Gaussian Naive Bayes Model for Color Classification

1st Marcus Schaller
Department of Electrical and Computer Engineering (ECE 276A)
University of California San Diego
La Jolla, U.S.A
mschalle@ucsd.edu

***Abstract*—Identifying and classifying objects is an important function of many autonomous systems. In order to do so, a variety of techniques are used to extract features from objects. One common technique is the Gaussian Naive Bayes technique. This paper will discuss using the Gaussian Naive Bayes model on extracted Red Green Blue (RGB) values to identify colors. Further it will discuss how this model can be applied to pictures to identify blue recycling bins. Finally it will discuss the results of using this model on a variety of images.**

I. Introduction

Color is used to distinguish a variety of objects; therefore, it is extremely important for autonomous systems to be able to use color to aid in identifying objects. In driving, traffic light color is used to signify when a vehicle should stop, slow down, or go. Not being able to identify such colors can lead to a perilous system. Additionally, as autonomous technology advances, jobs normally done by humans are now becoming more and more automated. Such jobs often require autonomous computer vision technology to visually select certain objects and the use of color identification may help to facilitate this selection. It is not too difficult to foresee a future in which garbage trucks are fully automated and are able to locate the bins without any human input. Volvo is already testing autonomous garbage collection trucks. A human operator rides on the truck, but doesn't need to drive it as the truck follows a pre-programmed path. The truck uses cameras and sensors to maneuver itself up to the trash bin to easily allow the human operator to empty the bins content into the truck. This allows for a much quicker process as the human doesn't have to be inside the truck to move it to the next bin. [1] In this model, color can be used to identify the next pick location. Gaussian Naive Bayes is a very common technique that can be used to identify color and is what was used to identify bins in this report.

II. Problem Formulation

Determining the color of a pixel can be done many different ways. The technique I used was the Gaussian Naive Bayes method. The Gaussian Naive Bayes method tries to find the probability of each pixel. Gaussian Naive Bayes will determine the probability of each class and take the maximum value when attempting to classify pixel colors.

$$y = \text{argmax}_y \, P(y) \prod_{i=1}^{n} P\left(x_i \mid y\right) \tag{1}$$

The above equation for classifying color can be broken down into simpler steps that can be translated into code. Firstly, for the problem to be formulated the naive assumption that each feature when conditioned are independent of each other must be made. Expressed in equation form this concept is shown as follows:

$$P(A \cap B) = P(A)P(B) \tag{2}$$

In the case of colors we assign X to the RGB value of the desired pixel and y to the classification of that pixel.

$$P\left(y \mid x_R, x_G, x_B\right) \tag{3}$$

After applying Bayes Theorem this can be rewritten as:

$$\frac{P\left(x_R \mid y\right) P\left(x_G \mid y\right) P\left(x_B \mid y\right) P(y)}{P\left(x_R\right) P\left(x_G\right) P\left(x_B\right)} \tag{4}$$

Since the denominator is constant we can remove that term. Given the pixel value, the classifier will calculate the probability that the pixel belongs to a given color class y.

$$i \in R, G, B \tag{5}$$

$$y = \text{argmax}_y \, P(y) \prod_{i=1}^{n} P\left(x_i \mid y\right) \tag{6}$$

To make it easier to calculate the probability of x belonging to classification y the Gaussian distribution is used. First the mean and standard deviation of each possible classification must be calculated from the dataset. In the case of part 1 of the project, y could be red, green, or blue. In the case of the red data-set the mean of the red value is calculated as follows:

$$\mu = \frac{\sum y_R = Red}{\sum y = Red} \tag{7}$$

The standard deviation of the red value is calculated as follows (I represents the indicator function):

$$\sqrt{\frac{\sum (y_{Ri} - y_R) * I(y = red)}{I(y = red)}} \tag{8}$$

Identify applicable funding agency here. If none, delete this.

To calculate the probability that a pixel is red the mean and standard deviation of the red, green, and blue values must be calculated for all values inside the red data set. Once each mean and standard deviation is found the probability can be calculated as follows, where n is the total values in the data set: [2] [3]

$$P(y) = \frac{1}{n} * I(y = red) \tag{9}$$

$$i \in R, G, B \tag{10}$$

$$P\left(x_i \mid y\right) = P(y) \prod \frac{1}{\sqrt{2\pi\sigma_{y_i}^2}} \exp\left(-\frac{\left(x_i - \mu_{y_i}\right)^2}{2\sigma_{y_i}^2}\right) \tag{11}$$

In the case of pixel classification once the probability is calculated for red, green, and blue the probabilities are compared and the pixel is assigned R, G, or B based on the max value. To detect the bin the probability is used to create a mask and filter out any pixels that are not above a certain probability.

