Color Detection Under Supervised Learning

Cyrus Anderson
University of Michigan
andersct@umich.edu

Jocelyn Bohr
University of Michigan
bjocelyn@umich.edu

Michael Lu
University of Michigan
lumike@umich.edu

Nghia Vo
University of Michigan
thnghia@umich.edu

Abstract

In this report, we present multiple supervised learning approaches to a common problem: color classification. From the simple k-nearest neighbors to machine learning SVM, there are a variety of approaches to this problem. Some solutions may perform better than others depending on the specific application and constraints. Here, our objective is to classify the colors of buoys on the surface of a pond during an autonomous robotic boat challenge under variations (e.g. time of day, weather, and lighting conditions). Our aim is to allow the autonomous robotic boat to use our classifier to make color classifications during the challenge. This report will compare results from basic non-learning algorithms such as majority-voting and average color, with more sophisticated learning algorithms such as Naive Bayes and multiclass SVM.

1   Introduction

Each summer, the autonomous robotic boat team UM:Autonomy at the University of Michigan competes in the Association for Unmanned Vehicle Systems International's RoboBoat competition held in Virginia Beach, Virginia against various other robotics labs and clubs. Since a pond serves as the site for the various challenges used to judge the robots, buoys act as the primary landmarks. However there are often a few other objects used as landmarks. Buoys often serve as the source of loop closures in the robot's long term path planning, allowing the robot to build an accurate internal map and minimize drift. Knowing additional features such as color can reduce contradictory information that produce incorrect mapping. Buoys also serve as boundaries or goals for certain challenges [2]. For example, some challenges involve determining the buoy color. Thus there is substantial need to accurately identify buoy color.

Currently, the UM:Autonomy robotic boat relies on color filters to determine color. The process involves specifying bounds in RGB values for each color of buoy, then classifying buoys according to their match with the closest color template. These filters are faulty because the tuning is highly susceptible to bias on current lighting conditions, which results in detections that work well for several hours before deteriorating in accuracy. This is especially apparent in Virginia where the sun will be high in the sky (casting glare on the buoys and water) for most of the day with the occasional cloud passing in front to cast a shadow on the entire competition ground. Rainy and overcast days present another set of lighting conditions. Even on days when the sun stays high in the sky, backlit green buoys can appear black *(fig.1 left)*, yellow can appear green, and washed out red may look orange. Additionally, white and black are difficult to filter since the filters for these colors generally have high false positive rates and end up including the very light or very dark patches of water produced with the noon sun's lighting *(fig.1 right)*.

In this paper, we examine the results of various supervised learning techniques applied to classifying buoy color, aim to create a procedure to accurately classify buoy colors in a variety of lighting conditions, and attempt to create a robust system that runs faster than the cameras' frame rate of 10 Hz while leaving sufficient resources for the boat's other processes.

Figure 1: Difficult buoys

2 Proposed Method

2.1 Preprocessing

The robotic boat has two Point Grey Flea 2, 1.3 MP cameras that publish images over the robot's Lightweight Communications and Marshalling (LCM) framework at 10 Hz. Logging utilities use LCM to capture and store various messages. Then the datasets are crafted from the different logs of the robot's runs. We label each boat photo by hand: marking the color and rectangle around each buoy within about 30 feet from the boat. From each rectangle, we extract the associated color histogram into a $256 \times 256 \times 256$ feature vector; each entry represents the frequency of the RGB color associated with the entry index. We then reduce this feature vector to a vector of size $64 \times 64 \times 64$, to make feature vectors less sparse and to decrease the classification time (since we are building this for a real time application).
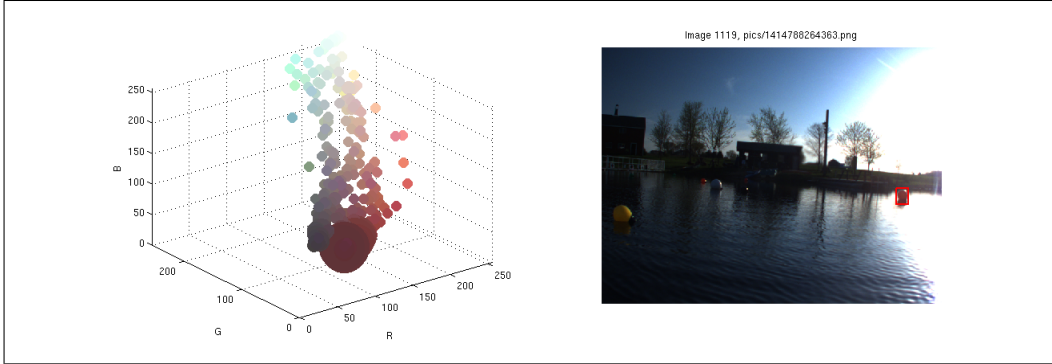

