Outdoor Terrain Classification

Anonymous Author(s)
Affiliation
Address
email

Abstract

In this paper we present a comparison of performances for six classifiers when applied to the problem of outdoor terrain classification. We chose Random Forest, K-Nearest Neighbor, Neural Networks, Linear SVM, Polynomial kernel SVM, Radial Basis Function SVM. We considered a basic form of the problem, where we formulate the problem as a binary classification task. The classes we consider are road and non-road. For feature vector we divide each image of 800x600 into patch of 20x20 and compute color histogram for the patch in all three color channels(namely R,G and B). We also present some results for parameter tuning, and error rates over a data set consisting of 35 thousand points. Based on the performances we conclude that classifiers capable of handling non linear data such as Neural Networks, Random Forest and KNN perform best and SVM family of classifiers is not able to cope well although performs better than chance.

1 Introduction

Outdoor terrain classification is one of the basic problem we need to solve to enable Autonomous navigation. A robot moving outdoors needs to have semantic knowledge of the terrain. Depth information alone might not provide all the details for efficient navigation. A tar road and a muddy patch have different friction coefficient, a human will naturally avoid the mud and stay on tar road. On the other had if robot has only 3d information than it would be unable to differentiate between tar and mud. This could potentially lead to robot getting stuck where it's unable to act. Using classification to assign meaning to patches will help planners plan better. The basic principle for classification was homogeneous visual characteristics, color for tar roads is very similar overall, but varies with road side objects like trees(green), mud(orange). Hence, we divide the image into patches of 20x20 size and compute a color histogram for all three channel, namely Red Green and Blue. Classifier are trained with these histograms which has similar signatures for patches belonging to a class. Although there is intra class variance withing color information, for e.g. most of the road patches although has dark grey color, lane strips and color saturation provide samples which are closer to white than they are closer to black. This kind of variance makes data highly non-linear, which dictates our choice of classifiers.

Research groups have been working on this problem for around a decade now, most of the papers focus on classifier which are capable of dealing with non-linear data like Random Forest(RF) [1] and Neural Network(NN) [2]. So, we choose these two classifiers which we assume should provide good performance. K-Nearest Neighbor(KNN) is also another classifier which works on the principle of homogeneity of data, but can perform if there are multiple clusters of data. We wanted to investigate it's performance on the data available to us. Another family of classifier we wanted to investigate are SVM classifiers. Considering the nature of data we expected that it's performance would not be as good as classifiers mentioned above, but does it do better than chance is what we wanted to know.

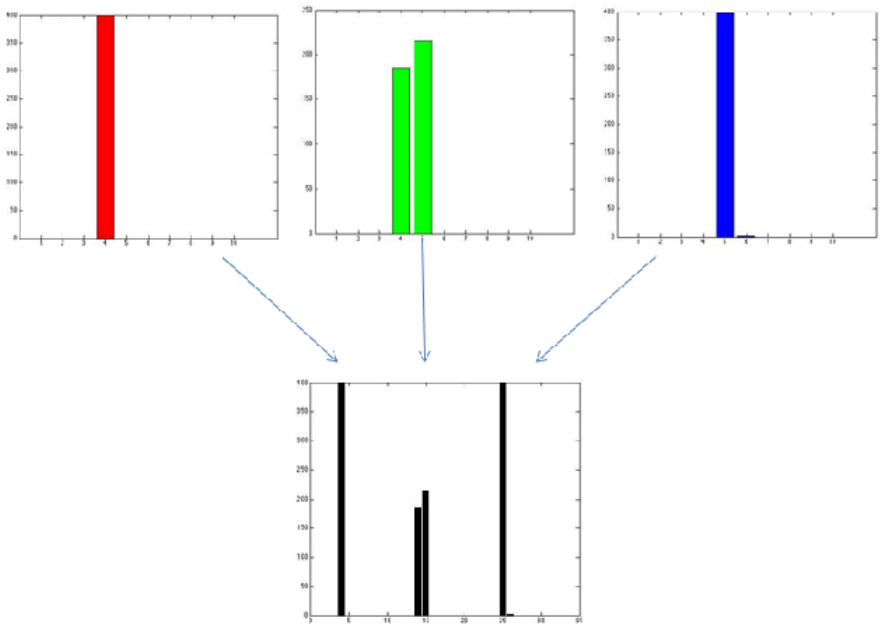

Figure 1: Feature Vector for a 20x20 patch

2 Data and Feature Description

Data Set: We have 100 data images (800x600) and corresponding label images. Each Data image is divided into 1200 patches of size 20x20. Total number of images available is 100. Total number of feature vectors derived from the data is 120,000. Initially data was distributed in the following way Holdout Data = 20,000 (DataPoints/6), Training Data = 50,000 (DataPoints*5/12), Testing Data = 50,000(DataPoints*5/12). Due to the large size of data, execution of LibSVM required a lot of time. Therefore the data size was reduced to the following Holdout Data = 10,000, Training/Testing Data 25,000.

Feature vector: Each Data image is divided into 1200 patches of size 20x20. Three histograms are created for each patch corresponding to three different color levels. Histogram is created with bin size of 10 resulting in a 1x10 vector for each color level. These three histograms are then concatenated to give one feature vector of size 1x30. E.g. of Histogram is shown in Figure 1.

3 Results

In this section we present and compare results for the six following classifiers: SVM-Linear, SVM-Poly, SVM-RBF, KNN, Random Forest, Neural Network.

3.1 Parameter tuning

While performing parameter tuning we considered a dataset of 10,000 samples and performed a 10-fold cross validation over it. Following subsections have detailed report for chosen classifiers.