III. Technical Approach

A. Pixel Classification

Utilizing the steps taken in the previous section one can calculate the probability that a pixel belongs to a given dataset. During the pixel classification part of the project the red, green, and blue datasets were used to gather nine different means for the R, G, and B value of each dataset as well as nine different standard deviations. From there the probability that a given RGB pixel value belonged to each dataset was calculated and assigned a 1,2,3 if it had a higher probability of belonging to red, green, or blue, respectively. A least squares linear classifier was also used to compare results. However, since the linear classifier is not allowed it will not be discussed further.

B. Recycling Bin Color Mask

The initial step for the bin classification software was to create a Gaussian Naive Bayes model for each color to be test. For the bin detection 9 different datasets were collected. The datasets are as follows; blue, dark blue, light blue, grey, white, black, green, yellow, and orange. In total this equated to over 8 million RGB values. Using these datasets, a Gaussian Naive Bayes model was created and nine probability values were assigned to each pixel of a given image. These probabilities were assembled into matrices that represented the shape of the original image. The values were then normalized between 0 and 1. A basic matrix filters was then used to create a Boolean mask. This involved starting with the blue, dark blue, and light blue values and setting a probability threshold so that any pixel that was greater than the threshold would be set to true on the mask. This resulted in a mask that contained too many true values as many of the background pixels were included in the mask. An example of the mask at this stage can be seen in figure 2. The other datasets were then used to determine which pixels were more likely to belong to the other color classes and were set to false if they were above a certain threshold. Pixels less than .09 in the blue data set were also assigned

Fig. 1. Original image.

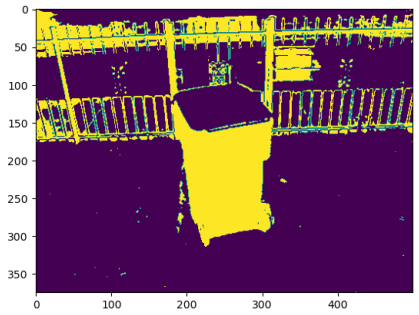

Fig. 2. Mask with blue pixels above threshold.

false This assists in filtering out many of the unwanted pixels. Figure 2 and figure 3 illustrate that this technique can be used to filter out the fence in the background.

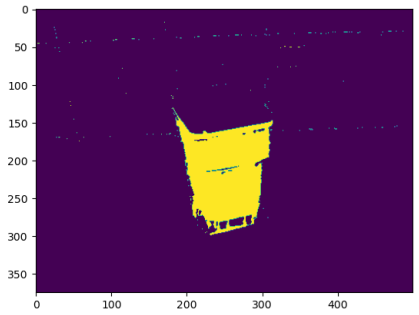

Fig. 3. Mask with blue pixels above threshold.

C. Image Morphology

The skimage toolbox was then used to further improve the image mask. Firstly, any small objects that were smaller than 0.03 percent of the total image size were removed. The process caused the bin to have some skeleton features as seen in figure 3. Therefore the areas surrounded by pixels were closed to create a solid mask. Other very slight adjustments were included as some special cases worked better with a slight

opening added after the closing. Figure 4 presents the mask after this step. For most images, this resulted in filtering out

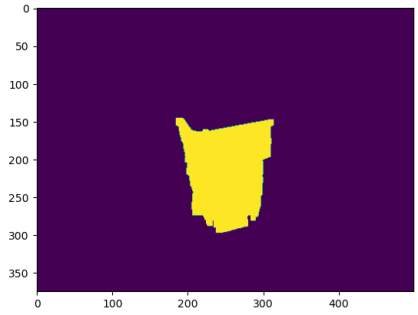

Fig. 4. Mask after initial morphology.

all background objects and the image would be ready for a bounding box. However, for bins that were set side by side an aggressive erosion was needed to prevent any pixels from touching between the bins. The mask was then dilated back so it would resemble the size of the original recycling bin. This can be seen in figure 5 and 6. At this point it was possible to begin filtering based on mask shape.

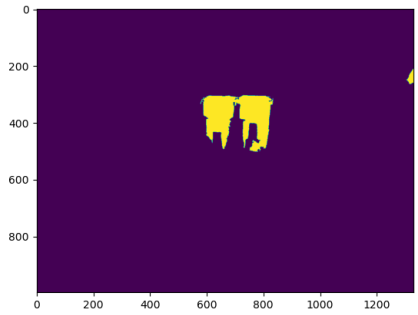

Fig. 5. Mask after initial morphology.

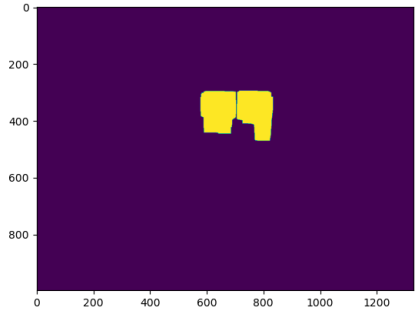

Fig. 6. Mask after erosion and dilation.

