Blue-bin detector via logistic regression

Hisu-Wen Yen A59010599 hsyen@ucsd.edu

***Abstract*—This report develops a blue-bin detector via logistic regression on color classification and morphological properties of detected blue regions.**

I. Introduction

Object detection technique is crucial in autonomous vehicles because it provides information of surrounding area for system, allowing system make decision based on these information. If we can develop blue-bin detector successfully, then we are able to build detector for any object which can be applied to autonomous vehicles. In this report contains two parts. First part is color classification for blue detection. Second part is algorithm on morphological properties to identify bin-shape.

II. Problem Formulation

A. Color classifications

Normally, color can be represented by RGB value, thus we can define any color x as vector with three dimension (red value, blue value, green value), and in computer we use unsigned 8 bits to store RGB value as convention.

$$\text{Color } x \in R^3 \tag{1}$$

$$R \in [0,255] \tag{2}$$

In reality, we treat any different x with similar color as a same color; e.g., [240, 10, 10] and [241, 10, 10] both are defined as red. In this section, we only use three colors (red, green, blue) to define x, and we unify each value (divided by 255). We set y as random variables for defined colors.

$$y = \begin{cases} 1 \; as \; red \\ 2 \; as \; green \\ 3 \; as \; blue \end{cases} \tag{3}$$

We assume there exist discriminative models $p(y|x,\omega)$ for $y = \{1,2,3\}$, then we pick highest one as corresponding color. Models will be like

$$\begin{cases} p(y = 1|x,\omega_1) \\ p(y = 2|x,\omega_2) \\ p(y = 3|x,\omega_3) \end{cases} \tag{4}$$

Then, collecting all given x as X, and we assume $Data = (X, y)$ are iid, then the joint likelihood will be

$$\begin{cases} p(y = 1|X,\omega_1) \\ p(y = 2|X,\omega_2) \\ p(y = 3|X,\omega_3) \end{cases} \tag{5}$$

Then, the problem becomes finding correct parameters ω that maximize likelihood functions.

$$\omega^* = \; arg \max_{\omega} p(y|X,\omega) \tag{6}$$

B. Bin-shape identification

After blue classification, we will get binary image (only 0 and 1), we use 1 to present blue regions and 0 to represent non-blue regions. Same as following matrix:

$$image = \{1,0\}^{m*n}$$

$$m = height, n = width \tag{7}$$

We can use Raster Scan to label connected components in a binary image, call labeled region A_i and we get k regions. We know recycle-bin has standard format, therefore, we can use function f_i developed by bin-shape to create a set W. Then,

$$A_i = \begin{cases} recycle - bin, if \; A_i \in W \\ non \; recycle - bin, if \; A_i \notin W \end{cases} \tag{8}$$

$$W = \{w | w \; satisfies \; f_i\} \tag{9}$$

The problem becomes finding suitable f_i.

III. Thectical Approches

A. Color classifications

We use logistic regression to solve the problem; first, we need to turn our label $y = \{1,2,3\}$ into One-Hot. One-Hot encoding ensure that distance between the class is same. Now label becomes

$$y = \begin{cases} [1,0,0] \; as \; red \\ [0,1,0] \; as \; green \\ [0,0,1] \; as \; blue \end{cases} \tag{10}$$

Then, we set $p(y|x,\omega)$ has form like

$$p(y|x,\omega) = \begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} + b \tag{11}$$

$$x = \begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix} = [r, g, b]$$

ω has two parts w_i (weights) and b_i (bias).

Combine three $p(y|x,\omega)$, it can be written as

$$[x_1 \quad x_2 \quad x_3]\begin{bmatrix} w_{r1} & w_{g1} & w_{b1} \\ w_{r2} & w_{g2} & w_{b2} \\ w_{r3} & w_{g3} & w_{b3} \end{bmatrix} + [b_r \quad b_g \quad b_b] = [z_1 \quad z_2 \quad z_3] \tag{12}$$

z_1 is value calculated for red, and so on. However, z_i isn't probability, we can use softmax function to turn z_i into probability.

$$s(z) := \left[\frac{\exp(z_1)}{\sum_j z_j} \cdots \frac{\exp(z_k)}{\sum_j z_j}\right] \tag{13}$$

$$p_i = \frac{\exp(z_i)}{\sum_j z_j} \tag{14}$$

With One-Hot label, we can use cross-entropy to define our loss function, and minimize it to obtain ω^*.

$$E(\omega) = -\ln p(Y|X,\omega)$$
$$= -\sum_{n=1}^{N}\sum_{k=1}^{K} y_k^{(n)} \ln p_k^{(n)} \tag{15}$$

