve a high accuracy using a network approach and using gradient descent.

References

[1] Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection,*CVPR, 2005*

[2] Christian Saigian, et al. Rapid Biologically-Inspired Scene Classification Using Features Shared with Visual Attention, Pattern Analysis and Machine Intelligence, 2007

[3] R.Muralidharan, et al. Object Recognition Using Support Vector Machine Augmented by RST Invariants, *IJCSI, 2011*

[4] R Gens, et al. Discriminative learning of sum-product networks, Washington University, 2012