Landmark Recognition

CS6375 MACHINE LEARNING – FINAL PROJECT REPORT

THE UNVERSITY OF TEXAS AT DALLAS

Pushpita Panigrahi (pxp171530)
Sankalpa Rath (sxr173830)
Siddharth Swarup Panda (ssp171730)
Sushrut Patnaik (sxp175331)

1. Overview

Most of us would have been in a situation where we would have forgotten the name of a place/monument we had visited the previous year or during the previous vacation. Landmark recognition can help here! The Landmark recognition project aims to correctly classify landmarks from a set of images. Landmark recognition uses OpenCV to predict landmark labels directly from image pixels, to help people better understand and organize their photo collections.

2. Dataset Description

The dataset considered for the project is taken from:

https://www.kaggle.com/c/landmark-recognition-challenge/data

The dataset contains URLs for images and each image depicts one landmark. The data set was constructed by clustering photos with respect to their geolocation and visual similarity. Each image has a unique id (a hash) and each landmark has a unique id (an integer).

The total training data size given in the original dataset is 1225029 with 14949 different classes or unique images. The number of attributes in data set is 3 namely ID (a hash value which are not using anywhere in our model), URL (url of the image), landmark_id (unique class label for each landmark). We have considered 30 classes or different landmarks for the project with 1150 images in the dataset. The Test dataset provided in the Kaggle website doesn't have class labels for the landmarks, so we are splitting the training dataset into 75-25 % to use as test and train data. We also verified that there are no missing data in the training dataset. There are no features given directly in the dataset so we are using HOG descriptor to extract features.

A snapshot of the dataset below:

id	url	landmark_id
c3dc215809020774	http://lh6.ggpht.com/-Gp0vNaw6YwU/R2qzGaHSuoI/AAAAAAAABsA/lhx7oEpKd_o/s1600/	30
c90ca2e5e4e8e94c	http://lh6.ggpht.com/-8A_ND4KGi-g/SIldGajvMJI/AAAAAAAAEDo/1egPBp5XB7k/s1600/	5
3095e6731e2ed7ce	https://lh3.googleusercontent.com/-iOAW2Vhgj0s/T3-kjvzpnCI/AAAAAAAAASo/lxPkHZXZ2yg/s1600/	22
78dfba75dba02234	http://lh4.ggpht.com/-okvXZmDTgv8/SsZjNwT-ocI/AAAAAAAAALI/pGIyADZDsK8/s1600/	18
3baa70bb2d2a9811	https://lh6.googleusercontent.com/-0z9F-mc1PAU/UkdYAdfxYtI/AAAAAAAAAg4/s9MX2cehoVk/s1600/	5
265259306f6f5b3f	https://lh4.googleusercontent.com/-MVjsZBqiBqc/RxuKx01RU6I/AAAAAAAAALc/Y1AcjDK0zOg/s1600/	8
f0179fc5b2121faf	http://lh5.ggpht.com/-5zJbdVKQpgs/SoJxEUPZdTI/AAAAAAAAAFs/ObqCzq7JvC4/s1600/	29
42785bc0d656b5ef	https://lh3.googleusercontent.com/-EvDGsEq3Uoc/TJgj1vb4UQI/AAAAAAAAD7k/DXSLj4drKk4/s1600/	5
c15c24c1417bac65	http://lh3.ggpht.com/-77NiOMFNqjQ/TeYPrPdUMlI/AAAAAAAAARg/9D9bBI4ibuw/s1600/	5
ddb1fe8d8f21e0e9	http://mw2.google.com/mw-panoramio/photos/medium/37226227.jpg	7
5f139f423e994b0c	http://lh5.ggpht.com/-VxU4OT-mwL8/Rkqo8TRJinI/AAAAAAAAAwI/cqutkNoRAjQ/rj/	26
1c074abe1af2a07e	http://lh5.ggpht.com/-D1Ca3CqIyoQ/Twyj5tMGLII/AAAAAAAABEI/0RUBmBKX2u8/s1600/	5
09aa44a18d2c81e7	https://lh5.googleusercontent.com/-LwRsEMMv0Pk/SkzEmBpcQ-I/AAAAAAAAHz4/xg6SYlmBCPY/s1600/	8
a96916e83669db21	https://lh5.googleusercontent.com/-jWOd3SFNfuY/SmcLdVCkJBI/AAAAAAAADo4/NZv_Rn7ZpsU/s1600/	5
3e6e02b2ea7c0d45	http://mw2.google.com/mw-panoramio/photos/medium/28810563.jpg	7
2fb28a4cc0d14747	https://lh4.googleusercontent.com/-ZvCDEw5ypyc/UfT_NQ7mpzI/AAAAAAAAFQg/t18n_Jh1buQ/s1600/	5
fa85e9d8ed60340e	https://lh4.googleusercontent.com/-MZ17lK9Wrjk/UkFI9gW0j8I/AAAAAAAAABY/izWDK-6rh6Y/s1600/	30

