Machine Learning for Color Classification and Image Segmentation

Aneeq Zia
Electical Engineering Department, National University of Ireland, Maynooth
aneeq91@hotmail.com

***Abstract*— This paper addresses the issue of color classification with specific application to Robo-Cup Soccer League. The main Machine Learning methods used are Linear Discriminant Analysis and Support Vector Machines though some other techniques involving different combinations of RGB and the principal components are also presented. A detailed comparison table of different methods of Data classification and some segmented images are given towards the end.**

I. INTRODUCTION

Colour classification is an important and challenging problem for all systems involved in computer vision. In Robotic soccer, the most common technique used for color classification is the one using a Look-Up Table (LUT). A LUT defines the color for each pixel value; hence, in the case of 256 levels for each color, LUT would be a vector with 16 million elements. Such a big vector is always a problem to store in systems with low memory, much like a humanoid robot itself. Hence, in order to address this problem, we can potentially use machine learning to reduce the memory usage significantly and still achieve very accurate classification of colors. Though machine learning can take a lot of time initially depending on the size of data and the technique used, once the machine has 'learned' the characteristics of the data, classification can be done very efficiently The rest of the paper is organized as follows. Section 2 and 3 give some background about Linear Discriminant Analysis and Support Vector Machines, respectively. The implementation of the methods on Matlab is discussed in Section 4. Section 5 presents the results. Conclusion is made in Section 6.

II. LINEAR DISCRIMINANT ANALYSIS

Linear Discriminant Analysis (LDA) is a popular statistics tool used for data classification and dimensionality reduction. It helps to better understand the distribution of the feature data. Though the data worked with for this paper was three dimensional, a two dimensional example is illustrated to explain the concept behind LDA more simply. Refer to figure 1. Consider the two data sets, A and B. The first step in applying linear discriminant analysis is to compute the mean of each data set, μ_A and μ_B, and the mean of the entire data set, μ. The mean of the entire data set can be calculated using Equation 1.

$$\mu = p_A.\mu_A + p_B.\mu_B \quad (1)$$

Where p_A and p_B are the apriori probabilities of the different classes. LDA uses the between-class and within-class variance to formulate a decision boundary for separating the two classes. As labeled in figure 1, $S_{W(a/b)}$ represent the within-class variance of the two data sets and S_B represent the between-class variance. So for a single set, the within-class variance can be calculated using equation 2.

$$S_W = \sum_i p_i.(x_{A/B} - \mu_{A/B})(x_{A/B} - \mu_{A/B})^T \quad (2)$$

The optimizing criterion for LDA involves maximizing the ratio of between-class variance to within-class variance. The result obtained from this optimization technique then defines the axes of the transformed space. The transformations are found as the eigenvector matrix of the criteria mentioned before. As shown in figure 1, the two data sets are projected onto a single line using the transformation matrix. In general, for a problem containing n classes, we need n-1 separating lines. Once the transformation is done, the means of the transformed data sets are calculated. The Euclidean distance of the transformation of the test vector is calculated from the two means of the transformed data sets. Consequently, the test vector is classified as the set whose transformed mean is closer to the transformed value of the test vector.

III. SUPPORT VECTOR MACHINES

A. Multi-Class

The basic principal behind Support Vector Machines is to construct a single or a set of hyperplanes in a higher

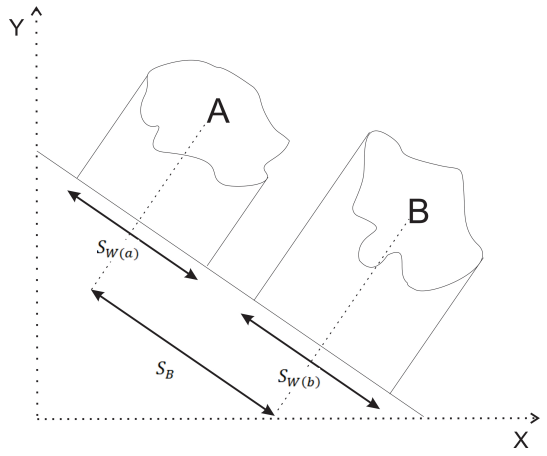

Fig. 1. Logic behind LDA

dimensional space (possible infinite), which can then be used for data classification. We label the training data $\{x_i,y_i\}$. For a linear SVM, we first find the class of separating hyperplanes subject to some constraints which results in equation 3.