D. Region Props and Shape Filtering

After the previous filters have been applied, often the resulting mask free of any non-bin objects. However, in images that included blue objects such as vehicles or the sky additional filtering was needed to eliminate these elements. Regionprops was utilized to extract features of the remaining objects. The height and width of each object was calculated and any object that had a height less than 80 percent of the width were filtered out. 80 percent was selected because any images that included a bin that was tilted and being rolled would be wider than it is tall. This threshold appeared to work well for all of the training and validation images. An additional filter was also used to remove any small objects that remained. Figure 7 and 8 show validation figure 70 and displays the filter removing the blue pool from the mask. A filter was then used to eliminate

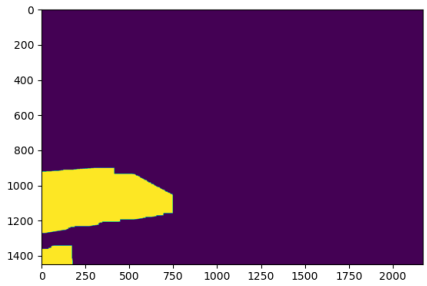

Fig. 7. Mask before height and width filter

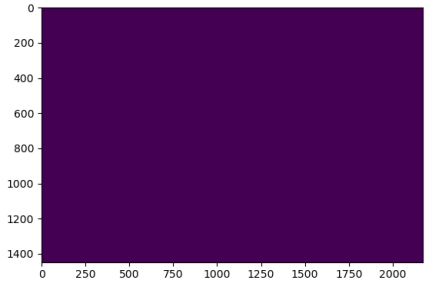

Fig. 8. Mask after height and width filter

any shapes that might look like a bin but were vastly smaller than any other bins. This filter took the item on the mask that covered the most area and any item smaller than 25 percent of that item were filtered out. This removed any small rectangular patches of light or any grey sidewalk that had passed the color filters. An additional filter was used to eliminate any bin shapes that show up as a result of the sky. An example of this is training image 45. In this image the trees segment the sky to create a recycling bin shape. This approach records the top of the very bottom shape. It then filters out any object for which

the bottom pixel is higher than the top of the bottom shape plus 10 percent of the total height of the image. This worked well on all of the training images and can be seen in figures 9-11. The final filter used was to filter out apparent bin shaped

Fig. 9. Image with sky bounded by trees

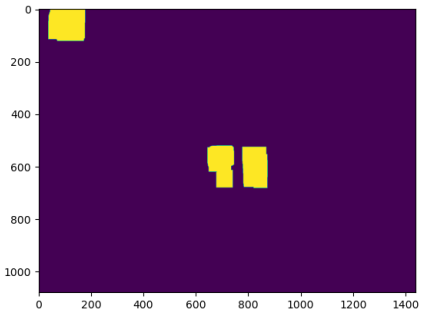

Fig. 10. Mask before filter

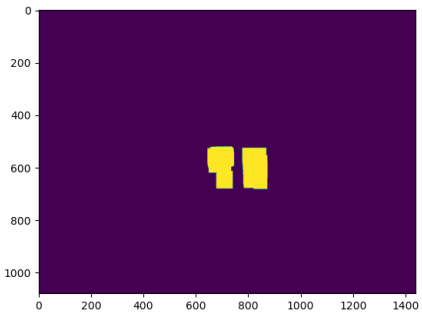

Fig. 11. Mask after filter

objects in the sky even if there was no bin below to compare it to. Objects that had a bottom pixel that were contained in the top quartile of the image were filtered out of the mask. Once this was completed regionprops was used to extract the bounding box coordinates of the final objects remaining in the mask and overlay the bounding box onto the original image.

IV. Results

A. Pixel Classification

The Gaussian Naive Bayes classifier proved to be successful in correctly classifying the validation pixels. In total the red pixels were 100 percent accurate, the green pixels were 97.06 percent accurate, and the blue pixels were 97.59 percent accurate. Since blue and green are so similar it is reasonable to expect some error for these colors. The following tables display the mean and standard deviations that were used in creating this model.