Here is why cross-entropy will work: in multi classes problem; for a sample t (One-Hot encoded), we assume its probability distribution with parameter μ_i which represents the probability when $t = class\ i$ is

$$p(t) = p(t|\mu) = \prod_{k=1}^{K} \mu_k^{t_k} \tag{16}$$

Combine all collected data, the data likelihood becomes

$$p(T|\mu) = \prod_{n=1}^{N}\prod_{k=1}^{K} \mu_k^{t_k^{(n)}} \tag{17}$$

From MLE, we can get correct μ by maximize (17), and we can take log on (17) since log is monotonical function.

$$\ln(T|\mu) = \sum_{n=1}^{N}\sum_{k=1}^{K} t_k^{(n)} \ln \mu_k \tag{18}$$

Maximize (18) equals minimize negative (18)

$$\min -\sum_{n=1}^{N}\sum_{k=1}^{K} t_k^{(n)} \ln \mu_k \tag{19}$$

(19) is same as (15), therefore, minimize the cross-entropy error function is based on Maximum Likelihood Estimation. To minimize (15), we can perform gradient descent, then we need to find its derivative with respect to $w\ and\ b$. We can apply chain rule to find its derivative.

$$\frac{\partial z_i}{\partial w_i} = \frac{\partial}{\partial w_i} x^T w_i + b_i = x^T \tag{20}$$

$$\frac{\partial z_i}{\partial b_i} = \frac{\partial}{\partial b_i} x^T w + b_i = 1 \tag{21}$$

$$\frac{\partial p_i}{\partial z_i} = \frac{\partial}{\partial z_i}\frac{\exp(z_i)}{\sum_j z_j} = \frac{\exp(z_i)(\sum_j z_j) - \exp(z_i)\exp(z_i)}{(\sum_j z_j)^2}$$
$$= p_i(1-p_i) \tag{22}$$

$$\frac{\partial E(\omega)}{\partial p_i} = \frac{\partial}{\partial p_i} - \sum_{n=1}^{N}\sum_{k=1}^{K} y_k^{(n)} \ln p_k^{(n)} = -\sum_{n=1}^{N} \frac{y_i^{(n)}}{p_i^{(n)}} \tag{23}$$

$$\frac{\partial E(\omega)}{\partial w_i} = \frac{\partial E(\omega)}{\partial p_i}\frac{\partial p_i}{\partial z_i}\frac{\partial z_i}{\partial w_i} = -\sum_{n=1}^{N} y_i^{(n)}(1-p_i) x^{T^{(n)}} \tag{24}$$

$$\frac{\partial E(\omega)}{\partial b_i} = \frac{\partial E(\omega)}{\partial p_i}\frac{\partial p_i}{\partial z_i}\frac{\partial z_i}{\partial b_i} = -\sum_{n=1}^{N} y_i^{(n)}(1-p_i) \tag{25}$$

$$w_i^{(t+1)} = w_i^{(t)} + \alpha \sum_{n=1}^{N} y_i^{(n)}(1-p_i) x^{(n)} \tag{26}$$

$$b_i^{(t+1)} = b_i^{(t)} + \alpha \sum_{n=1}^{N} y_i^{(n)}(1-p_i) \tag{27}$$

α is learning rate, and we can run (26) and (27) iteratively to update parameter w and b. Now we can put run our classifier pixel by pixel in any given image, relabel each pixel in image as {1 = red, 2 = green, 3 = blue}. In this report, we only care about blue region, we apply

$$pixel = \begin{cases} 1, if\ it's\ blue \\ 0, if\ it\ isn't\ blue \end{cases} \tag{28}$$

We will get binary image where 1 represents blue region.

B. *Bin-shape identification*

To detect bin-shape in the binary image, we first apply erosion to eliminate some noise. Erosion shape:

$$\begin{bmatrix} 00100 \\ 11111 \\ 00100 \end{bmatrix}$$

Then, we use scikit-image package to **label** each region and get **regionprops.** We know that blue bin won't be too small or too large, so the first condition is that

$$5\%\ of\ image\ width \leq blue\ region\ width \leq 80\%\ of\ image\ width \tag{29}$$

Then, we reconstruct our binary image, combine all overlapped bounding boxes into same label. However, it's sufficient to detect blue bin, some of the region might be sky or blue wall; therefore, we need to use the property of bin-shape. We know recycle-bin has standard format {19" x 26" x 36.75", 22" x 26.5" x 44", 23" x 31.5" x 46"}, each bin's width must be smaller than its height, so the second condition is that

$$blue\ region\ width < blue\ region\ height \tag{30}$$

Thus, all the labeled regions pass these two conditions (29) and (30) will be detected as blue bin.

IV. Result

A. *Pixel classification*

The weight and bias for problem 1.