3. Preprocessing

After retrieving each image from the URLs, we have resized them to 64 columns (width) and 32 rows (height) using cv2.resize function. In the second step we have converted the images to grayscale before performing the feature extraction on them. This was done using cv2.cvtColor(). We have not taken into consideration all the instances of the dataset where the URL fetched no images. Then we used HOG Descriptor to perform feature extraction.

4. Feature Extraction

In field of machine learning and image processing, feature extraction is a method of collecting information of the interesting parts of an image in the form of a compact feature vector. It is a commonly used method in solving computer vision problems such as image recognition.

In this project we have used Histogram of Oriented Gradients for feature extraction. The histogram of oriented gradients (HOG) is a feature descriptor used in computer vision and image processing for object detection. A feature descriptor in general is an algorithm that finds the significant areas in an image. The HOG feature descriptor counts the occurrences of gradient orientation in localized portions of an image. We have used OpenCV's HOG descriptor function in our project.

cv2.HOGDescriptor(winSize, blockSize, blockStride, cellSize, nbins)

5. Principal Component analysis

After doing the feature extraction, to reduce the number of correlated features we have performed Principal component analysis. PCA is a dimensionality reduction technique which gives us less number of features than the original feature set. It produces independent features that is a linear combination of the correlated features.

6. Parameters, Results and Analysis for various Classifiers

6.1 SVM:

Support Vector machine analyses the data, define the decision boundaries and uses the kernels to convert non-linear models to linear models.

parameters = {'C':[1.0,2.0,5.0], 'kernel' : ["linear", "rbf", "poly"], 'degree' : [3,4,5],'gamma':[1.0,1.5,2.0], 'max_iter':[-1],'random_state' : [1,2,3]}

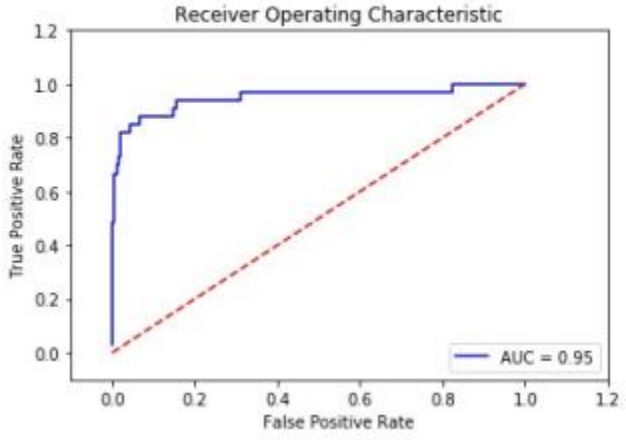
svc = SVC(probability=True)

RESULT FOR ABPVE PARAMETERS:

Accuracy : 0.68281938326

	precision	recall	f1-score	support
0	0.53	0.73	0.62	11
3	0.89	0.73	0.80	33
4	0.00	0.00	0.00	2
5	0.80	0.89	0.84	36
6	0.00	0.00	0.00	1
7	0.71	0.77	0.74	13
8	0.00	0.00	0.00	3
9	0.00	0.00	0.00	1
11	0.00	0.00	0.00	2
12	0.75	0.38	0.50	16
16	0.65	0.94	0.77	18
17	0.00	0.00	0.00	1
18	0.75	0.50	0.60	6
22	0.58	0.87	0.69	52
23	1.00	0.25	0.40	4
26	0.57	0.31	0.40	13
27	0.71	0.38	0.50	13
29	0.00	0.00	0.00	2
avg / total	0.67	0.68	0.65	227

RoC curve



6.2 GNB:

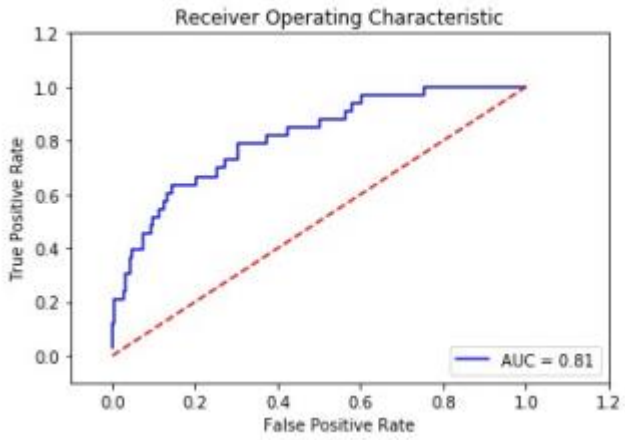
The gaussian naïve bayes classifier is a parametric method of classifier. It is a generative model that uses bayes theorem to predict the membership probabilities of each class such as a given instance belonging to particular class. The class with the highest probability is the most likely class. The distribution of values with each class is in the form of a normal distribution.

parameters = {'priors':[None]}

RESULT FOR ABPVE PARAMETERS:

Accuracy : 0.0704845814978

	precision	recall	f1-score	support
0	0.00	0.00	0.00	22
3	0.00	0.00	0.00	66
4	0.00	0.00	0.00	4
5	0.00	0.00	0.00	72
6	0.00	0.00	0.00	2
7	0.00	0.00	0.00	26
8	0.02	1.00	0.03	6
9	0.00	0.00	0.00	2
11	0.00	0.00	0.00	4
12	0.00	0.00	0.00	32
16	0.71	0.28	0.40	36
17	0.00	0.00	0.00	2
18	0.00	0.00	0.00	12
22	0.26	0.15	0.19	104
23	0.00	0.00	0.00	8
26	0.00	0.00	0.00	26
27	0.00	0.00	0.00	26
29	0.00	0.00	0.00	4
avg / total	0.12	0.07	0.08	454



6.3 KNN:

KNN is a type of instance based learning. It is a non-parametric method of classification where the input contains the K nearest training examples in feature space. There is no model creation involved , only the data is stored. When the test data is provided , a set of similar instances is retrieved from memory and the test data is classified.

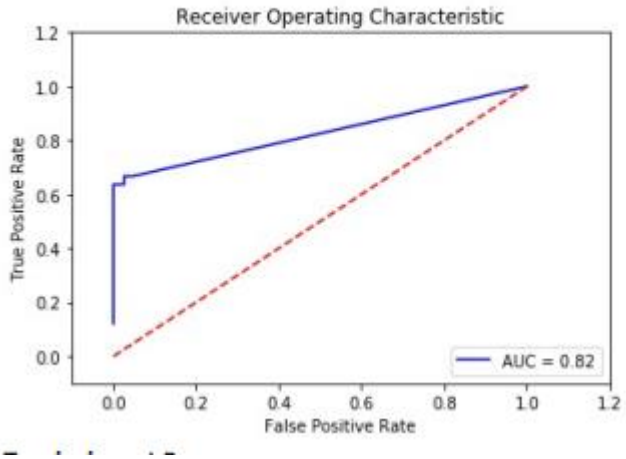
parameters = {"n_neighbors": [5,6,7], "algorithm": ['auto','brute'],"p": [2,3],"weights": ['uniform','distance']}