3.1.1 SVM-Linear

The parameter we tune for Linear SVM is C, which is penalty for allowing points inside the margin. Due to high training time for svm linear the only values of C considered were: 0.01, 0.1, 1, 10. Figure 2 is graph showing error rates plotted against C.

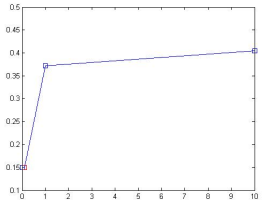

Figure 2: Error values of different C's for SVM Linear.

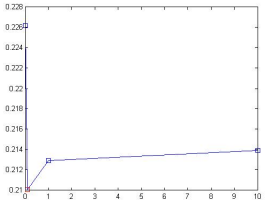

Figure 3: Error values of different C's for SVM Polynomial.

C	0.01	0.1	1	10
error	0.1497	0.1494	0.3724	0.4042

C with minimum error = 0.1.

3.1.2 SVM-Polynomial

The parameter we tune for plynomial kernel SVM is also C, we fix the order of the polynomial as 3. Due to high training time for svm-poly the only values of C considered were: 0.01, 0.1, 1, 10 Figure 3 is graph showing error rates plotted against C.

C	0.01	0.1	1	10
error	0.2262	0.2101	0.2129	0.2139

C with minimum error = 0.1.

3.1.3 SVM-RBF

The parameter we tune for plynomial kernel SVM is also C. SVM-RBF had relatively lower training time, hence the values considered for were: 0.01, 0.1, 1, 10, 20, 30 40, 50, 100 Figure 4 is graph showing error rates plotted against C.

C	0.01	0.1	1	10	20	30	40	50	100
error	0.3612	0.3439	0.301	0.2967	0.2968	0.2970	0.2969	0.2973	0.2974

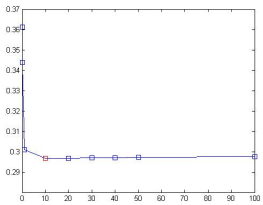

Figure 4: Error values of different C's for SVM RBF.

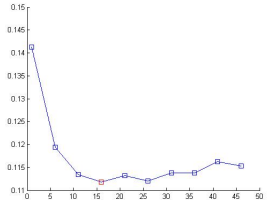

Figure 5: Error values of different K's for KNN.

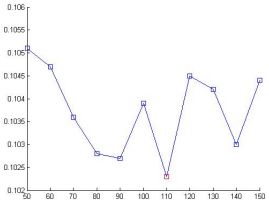

Figure 6: Error values for different number of trees for Random Forest.

C with minimum error = 10.

3.1.4 KNN

The parameter we tune for KNN is K, number of neighbour nodes to be considered. Values considered for K were 1 to 46 at interval of 5. Figure 5 is graph showing error rates plotted against K.

K	1	6	11	16	21	26	31	36	41	46
error	0.1413	0.1194	0.1134	0.1118	0.1132	0.112	0.1138	0.1138	0.1162	0.1153

K with minimum error = 16.

3.1.5 Random Forest

The parameter we tune for RF is number of trees(N) in the forest. Values considered for N were 50 to 150 at interval of 10. Figure 6 is graph showing error rates plotted against N.

N	50	60	70	80	90	100	110	120	130
error	0.1051	0.1047	0.1036	0.1028	0.1027	0.1039	0.1023	0.1045	0.1042

N with minimum error = 110.

3.1.6 Neural Network

The parameter we tune for Neural Network is number if nodes(n) in hidden layer. Values considered for N were 5 to 25 at interval of 5. We had planned to tune initial weights as well, but the matlab toolbox for neural network randomly initializes the weights and we were not able to find any way to supply our own weights.

Figure 7 is graph showing error rates plotted against n.

N	5	10	15	20	25
error	0.0922	0.898	0.0914	0.0851	0.0872

n with minimum error = 20

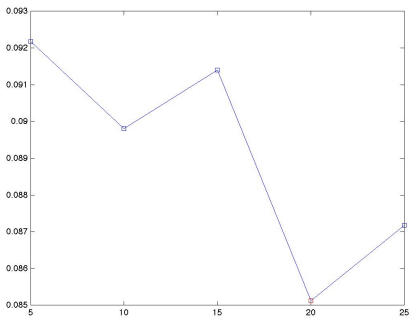

Figure 7: Error values for different number of trees for Neural Network.

3.2 Test Error

We now compare results for all the classifiers. These errors were computed on a data of size 25,000 samples. 5-Fold cross validation was performed over it to get an robust estimate of the error.

Classifier	Test Error
SVM-Linear	0.1495
SVM-Poly	0.3400
SVM-RBF	0.2856
KNN	0.1123
RF	0.1048
NN	0.0902

Conclusion: Performance of classifiers capable of dealing with non-linear data perform the best, Neural network performing the best with error rate of 0.09. SVM classifier perform bad, as we expected, although performance of linear SVM is unexpected. Convergence for Linear SVM was not achieved and the classifier returned the parameters estimated after maximum number of iterations, hence the error computed might not be a close approximation for true error. Although Neural Network showed better performance at every label and would be recommended at for this application.

References

[1] Chetan, J., K. Madhava Krishna, and C. V. Jawahar. "An adaptive outdoor terrain classification methodology using monocular camera." Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on. IEEE, 2010.

[2] Shinzato, P.Y.; Grassi, V.; Osorio, F.S.; Wolf, D.F., "Fast visual road recognition and horizon detection using multiple artificial neural networks," Intelligent Vehicles Symposium (IV), 2012 IEEE , vol., no., pp.1090,1095, 3-7 June 2012.