Weight

$$\begin{bmatrix} w_{r1} & w_{g1} & w_{b1} \\ w_{r2} & w_{g2} & w_{b2} \\ w_{r3} & w_{g3} & w_{b3} \end{bmatrix}$$

$$= \begin{bmatrix} 19.37039397 & -9.35861042 & -9.40289135 \\ -8.6467279 & 19.65500204 & -9.2323462 \\ -9.60991067 & -9.37327879 & 18.88571381 \end{bmatrix}$$

Bias

$$[b_r \quad b_g \quad b_b]$$

$$= [0.24267098 \quad 0.35778989, \quad 0.17203291]$$

Learning rate α = 0.1, number of iteration = 3000

Take a blue color [3 54 205] as example.

First, unify each value.

$$[3\ 54\ 205] => [0.0117\ 0.2118\ 0.8039]$$

Then apply (12) to get z.

$$[0.0117\ 0.2118\ 0.8039]\begin{bmatrix} 19.37039397 & -9.35861042 & -9.40289135 \\ -8.6467279 & 19.65500204 & -9.2323462 \\ -9.60991067 & -9.37327879 & 18.88571381 \end{bmatrix}$$
$$+ [0.24267098 \quad 0.35778989 \quad 0.17203291]$$
$$= [-9.0875 \quad -3.1240 \quad 13.2891]$$

Apply softmax to turn z into probabilities.

$$s(z) = \left[p_r\ p_g\ p_b\right] = [0.000001\ \ 0.000007\ \ 9.99999]$$

We choose highest probability, that is Blue, so the result is correct.

Result on validation set is 100% correct for all three colors. In gradescope autograder, it also gets 100% accuracy; therefore, logistic regression is successful in color classification.

B. Bin detection

To get more precision on blue recycle-bin, I use 8 colors (Red, Green, Blue, Skyblue, Black, White, Yellow, Gray).

The weight and bias for 8 colors classification:

Weights

$$\begin{bmatrix} 24.52 & -12.32 & -2.817 & -7.508 & -0.761 & -10.44 & 21.92 & 16.48 \\ -18.67 & 26.31 & -28.99 & -2.883 & -14.36 & 18.11 & 10.66 & 11.49 \\ -12.49 & -19.15 & 38.24 & 22.75 & -10.40 & 23.85 & -29.78 & -11.09 \end{bmatrix}$$

Bias

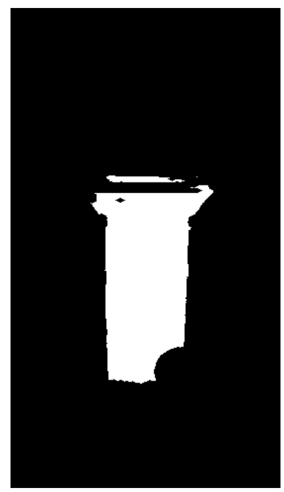
$$[4.450\ \ 6.722\ \ 1.011\ \ -6.196\ \ 13.58\ \ -20.11\ \ 1.746\ \ -4.203]$$

Take fig(1) as example, set blue region to 1 and others to 0, we will get binary image as fig(2).

Fig(1)

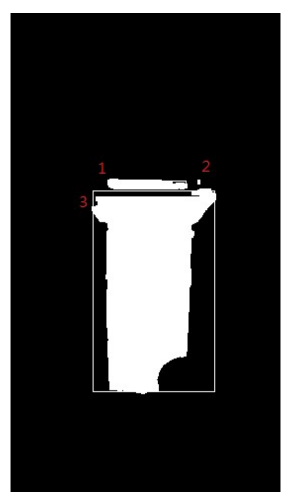
Fig(2)

Then we apply erosion twice to get rid of noise. Shown as fig(3).



Fig(3)

Then, we label each remaining region and box it. Shown in fig(4).



Fig(4)

From fig(4), we can tell there are 3 boxed region, and we need to determine whether it's recycle-bin. We use (29) and (30). For region 1, it fail condition (29), its width is larger than it height, so region 1 isn't recycle-bin. For region 2, it fail condition (30), its width is too small, so it can't be recycle-bin. Only region 3 passes all conditions, therefore, we say region 3 is recycle-bin.

Result on validation set is 90% correct. In gradescope autograder, it gets 80% accuracy.

References

1. http://medium.com/konvergen/cross-entropy-and-maximum-likelihood-estimation-58942b52517a
2. http://stats.stackexchange.com/questions/370723/how-to-calculate-the-derivative-of-crossentropy-error-function
3. http://www.analyticsvidhya.com/blog/2020/12/beginners-take-how-logistic-regression-is-related-to-linear-regression