$$y_i.((\mathbf{w}.x_i) + b) \tag{3}$$

where w is the weight vector and b is a threshold. In order to find the optimal hyperplane, the only task left is to solve a constrained optimization problem. After applying Lagrangian multipliers on the problem, we find the decision function for the hyperplane as given in equation 4.

$$f(x) \quad = \quad sign(\sum_{i=1}^{n} y_i.\gamma_i.(x.x_i) + b) \tag{4}$$

Where γ_i are the Lagrangian multipliers.On the other hand, non-linear SVM first involves the mapping of the input variable onto a higher dimensional dot product space and then finding the optimal hyperplane in the higher dimensional space to separate the different classes. The method for mapping the data onto a higher dimensional space is called the Kernel trick. There are various different kernel functions present that can be used for mapping a data set to a higher dimensional space; Radial Basis Function was used for this paper. Defining the Mercer Kernel as $k(x_i, y_j) = \phi(x_i).\phi(x_j)$, we just need to replace the dot product in equation 4 by the kernel to find the optimal separating hyperplane. Hence the general form of the optimal hyperplane becomes the one given in equation 5.

$$f(x) \quad = \quad sign(\sum_{i=1}^{n} y_i.\gamma_i.k(x.x_i) + b) \tag{5}$$

Figure 2 shows a three class problem with the separating boundaries drawn.

B. One-Class

The strategy of One-Class SVM is to map the data onto a feature space in order to separate the classes from the origin with maximum margin. In easy terms, One-Class SVM tries to fit, as tightly as possible, all the points in a single class inside an irregular sphere. This is illustrated in figure 3. All the test vectors within the irregular sphere are classified as belonging to that class where the points outside are labeled as unclassified. The decision function for a one-class SVM

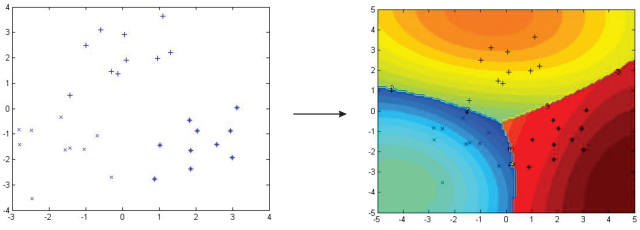

Fig. 2. Separation of data using Multi-Class SVM

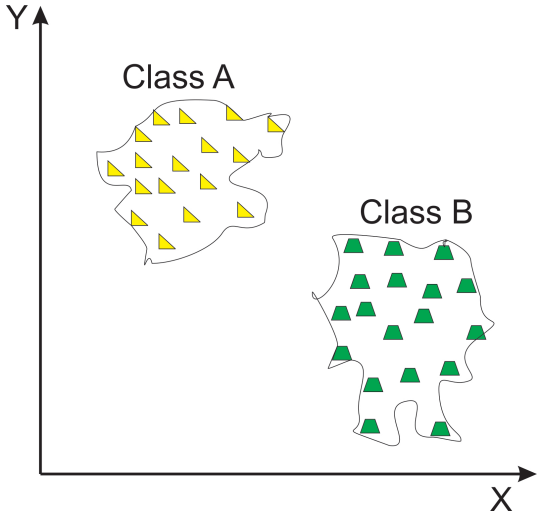

Fig. 3. Concept behind One-Class SVM

is quite similar to the one in equation 5 and can be found easily as given in equation 6.

$$f(x) \quad = \quad sign(\sum_{i=1} y_i.k(x.x_i) - b) \tag{6}$$

IV. IMPLEMENTATION

The first step of the research mainly included gathering a considerably large data set of different colors to work with. For this work, the color classification problem of Robo-Cup Soccer league was targeted. Hence, samples of yellow (goal post), blue (goal post), orange (ball), white (lines) and green (pitch) were taken. But as these are not the only colors present in the soccer arena, another class labeled as other was created which had samples of all the different un-useful colors present in the surroundings. Around 100k samples of different pixel values were taken. Once the sampling was done, in order to manipulate the data easily, a $n \times 4$ matrix was generated where n is the number of pixel samples (around 100k in this case). The first three columns of the matrix contained the RGB values of the pixel whereas the fourth column had the class label. The data was then divided into a training set and a test set.The different techniques for accurate color classification were then implemented on the data using Matlab.

