ECE276 Project 1: Color Classification and Recycling Bin Detection

Shivani Bhakta
Dept. of Electrical and Computer Engineering
University of California, San Diego
s9bhakta@ucsd.edu

I. Introduction

In this paper we consider a probabilistic color model to perform color detection on a given pixel and address a problem of recognize blue recycling bin in a given image. The goal is to use the color classification model to segment images based on its color, and detect the recycling bin using its shape and the same classifier in a given image. This problem is important to address and can have applications in automated trash-collection system, navigation and object detection for robots indoor as well as outdoors (i.e, self-driving cars or house-cleaning bots). This is an important problem for the robots to accurately map its surrounding and for it to be able to position itself and move safely.

In this paper we used a supervised learning approach with a discriminative classification model using softmax regression. It is useful for mapping the given the pixel of an image into a categorical probability mass function, which is then used for classifying pixels into different color based on their probability. To identify and detect the recycling bin, we first have to segment training image into recycling bin blue surface and other regions that are blue but not recycling bin blue. Using that data we use the classification model from first part and then using the general shape of a recycling bin we detect the bins. This model can be used to classify recycling bin from any new input images.

II. Problem Formulation

A. Color Classification

In given training and validation datasets of images, each image is of size 28×28. Thus, let $\Gamma \in \mathbb{R}^{m \times d}$, where $m = 28 \times 28$ represents each pixel of dimension d. (For our RGB dataset, we will have d = 3). Our dataset D= $\{x_i, y_i\}_{i=1}^{n}$ where $x_i \in \mathbb{R}^d$, $y \in \mathbb{R}^n$ and $y \in \{1, 2, 3\}^n = \{Red, Green, Blue\}$, consists of top left pixel from all n images in the training set. The goal of this problem to classifier any given pixel into one of the RGB colors. To train the model we have used softmax regression and thus have to use one-hot encoding for the labels y. That is, we convert $y \in \{1, 2, 3\}^n$ to $y \in \{(1, 0, 0), (0, 1, 0), (0, 0, 1)\}$. We will use our $X \in \mathbb{R}^{n \times d}$ to train and then used the model to classifies images of the similar dimensions from the validation set.

B. Recycling Bin Detection

In this problem the goal is to detect blue recycling bins in a given image. First, we hand-label regions in images from the training set into different colors labels (In our case, Recycling-bin blue, brown, green, and other blue) and use the classification model in last section to segment each image into different colors. Once we have the segmented images, using the shape statistics and other features of the recycling bins, we draw a bonding box for around each bin.

III. Technical Approach

A. Color Classification

Our Approach to color classification is to use supervised learning method with a discriminative probabilistic classification model to calculate the $p(y|X; w)$ with parameters w to approximate the unknown labels. To optimize the w using the dataset $D = (X, y)$, we use the Maximum Likelihood Estimation(MLE), which maximizes the likelihood of the data D given the parameter w (Equation 1 2). Basically, we want will be able to get a categorical probability distribution of our labels, and determine the color based on the highest probability. We will use Softmax Function to convert or map our input data into a categorical probability mass function, and choose the label with the highest probability. This means softmax function will give us our $\hat{y}_{predict} = \{P(red|X, w), P(green|X, w), P(blue|X, w)\}$. Note, all $P(y|X, w)$ add up to 1 and we will choose the label which has the highest probability. Softmax Function is a generalization of Logistic Regression (uses sigmoid function for binary classification) to do classification with K-classes (Equation 3).

Maximum Likelihood Estimation (MLE):

$$\textbf{Training:} w^* \in \arg\max_{w} p(y|X, w) \tag{1}$$

$$\textbf{Testing:}\ y* \in \arg\max_{y} p(y|x_*, w^*) \tag{2}$$

The Softmax Function,

$$s(z) = [\frac{e^{z_1}}{\sum_j (e_{z_j})} \cdots \frac{e^{z_k}}{\sum_j (e_{z_j})}] = \frac{e^z}{\mathbf{1}^T e^z} \in \mathbb{R}^K \tag{3}$$

