Color Classification and Recycling Bin Detection

Zhexu Li
Department of Electrical Computer Engineering
University of California, San Diego
San Diego, California, United States
zhl411@ucsd.edu

***Abstract*—In this project I implemented and trained a Gaussian Naive Bayes model from pixel data to distinguish among red, green, and blue pixels. Then I utilized the same probabilistic color model to recognize recycling-bin blue color and segment unseen images into blue regions. Given the blue regions, the algorithm is able to automatically detect blue recycling bins and draw a bounding box around each one.**

***Index Terms*—classification, segmentation, detection.**

I. Introduction

Object detection is one of the fundamental problems in computer vision. An accurate, fast, and reliable object detection algorithm has great potential in many industries including robotics, security, and transportation. Despite the current research focus has shifted to deep learning based approaches because of the rapid advance in neural network architectures, traditional probabilistic color model based object detection algorithms are still valuable because they require less computational resources than deep learning models to train, not to mention understanding the intuitions behind these models will provide a solid foundation for learning more advanced models. In this project, I first implement a Gaussian Naive Bayes model to classify red, green and blue pixels. Then I implement an image segmentation algorithm using the model parameters trained on a large hand-labeled color dataset consist of 5 colors. And combining the segmentation algorithm with shape statistics of blue recycling bins, I create an algorithm to automatically detect blue recycling bins in images and draw a bounding box around each one. While the purpose of this project is to learn the intuitions behind common probabilistic color models and their applications in object detection, the resulting algorithm could potentially be used in many scenarios including automatic recycling bin collection, and obstacle avoidance.

II. Problem Formulation

A. Pixel Classification

Given a matrix of pixels $X \in \Re^{n\times3}$, we are interested in classifying every pixel $x \in \Re^{1\times3}$ into a color category $y_i \in \{1, 2, 3\}$, where 1 = red, 2 = green, 3 = blue, and return a vector of predicted color labels y $\in \Re^{n\times1}$.

Using MLE estimations trained by Gaussian Naive Bayes, the problem becomes

$$\operatorname*{argmax}_{\theta,w} p(y, X|w, \theta) \tag{1}$$

And for the training process we have:

$$\theta_k^{MLE} = \frac{1}{n}\sum_{i=1}^{n} 1\{y_i = k\} \tag{2}$$

$$\mu_{kl}^{MLE} = \frac{\sum_{i=1}^{n} x_{il}1\{y_i = k\}}{\sum_{i=1}^{n} 1\{y_i = k\}} \tag{3}$$

$$\sigma_{kl}^{MLE} = \sqrt{\frac{\sum_{i=1}^{n}(x_{il} - \mu_{kl})^2 1\{y_i = k\}}{\sum_{i=1}^{n} 1\{k = y_i\}}} \tag{4}$$

And for the later testing process, given input vector x we evaluate the optimal predicted label y^* by:

$$y^* = \operatorname*{argmax}_{y} log\theta_y^{MLE} + \sum_{l=1}^{d} log\phi(x_l; \mu_{yl}^{MLE}(\sigma_{yl}^{MLE})^2) \tag{5}$$

B. Recycling Bin Detection

Train a GNB model using new dataset with 5 categories (recycle bin blue, green, gray, black, brown). Then for every input image $X \in \Re^{n\times3}$ we predict the optimal predicted labels y^* for every pixel $x \in \Re^{1\times3}$ and segment the image based on the predicted label (1 for recycle bin blue, 0 otherwise). With the image segmented, one can detect blue recycling bins using shape statistics of the recycling bins after processing the image, and draw bounding boxes around those objects.

III. Technical Approach

I first implemented a Gaussian Naive Bayes classifier based on the formulas mentioned above. The implemented algorithm, naive_train(X, y) intakes a n x 3 matrix of pixels X and a n x 1 matrix of labels y and returns MLE estimators w which is a dictionary of dictionaries consisted of the parameters mentioned above. Then I implemented a prediction algorithm, classify(x) which intakes a 1 x 3 pixel vector and returns the optimal predicted label based on the MLE parameters w and the formula above. These two algorithms allow us to finish the pixel classification part of this project.

Then, for the recycle bin detection part, I first create training dataset by segmenting the 60 training images using the provided ROIPOLY function. All training images were converted into HSV color space because brightness is an important information for our task. I decided to segment the image into 5 colors: recycle bin blue, green, gray, black, and brown. I chose these 5 colors because recycle bin blue is the color we are interested in, grass is green, ground is gray, shadow is black, and trees are usually brown. Once I collected the labeled color vectors, I reused the GNB training algorithm naive_train(X, y) and received the dictionary of training parameters w. Then I implemented the segment_image(img) algorithm, which intakes an image, convert the image to HSV color space, and for every pixel in the image, it use the GNB testing algorithm mentioned above with the training parameters w to classify the pixel into one of the 5 color categories, and return the segmented image once every pixel has been classified.