RESULT FOR ABPVE PARAMETERS:

Accuracy : 0.590308370044

	precision	recall	f1-score	support
0	0.24	0.73	0.36	33
3	1.00	0.52	0.68	99
4	0.00	0.00	0.00	6
5	0.96	0.61	0.75	108
6	0.00	0.00	0.00	3
7	0.78	0.54	0.64	39
8	0.17	0.33	0.22	9
9	0.00	0.00	0.00	3
11	0.00	0.00	0.00	6
12	1.00	0.56	0.72	48
16	0.37	0.94	0.53	54
17	0.00	0.00	0.00	3
18	1.00	0.17	0.29	18
22	0.70	0.87	0.78	156
23	0.14	0.25	0.18	12
26	1.00	0.08	0.14	39
27	0.83	0.38	0.53	39
29	0.00	0.00	0.00	6
avg / total	0.75	0.59	0.59	681

RoC curve



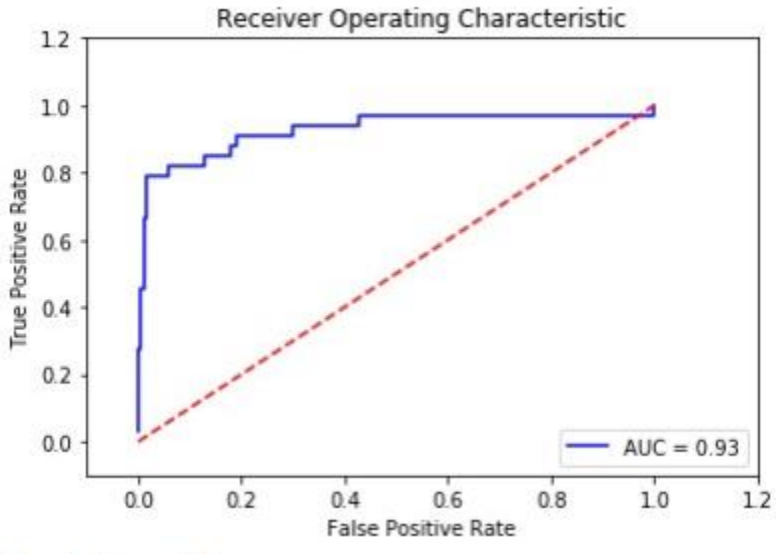
6.4 LR:

Logistic Regression is a type of discriminative classifier that directly learns the parameters of the model for P(Y|X). We create a model for P(Y|X) where the decision boundary is similar to that of a linear classification. It is used to describe the relationship between one dependent binary variable and one or more independent variables.

parameters = {'C':[1,5,10], 'max_iter' : [10,20,50], 'fit_intercept' : [True, False], 'penalty':["l1","l2"]}

Accuracy : 0.643171806167

	precision	recall	f1-score	support
0	0.45	0.45	0.45	44
3	0.83	0.76	0.79	132
4	0.00	0.00	0.00	8
5	0.70	0.78	0.74	144
6	0.00	0.00	0.00	4
7	0.69	0.85	0.76	52
8	0.00	0.00	0.00	12
9	0.00	0.00	0.00	4
11	0.00	0.00	0.00	8
12	0.62	0.50	0.55	64
16	0.57	0.94	0.71	72
17	0.00	0.00	0.00	4
18	0.33	0.17	0.22	24
22	0.60	0.79	0.68	208
23	0.00	0.00	0.00	16
26	0.80	0.31	0.44	52
27	0.60	0.46	0.52	52
29	0.00	0.00	0.00	8
avg / total	0.61	0.64	0.61	908



