Challenge Project Report

Bo Jianyuan

1. Introduction

This report proposes a credit card detection system which contains two main parts as we showed in Figure 1. The first part is credit card detection. We applied Haar Feature-based Cascade classifier to detect credit card. Besides, we use canny edge detector to find contour of detected credit card. The experiment results show that proposed system could detect credit card on any background and draw contour of detected card nicely.

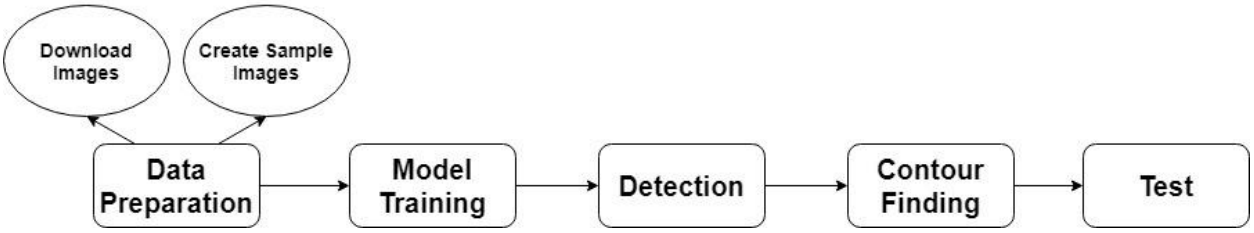

Figure 1. Framework of credit card detection system

The remaining report contains 4 sections. Section 2 is review of related methods and section 3 introduces credit card detection system in detail. Section 4 explains and analyzes the experiment results and Section 5 describe the conclusion and future works. All the materials and programs are on my Github page (https://github.com/BMDroid/Challenge-Project).

2. Literature Review

Haar Feature-based Cascade classifier was proposed by Viola *et.al* [1] firstly. This method is an efficient object detection algorithm with high detection rate. Inspired by [2], [1] proposed to use a set of features which are reminiscent of Haar Basis functions which are called integral image instead of use image intensity directly. Feature extraction part obtains large number of features. In order to make this algorithm rapid, [1] apply Adaboost to classify a small number of important features. After getting these important features, [1] combined successively more complex classifiers in a cascade structure. In [2], Lienhart *et.al* enhanced Haar Feature-based Cascade classifier by extending the feature set. This method can also be computed efficiently and achieve better performance on face detection than [2]. In this report, we apply the extended version on out credit card detection system.

3. Methodology

In this section, we introduce credit card detection system in 4 parts: data preparation, model training, credit card detection and contour finding.

Data Preparation

There is no dataset of credit card images online, but we need a lot of positive samples for the training. Thus, this project leverages on the OpenCV built-in function which can create many sample images with bounding box information.

1. Download Images
   For negative images, I selected the floor and fabric images from the ImageNet and resize them to 100 * 100, and the total number of the negative images is 1085.

Figure 2. Sample Negative Image

For the positive images, I downloaded credit card images from Google Image and choose 73 of them. Since we need use these images to create sample images for the training, the background of the credit card had all been cropped. Then I resized them all to 288 * 180.

Figure 3. Sample Positive Image

After the images were downloaded, the description file “neg.txt” which contains the file path for each negative image was created.

2. Create Sample Images

Since, we already know that for the face recognition task, the frontal face haar cascade classifier is trained by using only frontal face sample images with no face rotation in it. Thus, for our classifier, we should do the same which is to create many sample images with different credit cards with no rotation.

According to the OpenCV documentation of the haar cascade classifier training, the following command was used for creating the sample images.

```
opencv_createsamples -img pos_resize/pos_01.jpg -bg neg/neg.txt -info
samples/samples_{img[-6:-4]}.txt -pngoutput samples -num 128 -maxxangle 0 -
maxyangle 0 -maxzangle 0 -bgcolor 255 -bgthresh 8 -maxidev 40 -w 48 -h 30
```

We could see that the for each credit card image we created 128 sample images with 0 rotation on any axis. For the 48 width and 30 height, it follows the original credit card image ratio. And we need to conduct this command for all the credit card images. Then, we got a sample images size of 9344.