The first technique used was using different combinations of RGB and see whether it is possible to separate the 'clouds' of different colors (as shown in figure 4).Every possible combination was tried and the data was then separated using LDA. Another similar technique of taking different number of principal components of the data before applying LDA was also used. Applying SVM was a bit more complex than LDA. SVM algorithm can only find the separating boundary of two classes at one time. Hence, a popular technique known as the 'one-versus-all' was used. For that, a total of 6 classifiers are 'learned'; each being able to classify the test vector as being a certain color or not being that color. So a test vector was passed through each of these

TABLE I

SAMPLE CONFUSION MATRIX

C	1	2	3	4	5	6
1	0.94	0	0	0	0	0.06
2	0	0.98	0	0	0	0.02
3	0	0	0.96	0	0	0.04
4	0	0	0	0.95	0	0.05
5	0	0.01	0	0	0.88	0.11
6	0	0	0	0	0.01	0.99

classifiers in order to classify it as belonging to any one class. But as the data may be noisy and any two classes might be overlapping, such a technique can have some ambiguous cases like more than one classifier claiming it to be belonging to its class. In order to address this problem, a variable was created which would calculate the number of classifiers which are successfully classifying the test vector. After that, different probabilities were assigned to different classes and the test vector was clsasified depending on those probabilities.

V. RESULTS

In order to measure the performance of the different machine learning methods, a confusion matrix was calculated for each method. A confusion matrix is a $n \times n$ matrix, where n is the number of different classes in the data. Each column of the matrix represents the instances in a predicted class, while each row represents the instances in an actual class. A benefit of using confusion matrix as a measure of performance is that it makes it easy to see if the system is confusing two classes i.e. mislabeling one class as some other quite frequently. An example of a confusion matrix is given in table 1. The elements in the diagonal of the confusion matrix represent the percentage of points in each class which were classified correctly. So the value 0.94 in the first element of the matrix states that the class 1 was correctly classified 94 percent of the times. In order to incorporate the values of the confusion matrix into a single value, we calculate the Misclassification rate (MCR) and the Normalised Misclassification rate (NMCR). Taking C to be the confusion matrix and L to be the number of points in each class, equations 7 and 8 can be used to calculate the misclassification rates.

$$MCR = \frac{1}{N}\sum_{j} C_{j,j}.L_j \quad (7)$$

$$NMCR = \frac{1}{6}\sum_{j} C_{j,j} \quad (8)$$

where N is the total number of data points and L_j represents the number of points in the j^{th} class. Values of the two misclassification rates were calculated for different methods which include using LDA with different combination of RGB and different number of principal components and with Multi/One-Class SVM. The results are displayed in table 2.

As evident from Table 2, SVM and LDA with RGB and three principal components worked quite well as compared to others. In order to visually see the results of the classification, some segmented images were formed by replacing each pixel of the image by the true color pixel values of the predicted color. Some of the segmented images along with their original look are given in Figure 5.

