Color Classification and Recycling Bin Detection

Muqing Li — PID: A92108137

1 Introduction

With the increasing trend of building an environment-friendly community, collecting recyclables has become very important. However, finding all blue recycling bins over one road when collecting is still a problem for the human employee. People might not identify each container precisely over a long time. Hence, it will be better if we have an excellent classifier to help human drivers detect the recycle bins and improve the efficiency of collecting recyclables.

In this report, I proposed one model that can easily and quickly find and make bounding blue recycling bins. This model first uses one color classifier based on logistic regression, morphological operations, and bounding area extension to identify blue recycling bins.

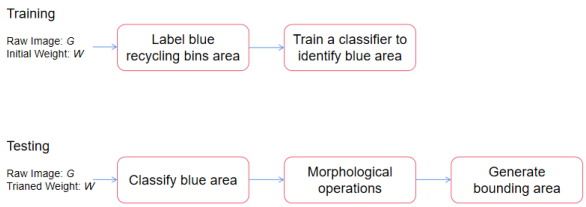

Figure 1: Overview of Training and Testing Process

2 Problem Formulation

2.1 Color Classification

Consider we have an pixel X with 3 layers of colors: (R, G, B), and we need to identify which color it belongs to. Let the true image colors to be an $3 * 1$ one-hot encoder Y. We want to train a $3 * 3$ weight w which project X into $\hat{Y}$ depending on $X^T * w$, which is the probabilistic prediction. The goal is finding w s.t. $min|\hat{Y} - Y|$.

2.2 Blue Bin Identification

Consider an graph G has (n, m) pixels, among which each pixel has three layer of base colors: (R, G, B). Then each color has a size of $n * m * 3$ pixels. We want to find a $3 * 2$ weight to $min|\hat{Y} - Y|$, where Y is the $m * n * 2$ labeled (n, m) pixels image with last 2 one-hot encoder, and $\hat{Y}$ depends on $X^T * w$ which is the probabilistic prediction.

3 Technical Approach

We use $softmax$ as normalized exponential function to generate logistic function to multiple dimensions. $\hat{y} = softmax(X^T * w)$. Also, the $softmax$ σ is defind as $\sigma(z)_j = \frac{e^{X^T * w}}{\sum_{k=1}^{K} e^{x^T * w_k}}$, where j is

the j_{th} class, K is total number of classes, and $\sigma(z)_j$ is the probability of j_{th} class.

3.1 Color Classification

For color classification, we want to generate a $3 * 3$ weight w to minimize the loss. Hence, we are using the following algorithm for training.

Algorithm 1 Color Classification

Initialize the weight w, learning rate lr, number of training time *epoch*
Get input X, number of training examples M in X
for iteration = 1,2,3,...,epoch **do**
$\hat{y} = softmax(X^T * w)$
$dw = \frac{-1}{M} * (Y - \hat{Y}) * X + lr * w$
$w = w - lr * dw$
end for

Since we only generate the probability of our model output, we use $argmax$ to get the position of probability. For example, if the probability we generate is $[0.8, 0.1, 0.1]$, the $agrmax$ output would be $[1, 0, 0]$.

3.2 Blue Bin Identification

For this problem, other than the 3-color classification in the previous problem, we want to generate a binary output to only verify whether this pixel is blue. Hence, our output is a two dimension one-hot encoder, and weight w is an $3 * 2$ matrix. Before training, we use *roipoly* function provided for generate the training labels. Suppose each image has size of (n, m) pixels and $3 - RGB$ channels. The *roipoly* will generate an (n, m) matrix with each elements in it to be either 0(non blue) or 1(blue).

We using the following algorithm for training.

Algorithm 2 Blue Bin Identification Training

Initialize the weight w, learning rate lr, number of training time *epoch*, number of images for training I
for iteration = 1,2,3,...,epoch **do**
for i = 1,2,3,...,I **do**
flatten image X from $(n, m, 3)$ to $(n * m, 3)$
$\hat{y} = softmax(X^T * w)$
$dw = \frac{-1}{M} * (Y - \hat{Y}) * X + lr * w$
$w = w - lr * dw$
end for
end for

We use the following for algorithm to generate the boundary box:

Algorithm 3 Blue Bin Identification

for i = 1,2,3,...,image **do**
Generate the binary image by Logistic Regression
Morphological Operation: Opening and Closing
Get bounding box upper left point and lower right point by *regionprops*
Extend the boxes by some pixels
end for

When generating the box, we find that, the width of any blue recycle bin is always shorter than length. Hence, we only select the boxes with a greater value in length.

4 Result

4.1 Parameter

4.1.1 Weight for color classification

$$\begin{bmatrix} 2.75162998 & -1.4468946 & -1.51191691 \\ -1.21596619 & 2.48733989 & -1.65745635 \\ -1.13724369 & -1.41401972 & 2.15218174 \end{bmatrix}$$

4.1.2 Weight for blue dedection

$$\begin{bmatrix} 0.86034344 & 0.31841669 & -0.33153088 \\ -0.86034344 & -0.31841669 & 0.33153088 \end{bmatrix}$$

4.2 Bounding Box Coordinates

Image	Coordinate
1	[[192, 147, 326, 301]]
2	[[8, 345, 152, 519]]
3	[[154, 76, 271, 200]]
4	[[344, 96, 475, 292]]
5	[[799, 402, 943, 641]]
6	[]
7	[[694, 295, 843, 528] [572, 310, 720, 523]]
8	[]
9	[]
10	[]

4.3 Results and Discussion

4.3.1 Results

For Pixel Classification, the train precision is 0.9926908500270709; validation result is 0.975904; and test result is 9.879518072289157/10.

For Blue Bin Detection, since we only use the train data for training weight w, we do not perform detection during training. For validation, the precision is 100%, and for test, the score is 9.5/10

4.3.2 Discussion

I think these results generally satisfy the requirement of the problems. The algorithms work very well in detecting the true boxes in most cases. We do not directly train the classifier to detect boxes, instead we manually define what is box. As a result, the precision could be compromised in some cases. To improve the algorithm's performance in the future, Convolutional Neural Network could potentially be a good fit in training how to find the box.

4.4 Examples of segmented color images, next page

Since we do not convert color to RGB. It might not looks like blue. However,in actual image, it is blue. **Examples see next page.**

Figure 2: Example 1

Figure 3: Example 2

Figure 4: Example 3