Figure 2: Histogram visualization with labelled red buoy

2.2 Classification

In this project, we run our training data with multiclass SVM and multinomial Naive Bayes, and are able to classify with high accuracy.

We began with SVM because it is a parametric algorithm; we do not need to store entire training data set to make predictions on the boat. It is also kernelized, which provides us with the flexibility to create arbitrarily complex boundaries. SVM prediction can be used to calculate confidence values for the decision by taking the distance from the separating hyperplane, which can be a useful measurement. SVM does not make any assumptions about the underlying

distribution of the data. SVM is a widely used classification algorithm, so it can appropriately serve as our baseline classification algorithm. We used LIBLINEAR for our SVM classification, which utilizes a one-vs-rest multi-class classification scheme. We achieved high accuracy with LIBLINEAR, so we decided not to use any type of SVM kernelization yet. Using a linear kernel with SVM is preferrable, as it is faster to train and test. Making decisions quickly is essential to this application, since we want to classify colors faster than the rate at which photos are captured (10 Hz).

We also used Naive Bayes because it has a simple implementation and it performs well even on small sets of training data. Naive Bayes is also parametric; we do not need to store the entire training data set to make decisions during the real time application. A parametric classifier is essential, since it is impractical to store all training data on the boat to make decisions in real time. Naive Bayes is also fast in training and testing, which is ideal for this application. The multinomial Naive Bayes classifier works well for classification with discrete features, such as color frequencies of pixels in a certain section of a photo, or word frequencies of a text document (e.g. the classic spam application). We used MATLAB's multinomial Naive Bayes classifier, which similarly performs with high accuracy.

3   Related Work

Color histograms have been in use for some time since their introduction [8] and are popular due to their invariance to translation and rotation in viewing perspective, and their robustness under changes in viewing angle and scale. The low computational complexity of color histograms makes them ideal for fast paced environments unsuitable for many more complex color detection techniques from computer vision. Furthermore, color histograms have been successfully applied to robotic soccer [3] and other realtime applications [4, 9]. A different approach is color constancy, which considers the color of the light source with that of the object to determine the object's color under a canonical light source from which a classification can be made. A standard approach in an unconstrained environment is to estimate parameters of the light source and then compute the object's illumination invariant descriptor under the canonical light source [1, 5]. Since early formulations of color constancy algorithms [6], more recent versions have reduced complexity to that of a FFT [7].

We want to compare results from the machine learning algorithms to results from non learning algorithms. This isn't a widespread problem people are trying to solve, so we could not find specific "state-of-the-art" algorithms. Thus we implemented two algorithms seemed reasonable for this application: majority vote and average vote.

3.1   Majority vote

Begin by iterating over the feature vector (described earlier in preprocessing), and choose the index with the highest count. Then compute the RGB vector by converting the index in the feature vector to the index in the original histogram, and store it as the most popular color. Compute the Euclidean distance between this RGB vector and the RGB vectors of our 6 colors, and classify the associated buoy as the color which yields the smallest distance.
The result was not as high as we expected, as many of the non-black buoys were classified as black because the shadow and water surface included in the picture popularized the count of black/near-black colors in the feature vector. In our 130-size test data, we average an accuracy of 21%.

3.2   Average vote

Begin by iterating over all the RGB values, then construct an RGB vector representing the total weighted sum of the RGB counts in the feature vector. Find the average RGB value by dividing by the total counts in the feature vector. Compute the Euclidean distance between this RGB vector and the RGB vectors of our 6 colors and classify the associated buoy as the color which yields the smallest distance.
This achieved higher accuracy than the majority vote algorithm. This can again be attributed to the

mixture of background in the picture boxes. In our 130-size test data, we average an accuracy of 48%. This accuracy is too low to be sufficiently useful.