With the image already segmented, I implemented the get_bounding_boxes(img) algorithm. It intakes a segmented image, convert every pixel to 1 if it's classified to be recycle bin blue, and 0 otherwise, then perform morphological operations to process the image. I chose to perform 2 erosion, 3 dilation, and 1 opening using disk 7, because this process seemed to remove a lot of noise, and reconnect the disconnected areas. Then the processed image is labeled by the skimage.measure.label(img) and grouped into different regions by skimage.measure.regionprops(labeled_image). The algorithm then select regions with its area >= 5% of the total image area to filter out small noise regions, and based on the shape characteristics of blue recycle bins the algorithm further select regions with height (y) > 1.2 * width (x) and height (y) < 2 * width (x) to filter out likely non recycling bin regions. These shape characteristics were selected because a typical blue recycling bin should be large enough (> 5% of the image) and it should be a standing rectangular (y > x) but should not be way too thin (y < 2 * x). And once a region which meets all the above criteria was selected, the algorithm draw a rectangular bounding box around the region, which indicates a blue recycle bin is detected.

IV. Results

A. Pixel Classification

The accuracy of the GNB color model was very consistent. It achieved 98.1% accuracy on the training set, 98.2% accuracy on the validation set, and 98% accuracy on the testing set on gradescope. And both the accuracy and the parameters returned by my model matches the accuracy and parameters returned by sklearn's GNB, which suggests my implementation should be correct. The resulting parameters trained on the training set are:

Mean: {Red: 0.36599891716296695, Green: 0.3245804006497022, Blue: 0.3094206821873308}

Covariance:
Red: {0: 0.037059271217808214, 1: 0.06196869331835154, 2: 0.06202254921782519}
Green: {0: 0.05573462963129557, 1: 0.03478592727001181, 2: 0.0560218800651483}
Blue: {0: 0.05453762216925839, 1: 0.056833309793085915, 2: 0.03574060702528315}

I also implemented a Gaussian Discriminant Analysis model and trained it on the training set, but the validation accuracy was lower than the GNB model so I chose to stick with the original GNB model.

B. Recycling Bin Detection

The resulting parameters trained on the labeled training dataset are:

mean: {Recycle Bin Blue: 0.3705880436445994, Green: 0.13779946255470984, Gray: 0.07166102005223834, Black: 0.2855520834468152, Brown: 0.1343993903016372}

Covariance: 1: {0: 218.117429712996, 1: 5290.097551779682, 2: 1510.6213744944737}, 2: {0: 2065.503776927488, 1: 2177.2776860167844, 2: 3108.309013377322}, 3: {0: 481.99921375546154, 1: 3229.17580599338, 2: 1889.4591365411018}, 4: {0: 1887.854466282301, 1: 419.2863446390129, 2: 1453.8721296699368}, 5: {0: 697.1487520579971, 1: 1184.3587701300614, 2: 746.614418613}

Using the trained parameters above, the image segmentation algorithm seemed to perform well at segmenting blue objects, but it usually classify other blue-ish pixels into recycle-bin blues and create some false positives, which suggests the algorithm might have a high recall but low precision, and additional image processing is required before detection:

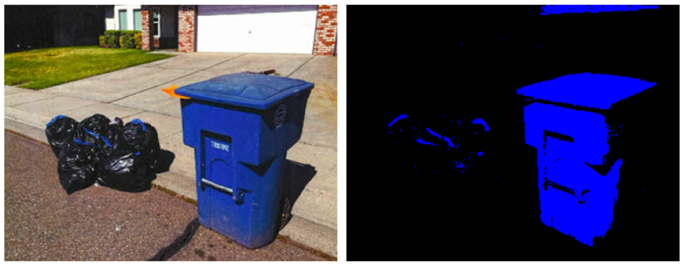

Fig. 1. A validation (0063.jpg) example with a little false positives.

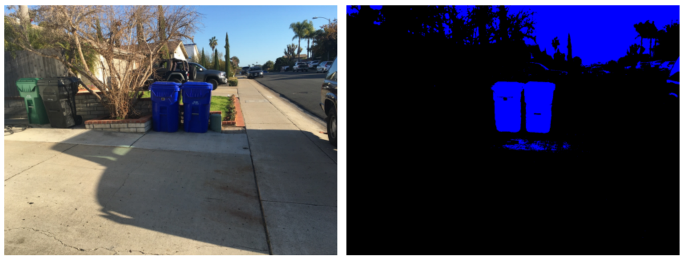

Fig. 2. A validation example (0067.jpg) with a lot of false positives.