TABLE I
Red Dataset

Parameter	Red	Green	Blue
Mean	0.7525	0.3480	0.3489
Standard Deviation	0.1925	0.2489	0.2490

TABLE II
Green Dataset

Parameter	Red	Green	Blue
Mean	0.3506	0.7355	0.3294
Standard Deviation	0.2360	0.1865	0.2366

TABLE III
Blue Dataset

Parameter	Red	Green	Blue
Mean	0.3473	0.3311	0.7352
Standard Deviation	0.2335	0.2383	0.1890

B. Recycling Bin Identifier

The recycling bin identifier was mostly successful. The model was able to successfully apply a bonding box to 100 percent of the validation pictures and 73.3 percent of the test pictures. I focused on getting the model to work 100 percent on the validation photos and this contributed to unresolved issues in the training photos. I noticed that all of the recycling bins in the validation photos were dark blue or in some sort of shading. As a result, my model ended up being more successful at identifying recycling bins of a darker shade of blue than a lighter shade. Even after adding a light blue dataset the model was unable to successfully notice every light colored bin in the training photos. This is seen in figure 12 and 13. Given more time I would modify my model to better identify these bins by further tuning my color filters. The model does identify bins with a blue background very well though as seen in figure 14 and 15. Overall the model could correctly identify nearly all bins that were not a light blue color. Table 4 displays all of the mean and standard deviations used in my model. Table 5 displays the bounding box coordinates for each of the validation photos.

TABLE IV
RECYCLING BIN DATASETS

Parameter	Red	Green	Blue
Dataset	Blue		
Mean	0.1687	0.2590	0.5041
Standard Deviation	0.1450	0.1791	0.1896
Dataset	Light Blue		
Mean	0.3385	0.4913	0.7091
Standard Deviation	0.1685	0.1801	0.1615
Dataset	Dark Blue		
Mean	0.1345	0.1717	0.4039
Standard Deviation	0.1015	0.1003	0.1088
Dataset	Grey		
Mean	0.2982	0.3484	0.4152
Standard Deviation	0.1265	0.1276	0.1336
Dataset	White		
Mean	0.6942	0.7422	0.8097
Standard Deviation	0.1500	0.1206	0.0938
Dataset	Black		
Mean	0.1087	0.1453	0.1869
Standard Deviation	0.0707	0.0888	0.1123
Dataset	Green		
Mean	0.4179	0.5801	0.4934
Standard Deviation	0.1720	0.2051	0.1908
Dataset	Yellow		
Mean	0.8245	0.6264	0.0321
Standard Deviation	0.1346	0.2143	0.1256
Dataset	Orange		
Mean	0.9566	0.0663	0.1281
Standard Deviation	0.0883	0.0935	0.0661

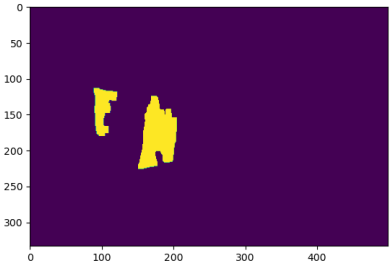

Fig. 12. Light Color Bin Mask

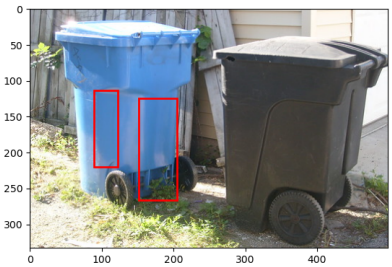

Fig. 13. Light Color Bin Bounding

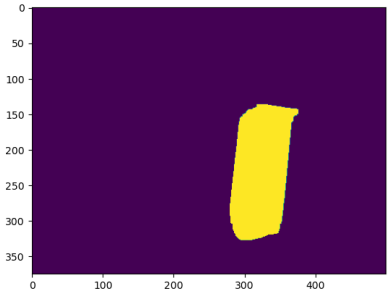

Fig. 14. Light Color Bin Bounding

Fig. 15. Light Color Bin Bounding

TABLE V
RECYCLING BIN POSITIONS

Validation Number	X1	Y1	X2	Y2
61	184	146	311	294
62	25	347	132	497
63	173	94	273	220
64	349	106	464	272
65	812	413	927	580
66	*	*	*	*
67 Bin Right	705	298	833	473
67 Bin Left	577	299	703	448
68	*	*	*	*
69	*	*	*	*
70	*	*	*	*

ACKNOWLEDGMENT

Thank you Professor Atanasov as well as Mo and Arash. Overall I found this project challenging but very rewarding. I appreciate the effort you put into making this project feasible and I am sorry you had to answer so many questions about uploading to gradescope. Now it is time for me to get some sleep.

REFERENCES

[1] P. Sawers, "Volvo's testing an autonomous garbage collection truck," VentureBeat, 18-May-2017. [Online]. Available: https://venturebeat.com/2017/05/17/volvos-testing-an-autonomous-garbage-collection-truck/. [Accessed: 31-Jan-2021].

[2] N. Kumar, "Naive Bayes Classifiers," GeeksforGeeks, 15-May-2020. [Online]. Available: https://www.geeksforgeeks.org/naive-bayes-classifiers/. [Accessed: 31-Jan-2021].

[3] N. Atansov, "Gaussian Discriminant Analysis," in ECE 276a.