is used to get the predicted probability labels. Here the input $z = x_i^T w$ and k are the number of classes. However, due to some of the pixel values, we were getting a overflow value. Therefore, we can used the

$$s(z) = s(z - c\mathbf{1}) \tag{4}$$

where $c \in \mathbb{R}$ is a constant, specifically, I used $c = max_i(z_i)$. Notice that, $\frac{e^{x_j}}{\sum_i(e^{x_i})} = \frac{e^{x_j - c}}{\sum_i(e^{x_i - c})} = \frac{e^{-c}}{e^{-c}} \frac{e_{x_j}}{\sum_i(e^{x_i})}$.
Therefore, we can use the constant **c** to avoid overflow in the softmax exponential and does not get too big.

$$\begin{aligned} p(y|X, W) &= \prod_{i=1}^{n} e_{y_i}^T s(Wx_i) \\ &= \prod_{i=1}^{n} e_{y_i}^T \frac{exp(Wx_i)}{1^T exp(Wx_i)} \end{aligned} \tag{5}$$

To optimize the parameters W using MLE, we compute the gradient of the data log-likehood (Equation 5), we get

$$\bigtriangledown_w [logp(y|X, W)] = \sum_{i=1}^{n} (e_{y_i} - s(Wx_i)) x_i^T \in \mathbb{R}^{K \times d} \tag{6}$$

$$W_{MLE}^{(t+1)} = W_{MLE}^{(t)} + \alpha (\sum_{i=1}^{n} (e_{y_i} - s(W_{MLE}^{(t)} x_i)) x_i^T) \tag{7}$$

Now, Putting everything together, lets us say we want to train using our data $X \in \mathbb{R}^{n \times d}$ and find the weights $W \in \mathbb{R}Kxd$. We will have to consider a bias that might play a role. Then our equation becomes $z = XW + b$. Note that, due to the bias, we will be adding a columns of 1 at the end of X (so now $X \in \mathbb{R}^{n \times d+1}$) and W dimension would therefore change to $W \in \mathbb{R}^{d+1 \times K}$. Now, we pass z into our softmax equation(4). Hence, we at the end we have $\hat{Y} = s(z)$.

Overall Algorithm to compute the weights goes like this:

$w \leftarrow 0$
for num of epochs **do**
    $W^{t+1} = Wt + \alpha (\sum_{i=1}^{n} (e_{y_i} - s(W_{MLE}^{(t)} x_i)) x_i^T)$
end for

Here α is learning rate (came out to be 0.001 for part 1). Moreover, We ran classifier for different number of epochs and it took atleast 50 epochs to reach the 100 percent accuracy for part 1, color classifier.

B. Recycling Bin Detection

For Recycling bin detection, first we needed to label appropriate regions in the training images with its color labels. We choose to have four classes: Recycling bin Blue, Brown, Green and None-Recycling bin blue. Here we have two classes with blue so that we can differentiate between things that are recycling bin blue color and things that are not. This labeling was done using the RoiPoly function provided. After the RoiPoly "cropping", mask of the original image was generated, which was saved to for each label in .npy file, so that during training, this mask images can be used to get the original pixel values from the original image for each individual color categories of the four classes we choose. Important thing to consider here is that, these mask label were hand-generated, therefore, there are lot of possibilities of error and hence our model probably won't be as accurate as our color classification test results. After, the label was done, the next step was to put together all the pixels for each of the four classes to train the classifier. This was done by using the mask values to get the pixels from the original image. Once the data was formatted in similar format as previous part (i.e. $X \in \mathbb{R}^{n \times d}$) and $y \in \{1, 2, 3, 4\}$ because of fours classes. After, the weights w were generated to classify different color objects in the test images using the same classifier.

During the training, several things were changed from the color classification, one of them is alpha (learning rate), it was reduced significantly. Moreover, we looked at how the model perform with the different color spaces such as RGB, YUV and HSV. This matters due the lighting conditions in some of the image sample as mentions in the assignment. In this case, YUV and HSV performs better than RGB. Most importantly, number of epochs at which the model was trained also varied for this problem, it was a lot, and it is probably due to the increased about of data points used to train the model.