After processing the images using the morphological operations mentioned above, the detector was able to draw bounding boxes around the objects which are identified to be blue recycle bins. The results are displayed in the appendix. I then evaluated the performance of my algorithm using recall and specificity. The detection algorithm achieved about 70% recall and 70% specificity on the training images, 50% recall and 100% specificity on the validation set, and 8.5 / 10

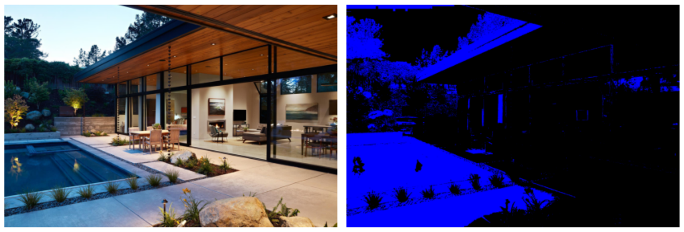

Fig. 3. A validation example (0070.jpg) with a lot of false positives.

on the testing images on gradescope. The inconsistency was likely contributed by the heavy image processing, for example, compare fig 2 and fig 10 we can see the morphological process merged the two detected blue bins together into one square object, which doesn't meet the shape requirement so the algorithm returned no blue bin was found. But I chose to heavily process the images because that way it largely reduce the amount of false positives, which explains why my algorithm has relatively high specificity. On the other hand, fig 7 and 8 shows the limitation of our GNB color model, the classifier identified a lot of background noises right next to the bin to be recycling bin blue, which basically merged the bin into the background, and the detector was not able to identify the bin given a messy masked image, which resulted in false negatives. This behavior happened less frequently in the training set, instead there were a lot more false positives, which led to a more balanced recall and specificity on the training set. And because I didn't want to overfit the validation set, I decided not to modify the processing and detection algorithm. The model achieved 8.5 / 10 on the testing set on gradescope, with case 3 received 0.5 / 1, and case 7 received 0 / 1.
In conclusion, despite the limitations we discussed above, GNB color model still has its advantage in simplicity, flexibility, and training speed, compared to other complex models. And it's clear that the performance of a GNB based detector is largely determined by the quality of training dataset and shape characteristics defined, and the image processing techniques applied. Which suggests that, with enough tweaking and better training set, the GNB based detection algorithm could achieve much better performance.

V. Appendix

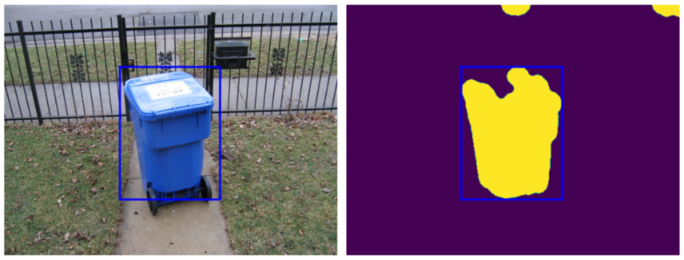

Fig. 4. 0061.jpg, estimated coordinates: [[172, 92, 323, 291]]

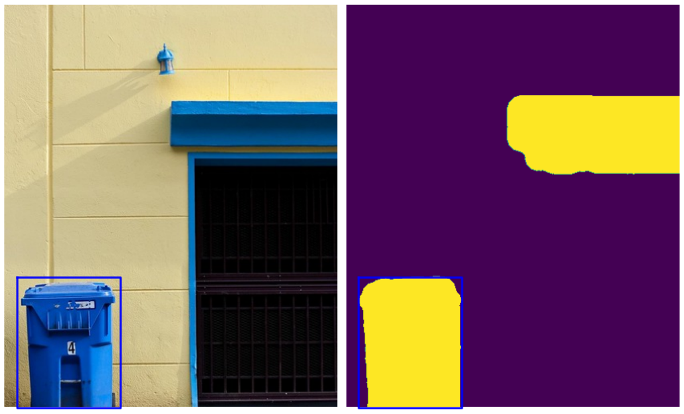

Fig. 5. 0062.jpg, estimated coordinates: [[15, 337, 143, 499]]

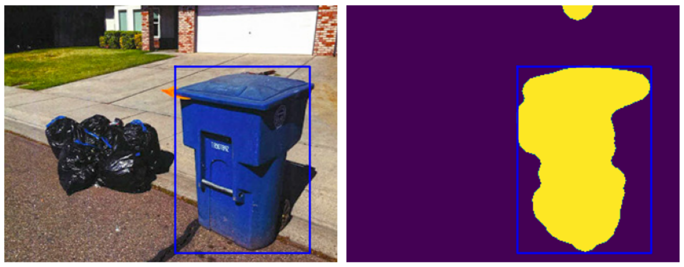

Fig. 6. 0063.jpg, estimated coordinates: [[166, 59, 297, 240]]