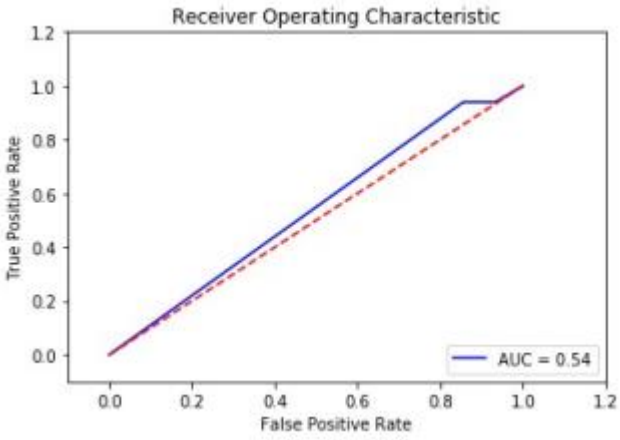
6.5 DT:

The default values for the parameters controlling the size of the trees (e.g. max_depth, min_samples_leaf, etc.) lead to fully grown and unpruned trees. So, to control the tree size, various values were considered for these attributes.

parameters = {'max_depth':[35,50,70,90], 'max_features' : [4,"sqrt","log2"], 'max_leaf_nodes' : [35, 50, 25, 10],'min_samples_split':[5,3,7]}

Accuracy : 0.242290748899

	precision	recall	f1-score	support
0	0.00	0.00	0.00	55
3	0.00	0.00	0.00	165
4	0.00	0.00	0.00	10
5	0.32	0.33	0.32	180
6	0.00	0.00	0.00	5
7	0.00	0.00	0.00	65
8	0.00	0.00	0.00	15
9	0.00	0.00	0.00	5
11	0.00	0.00	0.00	10
12	0.00	0.00	0.00	80
16	0.14	0.11	0.12	90
17	0.00	0.00	0.00	5
18	0.00	0.00	0.00	30
22	0.24	0.79	0.36	260
23	0.00	0.00	0.00	20
26	0.00	0.00	0.00	65
27	0.00	0.00	0.00	65
29	0.00	0.00	0.00	10
avg / total	0.12	0.24	0.14	1135



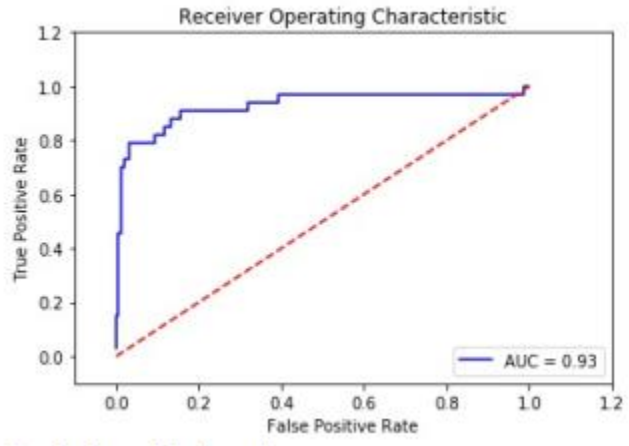
6.6 MLP:

The multilayer perceptron is a type of feed forward neural networks. It is a group of perceptrons joined together to achieve better and accurate results. It consists of atleast 3 layers of nodes, an input layer, hidden layer and an output layer. All the nodes except the input nodes consists of a summing function and an activation function. It uses backpropagation for training.

parameters = {'hidden_layer_sizes':[70,100,120], 'activation' : ["logistic", "tanh", "relu"], 'learning_rate' : ["constant", "invscaling", "adaptive"], 'max_iter':[750,1000]}