After the weights were generated, we needed to detect the recycling bins in the test images and draw bounding box around it. This is done by finding the cluster of points that were classified with the same class and see if the region may be in the same shape or form of a recycling bin. This was done using the regionprops and label function from the Scikit-image package. If there are any cluster that looks like in the shape of a recycling bin, then we draw a bounding box around it.

IV. Results

A. Color Classification

Parameters Used by the Color Classification: For the training in this part, one pixel from each training image from all three colors was used. Thus the total number of data points used to train was fairly small (n = 3694). Thus, the learning rate was also low $alpha = 0.001$, and number of epochs it took to reach an accuracy of 100% was $epochs_num = 50$. There was not parameters to change and the classifier was perfect after all of the above. The weights that were calculated are:

5.112031480488034241e+00	-2.556549723279263109e+00	-2.555481757208770244e+00
-2.466116777604632659e+00	4.935383930183192547e+00	-2.469267152578558999e+00
-2.417497163219497036e+00	-2.477561574932541699e+00	4.895058738152038735e+00
-8.038082902588955525e-02	4.575377677882047633e-02	3.462705224707009893e-02

Fig. 1. Weights for Color Classification

B. Recycling Bin Detection

For this part the training samples were generated from the images provided. Each pixel from all the mask images were used as a data points and therefore, we had Millions of data points for each of the four color classes. Since we had such large training dataset our learning rate alpha came out to be $alpha = 0.0000001$ and $epochs_num = 100$. Many different combination of these two parameters were tried, but we concluded that these were the sufficient numbers and the model was not going to significantly improve with any more change in the parameters.

I have added the test images where, we were successfully able to detect the recycling bins. The accuracy from the test code is also included below.

Fig. 2. Validation image 0065.jpg

Fig. 3. Validation image 0064.jpg

Fig. 4. Validation image 0070.jpg

Fig. 5. Validation image 0066.jpg

Fig. 6. Validation image 0067.jpg

Fig. 7. Validation image 0063.jpg

Fig. 8. Validation image 0062.jpg

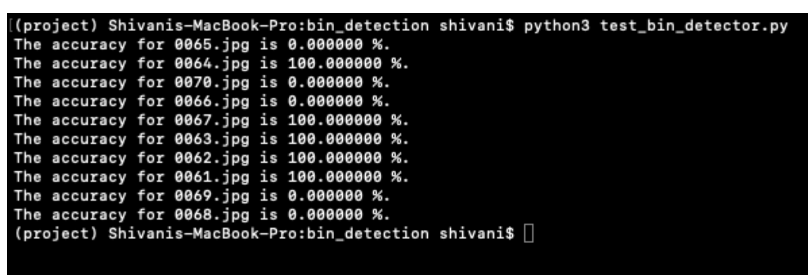
Fig. 9. Validation image 0061.jpg

Fig. 10. Validation image 0069.jpg

Fig. 11. Validation image 0068.jpg

REFERENCES

[1] Timothy D. Barfoot State Estimation for Robotics http://www.cs.cmu.edu/ tom/mlbook/NBayesLogReg.pdf
[2] Aston Zhang and Zachary C. Lipton and Mu Li and Alexander J. Smola *Dive into Deep Learning*. https://d2l.ai 2020
[3] I discussed the project with Keshav Rungta and Saikiran Komatineni



```
(project) Shivanis-MacBook-Pro:bin_detection shivani$ python3 test_bin_detector.py
The accuracy for 0065.jpg is 0.000000 %.
The accuracy for 0064.jpg is 100.000000 %.
The accuracy for 0070.jpg is 0.000000 %.
The accuracy for 0066.jpg is 0.000000 %.
The accuracy for 0067.jpg is 100.000000 %.
The accuracy for 0063.jpg is 100.000000 %.
The accuracy for 0062.jpg is 100.000000 %.
The accuracy for 0061.jpg is 100.000000 %.
The accuracy for 0069.jpg is 0.000000 %.
The accuracy for 0068.jpg is 0.000000 %.
(project) Shivanis-MacBook-Pro:bin_detection shivani$
```

Fig. 12. The accuracy for the validation images presented above.