Character Recognition from Google Street View Images

Ritesh Kumar (11602) Srikant Singh (12729)
Guide : Professor Amitabha Mukerjee

Introduction

The recognition of arbitrary characters from natural photographs is not a simple or easy problem. In this project we wish to tackle a similar problem, which is the recognition of single characters from Google Street View images. A good solution to this problem can be used in applications involving addition of information to existing data on Google Maps, helping the visually impaired in navigation around places, etc.

Problem Description

The problem involves classification, where our work involves recognition of the single character in the given image and then assigning it to its proper character class. The approach to this problem involves extraction of features from images , which is followed by training of our model to predict the class of subsequent images that are given.

Fig. 1 – Example of the images in our data set

Methodology

As mentioned above we divide our work into two parts –

1. Feature extraction
2. Learning on Feature Vectors

Feature Extraction –

- Histogram of Oriented Gradients (HOG) : HOG is a feature descriptor that finds the local 1-D histogram of gradient features over pixels of the cell, and after normalization the descriptors are fed into a recognition system. We tries different cell sizes for images (namely 2*2 , 4*4 and 8*8) and chose the 4*4 cell size as it was a balance between increase in dimensionality and encoding of enough information.

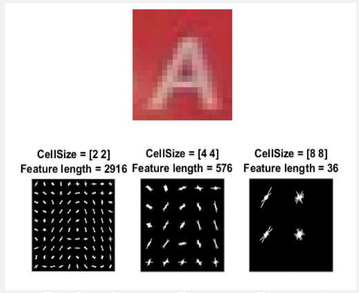

Fig 2 – Image Feature Extraction using HOG

- Scale-invariant feature transform (SIFT) : In SIFT, features and descriptors are extracted for all the images. Then we cluster the set of feature descriptor for the amount of bags using k-mean clustering. Given a new image we extract the descriptor and match it with the vocabulary to build histogram.

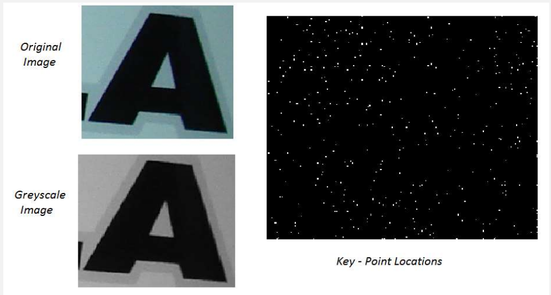

Fig 3 – Image Feature Extraction using SIFT

- Vector of pixel values : One of the simplest feature extraction implementation. The intensity of image pixels is used as a feature for classifiers. We converted the images to greyscale before proceeding.

Learning on Feature Vectors –

- Random forests : They are extensions of decision trees incorporating averaging between multiple decision trees to give better results. They show lower variance than decision trees.
- K – nearest neighbours with LOOF_CV : The best accuracy we got was for K=1. In LOOF-CV model a single data point is tested after being trained with the remaining points in the data, which makes it intractable to perform. k-NN is used to do it efficiently. We examined with both Euclidean and cosine distance function.
- Support Vector Machines : SVM classifier with linear kernel was used to learn the feature vectors.

We implemented the following four algorithms –

- Pixel values as image features followed by Random forests for training data.
- HOG followed by SVM
- Pixel value extraction followed by K-NN.
- SIFT followed by SVM

The results obtained were submitted to kaggle.com[1] for calculation of their accuracies.

Results

Algorithm Used	Accuracy obtained
Pixel Value Vector – Random Forest	46.368%
HOG – SVM	57.491%
Pixel Value Extraction – K-NN	43.586%
SIFT – SVM	54.077%

Table 1 – Algorithm Vs. Accuracies

Conclusions

- From the results we can see that HOG and SIFT are better for feature extraction than Vector of pixel values.
- Our accuracy suffers from conflicting characters such as ‘0’ , ‘O’ and ‘D’; ‘1’ ,’I’ ‘(uppercase ‘i’)and ‘l’ (lowercase ‘L’). Also context from words could have helped us , but as we use single character images, hence we can’t avail that benefit.

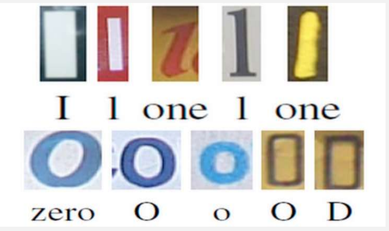

Fig – 4

Future Work

- We can use boosting algorithms, which essentially involve using 2 or more classifiers instead of the one we are using at a given time. In case of conflicts we can use both the classifiers to arrive at a result.
- We can use neural networks as classifiers. Many OCR algorithms have been developed using neural networks, especially deep convolution neural nets. They give the highest accuracies with appropriate feature extractors (accuracies up to 97.28% have been obtained using GoogleNet)[2]

References

1. *www.kaggle.com/c/street-view-getting-started-with-julia*
2. *Recognizing From Google Street View Images; Guan Wang, Jingrui Zhang*