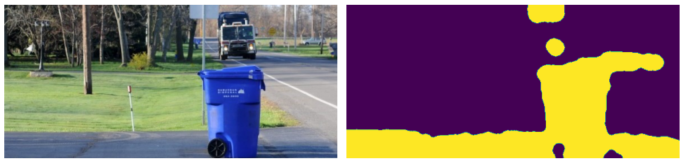

Fig. 7. 0064.jpg, estimated coordinates: []

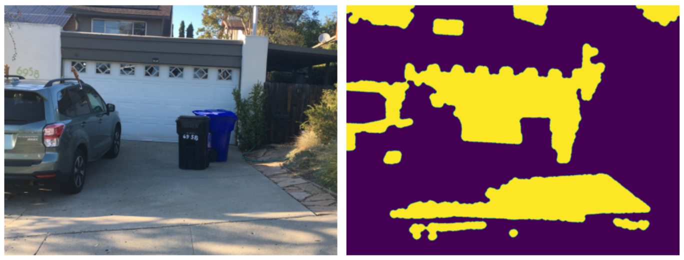

Fig. 8. 0065.jpg, estimated coordinates: []

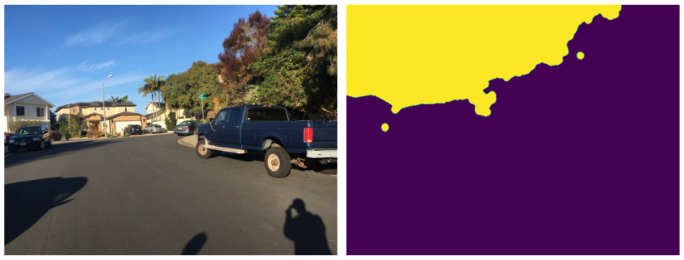

Fig. 9. 0066.jpg, estimated coordinates: []

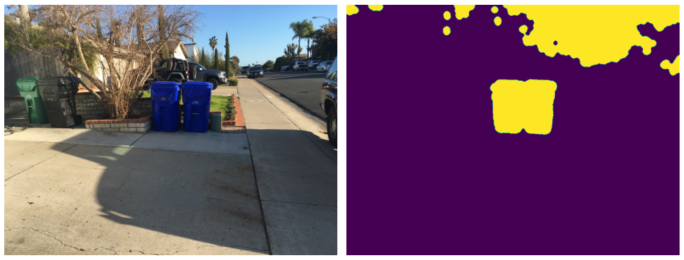

Fig. 10. 0067.jpg, estimated coordinates: []

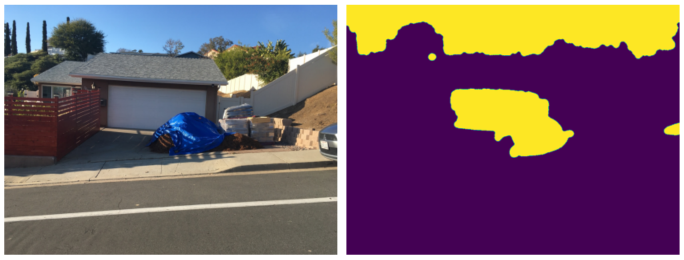

Fig. 11. 0068.jpg, estimated coordinates: []

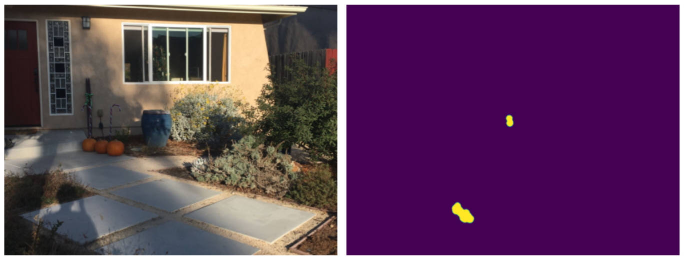

Fig. 12. 0069.jpg, estimated coordinates: []

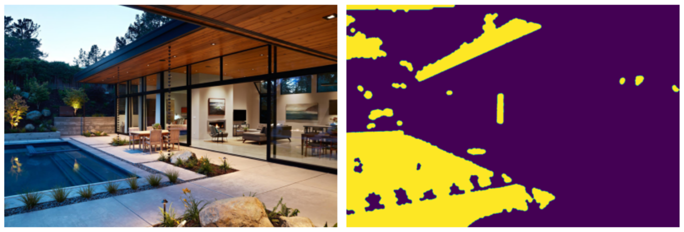

Fig. 13. 0070.jpg, estimated coordinates: []