Accuracy : 0.629955947137

	precision	recall	f1-score	support
0	0.40	0.36	0.38	66
3	0.72	0.70	0.71	198
4	0.00	0.00	0.00	12
5	0.67	0.78	0.72	216
6	0.00	0.00	0.00	6
7	0.75	0.92	0.83	78
8	0.00	0.00	0.00	18
9	0.00	0.00	0.00	6
11	0.00	0.00	0.00	12
12	0.57	0.50	0.53	96
16	0.59	0.94	0.72	108
17	0.00	0.00	0.00	6
18	1.00	0.17	0.29	36
22	0.58	0.81	0.68	312
23	0.00	0.00	0.00	24
26	1.00	0.23	0.38	78
27	0.62	0.38	0.48	78
29	0.00	0.00	0.00	12
avg / total	0.61	0.63	0.59	1362



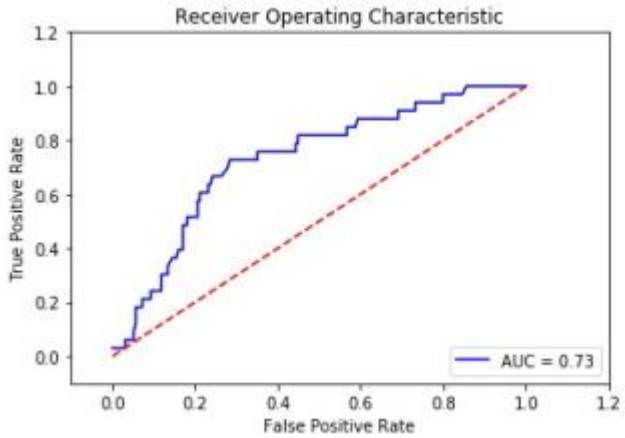
6.7 Adaboost:

Adaboost is short for Adaptive boosting. Boosting refers to combining a set of weak Classifiers to produce a strong classifier. It is type of ensemble method where we start with equal weights for all instances. Then we need to increase the weights of those instances where the hypothesis makes an error.This is repeated for multiple hypotheses and a final hypothesis is created using the weighted average of all the hypothesis.

parameters = {"n_estimators": [80], "learning_rate": [0.8],"random_state": [80],"algorithm": ['SAMME']}

Accuracy : 0.321585903084

	precision	recall	f1-score	support
0	0.00	0.00	0.00	77
3	0.22	0.39	0.28	231
4	0.00	0.00	0.00	14
5	0.65	0.42	0.51	252
6	0.00	0.00	0.00	7
7	0.00	0.00	0.00	91
8	0.00	0.00	0.00	21
9	0.00	0.00	0.00	7
11	0.00	0.00	0.00	14
12	0.00	0.00	0.00	112
16	0.00	0.00	0.00	126
17	0.00	0.00	0.00	7
18	0.00	0.00	0.00	42
22	0.32	0.87	0.46	364
23	0.00	0.00	0.00	28
26	0.00	0.00	0.00	91
27	0.00	0.00	0.00	91
29	0.00	0.00	0.00	14
avg / total	0.21	0.32	0.23	1589



7. CONCLUSION

We used various supervised and unsupervised machine learning techniques as part of this project. Then used HOG descriptor for feature extraction. We performed Principle Component Analysis to reduce the number of features that are highly correlated. We then trained and tested the dataset on the various supervised learning techniques. ROC curve gave us a graphical representation on the comparison of the various techniques. We also evaluated precision-recall-f1 score – support values for different models. Upon comparing the accuracy of all the models, we concluded that SVM gives the best result on our train and test dataset. The SVM has area under ROC curve of 95%. The training data set can be increased to improve the performance of the models.

8. REFERENCES

- H. Noh, A. Araujo, J. Sim, T. Weyand, B. Han, "Large-Scale Image Retrieval with Attentive Deep Local Features", Proc. ICCV'17
- http://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html
- http://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html
- http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html
- https://www.learnopencv.com/histogram-of-oriented-gradients/
- http://study.marearts.com/2014/04/example-source-code-of-extract-hog.html

-