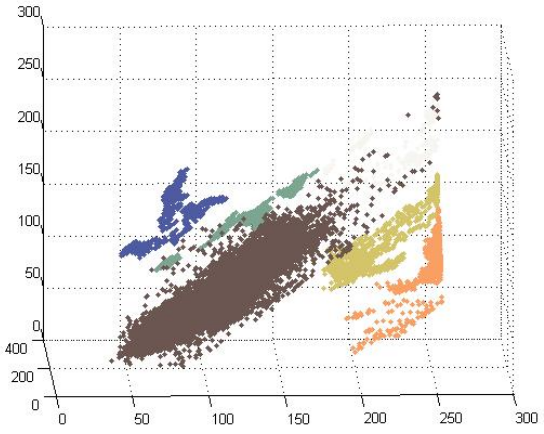

Fig. 4. 3D plot of different pixel values of the sample taken

TABLE II

MISCLASSIFICATION RATES FOR VARIOUS DIFFERENT TECHNIQUES IMPLEMENTED

Method	1	2	3	4
Gray Scale	42.14	67.15	41.94	67.06
RGB	6.01	3.62	5.98	3.62
R	70.73	52.79	70.50	52.97
G	42.37	67.29	42.27	67.22
B	87.27	92.76	87.11	92.35
RG	10.65	28.30	10.72	28.90
RB	6.37	5.13	6.39	6.12
GB	14.37	11.08	14.23	11.43
PCA1	42.82	67.49	42.84	67.49
PCA2	6.66	5.70	6.36	6.05
PCA3	6.01	3.60	5.98	3.92
Multi-Class SVM	4.35	4.35	4.42	4.42

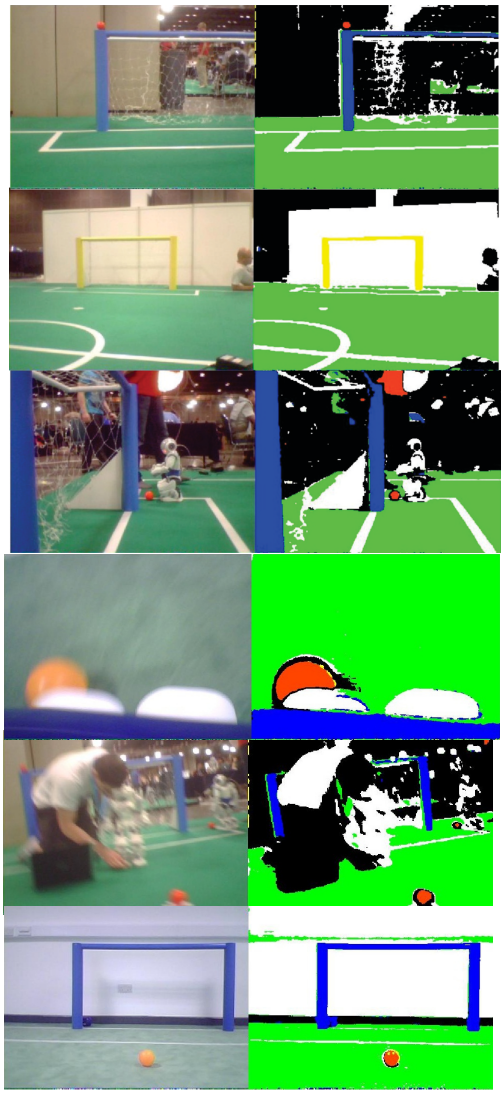

Fig. 5. Example of segmented images

VI. CONCLUSION

The methods presented in this paper indeed provide very good results for color classification. As evident from table 2, SVM and LDA with all three values of color pixels and all three principal components can separate the different colors with very less error. Due to some time constraints, more methods for data classifications could not be tested which could potentially provide even better results. It will be interesting to see how other methods like Neuro-scale and Relevance vector machines would work on this color classification problem.

REFERENCES

[1] S. Balakrishnama, and A. Ganapathiraju,Linear Discriminant Analysis - A Brief tutorial

[2] Yunqiang Chen, Xiang Zhou, and Thomas S. Huang, One-Class SVM for learning in image retrieval, IEEE Int'l Conf. on Image Processing 2001, Thessaloniki, Greece

[3] Michael J. Quinlan,Machine Learning on AIBO Robots,Unpublished doctoral dissertation, University of Newcastle, Australia.

[4] Michael J. Quinlan, Stephan K. Chalup and Richard H. Middleton, Techniques for Improving Vision and Locomotion on the Sony AIBO Robot.

[5] Michael J. Quinlan, Stephan K. Chalup and Richard H. Middleton, Application of SVMs for Colour Classification and Collision Detection with AIBO Robots.

[6] Bernhard Scholkopf, John C. Plattz,John Shawe-Taylory, and Alex J. Smolax, Robert C. Williamsonx,Estimating the Support of a High-Dimensional Distribution.

[7] David Lowe, and Michael E. Tipping, Novel Topographic Feature Extraction using RBF Networks,

[8] Tominaga, S, A color classification method for color images using a uniform color space