4 Experimental Results

Using both linear SVM and multinomial Naive Bayes, we achieve high accuracy rates (compared to baseline methods discussed in related works). With a sufficient number of training examples, both methods achieve over 98% accuracy. Naive Bayes performs marginally better, and the advantage of Naive Bayes is emphasized when fewer training examples are used. Each average accuracy is calculated by selecting a random training set of size n, then selecting a random testing set of size 130 from the remaining data, and finally train and test classifiers accordingly. The average accuracy is taken over 10 classification results. Although these accuracies are high, we believe one contributing factor can be attributed to a fairly homogeneous data set. We do not have enough variation in our data set, so classification is easier. This problem, and how we intend to solve it, will be discussed in the future milestones section. Experimental results are shown below.

Table 1: Linear SVM Average Accuracy

EXAMPLES	ACCURACY
50	89.08
100	93.77
200	96.92
400	97.85
700	98.38

Table 2: Multinomial Naive Bayes Average Accuracy

EXAMPLES	ACCURACY
50	92.85
100	96.54
200	97.31
400	98.38
700	98.31

5 Future Milestones

One problem with our method so far is that most of the data is from one log, resulting in a very homogenous data set. The log was taken on a sunny day with few clouds, so the various algorithms have poor accuracy in classifying colors on cloudy days. Within one week from the due date of the progress report, we plan to label at least another 1000-5000 buoys, including many more blue and green buoys, and more buoys in varied lighting conditions. We hope that after running our algorithms on this data set with variation, we have a more realistic classification accuracy, which will perform better during the real application. We also want to try to add a feature which encodes the direction of lighting, to be completed within two weeks. Other ideas include capturing local binary patterns in the images, and using K-Means or Gaussian Mixture Models to cluster the pixels before creating each color histogram.

6 Conclusion

Our group's aim is to correctly detect the colors of buoys in pictures. In building a fast and robust algorithm to determine color under a variety of conditions, the robotic boat can use color as a trusted attribute to make higher level route planning decisions. We hope that this will enable the boat to perform well. So far, we have only trained and tested using multiclass SVM and multinomial Naive Bayes. As of now, the results have been accurate. We surmise that this accuracy results from our

overlapping data. The majority of our data overlaps because we obtained our data from a single recording log. Next, we aim to obtain a less homogenous data set by retrieving more data from other recording logs and performing error analysis with the new data. We are also aiming to add a feature that accounts the direction of lighting by the sun.

References

[1] Agarwal, V., Abidi, B.R., Koschan, A., & Abidi, M.A. (2006). An Overview of Color Constancy Algorithm, *Journal of Pattern Recognition Research*, 1(1), 42-54

[2] AUVSIfoundation (2014). 7th RoboBoat Competition - Final Rules. Retrieved from https://higherlogicdownload.s3.amazonaws.com/AUVSI/fb9a8da0-2ac8-42d1-a11e-d58c1e158347/UploadedFiles/RoboBoat$_2$014$_{final_rules.pdf}$

[3] Browning, B. & Govindaraju, D. (2003). Fast, Robust Techniques for Colored Object Detection in Variable Lighting Conditions, *United State Army, U.S.A.*

[4] Chakraborty, M. & Fuentes O. (2009). Real-time Image-Based Motion Detection Using Color and Structure, *International Conference on Image Analysis and Recognition*

[5] Forsyth, D.A. (1990). A Novel Algorithm for Colour Constancy, *International Journal of Computer Vision*, 5(1), 5-36

[6] Land E.H. & McCain J.J. (1971). Lightness and Retinex Theory. *JOSA*, 61(1), 1-11

[7] Morel, J., Petro, A.B., & Sbert, C. (2009). Fast Implementation of Color Constancy Algorithms, *Proc. SPIE 7241*

[8] Swain, M. J. & Ballard, D. H. (1991). Color Indexing, *International Journal of Computer Vision*, 7(1), 11-32

[9] Zhang H. & Zhao D. (2004). Spatial Histogram Features for Face Detection in Color Images, *PCM'04 Proceedings of the 5th Pacific Rim conference on Advances in Multimedia Information Processing* 1(1), 377-384