Figure 4. Created Sample Image

Model Training

With all the files prepared, we could finally train our model. To get a model with relatively high accuracy and avoid high false alarm rate, several training parameters are tested. If we set the training stages too large and the maximum false alarm rate too high, the haar feature based classifier will not able to detect anything according to the experiments.

Table 1. Parameters for Training Cascade Classifier

No.	numStages	maxFalseAlarmRate	Performance
1	20	0.5	High false alarm rate
2	20	0.2	Cannot detect anything
3	**6**	**0.2**	**Good accuracy and low false alarm rate**

I choose the 3rd parameters in the table to train the model. The command is showed below:

```
opencv_traincascade -data output -vec pos.vec -bg neg/neg.txt -numPos 1000 -numNeg
500 -numStages 6 -precalcValBufSize 1024 -precalcIdxBufSize 1024 -featureType HAAR -
minHitRate 0.995 -maxFalseAlarmRate 0.2 -w 48 -h 30
```

The ratio of the positive images and negative images is set to be 2:1. Besides, we could see that the width and height is same as in the sample creating stages. This is very important for classifier training.

The whole training process took about 1 hour to complete, by conducting the same command we could get the final training model. Then we could use the visualization tool of the haar cascade classifier to see which feature are selected for each training stages.

Figure 5. Stage 0

Figure 6. Stage 5

From visualization we could see that the classifier is selecting more useful features during the training.

Detection and Contour Finding

For this part, we have two tasks. One is to detect whether a credit card is in the image, the other is to draw the contour of the credit card.

1. Detection
   For the detection, we want the test images similar to our training samples. Thus, we need to resize them to similar size. After tuning the parameters, 150 pixels is chosen to be the standard for resizing.
   Then, we convert the resized images to gray scale and uses Gaussian blur with kernel size of 3. Finally, the model is loaded for detection. Once the card is detected, a bounding box will be created based on the detection. Though, the classifier will correctly detect the credit card in the image, the bounding box will not exactly match the contour of the credit card according to the experiments.

Figure 7. Detected Bounding Box

   These may be improved with more different training samples. Another solution needs to be found based on the limitation of the size and variance of my dataset and computational power.
2. Contour Finding
   Since the bounding box of the detection already fits most of the credit card area, could we find the contour of the possible object in the image and compute the interest of union between the actual detection and the possible object's contour to decide is the contour belongs to the detected credit card.
   Canny edge detection is used for finding the contour. Then I chose the largest external contour by using hierarchy information. And the largest contour was transformed into rectangle for easier computation of the IoU.
   According to the test, the actual detection box and the largest contour formed bounding box has very large IoU in most cases.

Figure 8. Detected Bounding Box in Red and the Largest Contour Box in Green

And 0.5 is chosen to be the smallest IoU. If the IOU falls below 0.5, the actual bounding box will be drawn on the image. Otherwise, the largest contour will be drawn.

Figure 9. Contour Drawn After Detection

4. Results

For detection accuracy, many test images are a mixed of newly created sample images and negative images. And there are 510 positive images and 510 negative images in the test set. And the error rate is 37%. For 510 negative images, there are only 39 wrong classified images. Thus, the classifier has very high true negative rate.

Overall, the model can detect the credit card images with reasonable accuracy.

5. Future Work

For robust detection, more credit card images could be collected for building more various training samples. Besides, the classifier is more sensitive for properly aligned credit card in the image without rotation. In the future, we could do the segmentation of the credit card image first then crop them and rotate them to proper angle for better credit card detection.

Reference

[1] Viola, P. and Jones, M., 2001. Rapid object detection using a boosted cascade of simple features. *CVPR (1)*, *1*, pp.511-518.

[2] Papageorgiou, C.P., Oren, M. and Poggio, T., 1998, January. A general framework for object detection. In Sixth International Conference on Computer Vision (IEEE Cat. No. 98CH36271)(pp. 555-562). IEEE.

[3] Lienhart, R. and Maydt, J., 2002. An extended set of haar-like features for rapid object detection. In *Proceedings. International Conference on Image Processing* (Vol. 1, pp. I-I). IEEE.