Learning-Based License Plate Detection Using Global and Local Features

Huaifeng Zhang, Wenjing Jia, Xiangjian He and Qiang Wu
Computer Vision Research Group,
University of Technology, Sydney
E-mail: {hfzhang, wejia, sean, wuq}@it.uts.edu.au

Abstract

This paper proposes a license plate detection algorithm using both global statistical features and local Haar-like features. Classifiers using global statistical features are constructed firstly through simple learning procedures. Using these classifiers, more than 70% of background area can be excluded from further training or detecting. Then the AdaBoost learning algorithm is used to build up the other classifiers based on selected local Haar-like features. Combining the classifiers using the global features and the local features, we obtain a cascade classifier. The classifiers based on global features decrease the complexity of the system. They are followed by the classifiers based on local Haar-like features, which makes the final classifier invariant to the brightness, color, size and position of license plates. The encouraging detection rate is achieved in the experiments.

1. Introduction

License plate recognition (LPR) has been adopted widely into numerous applications such as unattended parking, security control and stolen vehicle verification. In the LPR system, license plate detection is the most crucial step. It is extremely difficult to detect license plate from cluttered background efficiently because of the affection of variant illumination, perspective distortion, interference characters, etc. Most of previous license plate detection algorithms are restricted in certain working conditions, such as fixed backgrounds[1], known color [2], or fixed size of the license plates [3, 4]. Therefore, detecting license plate under various complex environments is still a challenging problem.

In the previous years, some researchers have been working on license plate detection in complex conditions. Chang et al. [5] proposed a robust license plate detection algorithm using color edge and fuzzy disciplines. However, their algorithm can only detect the license plates with specific colors. In [6], Matas and Zimmermann proposed an algorithm to detect license plate and road sign. They used character regions as basic units of license plate, which makes the algorithm hardly distinguish interference characters from the true license plates. Kim et al. [3] proposed another license plate detection algorithm using both global and local features. License plate templates were used as local features in their algorithm. In most cases, however, general plate templates are very difficult to be constructed. Moreover, the sizes of the global features used in their algorithm were fixed. Hence the application of this algorithm was restricted extremely.

Recently, Haar-like features were widely used for object detection [7, 8]. The classifiers based on Haar-like features can detect objects from cluttered background despite the variance of the illumination, the color, the position and the size of the objects. However, one problem of these algorithms is that too many features are included in the classifiers, which makes the system very complex and unstable. Chen and Yuille [9] constructed a simple cascade classifier for text detection using statistical features. However, only statistical features always result in high false positive rate in practice.

In this paper, we construct a cascade classifier including both global statistical features and local Haar-like features. Using Haar-like features makes our classifier be invariant to the brightness, color, size and position of license plates. On the other hand, using global statistical features makes the final classifier simple and efficient.

The rest of the paper is organized as follows. The framework of our algorithm is introduced in Section 2. Two global statistical features are defined in Section 3. Then local Haar-like features and AdaBoost algorithm are described in Section 4. Experimental results are presented in Section 5. At last the paper is concluded in Section 6.

2. The Framework of the Algorithm

In our algorithm, a cascade classifier [8] is constructed to increase the detection speed, in which the first two layers are based on global features and the following layers are based on local Haar-like features. In this section, we introduce the algorithm in two aspects: testing and training.

2.1. Training

Positive samples and negative samples are needed in the training procedure. The positive samples are obtained through labeling the license plate regions from the vehicle images. The negative samples are randomly extracted from different images which do not contain license plate. All of the samples are scaled to 48*16 for the convenience of training.

Firstly, for all the samples, the values of one of the global statistical features, called Gradient Density, are calculated. A classifier is obtained by selecting the threshold which classifies all the positive samples as positive ones. All the samples, including the positive samples and negative samples, which are classified as positive ones (true positives and false positives) are used to train the classifier of the second layer. This classifier is based on another statistical feature, called Density Variance. The input samples are classified again and the positive ones obtained in this layer are used to train the classifier of the third layer. Similarly, the samples classified as positive ones by the third layer are input to the fourth layer, and so on. The training finishes when the given false positive rate is reached. In our algorithm, we trained four layers of classifier based on Haar-like features and AdaBoost leaning procedures, which is the layer 3 through layer 6 in the final cascade classifier.

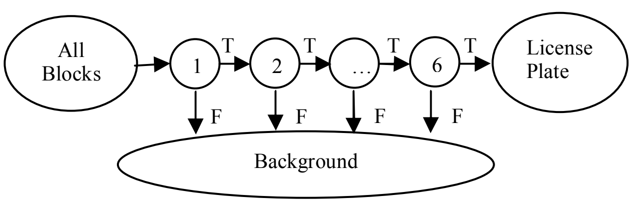


Fig. 1 The working flow of cascade classifier, where 1,2,…,6 represent the layers.

2.2. Testing

When an image is input into the classifier, a mask of 48*16 is used to capture the same size of pixel block on the image. This mask will go through the whole image. At each position, the cascade classifier is used to verify if the block covers a license plate. A cascade classifier can be taken as a degenerate decision tree as shown in Fig. 1. A positive result from the classifier on an upper layer triggers the classifier on the next layer. A negative outcome at any layer leads to an immediate rejection of the block. Then it slips to the next position and the same procedure is repeated.

The detection is implemented in multiple scales. In order to detect license plates of variant sizes, the block size is scaled up from 48*16 to 240*80, with a scaling factor of 1.2.

3. Global Statistical Features

Statistical analysis shows that the regions of license plates have some common global characteristics. In this section, two statistical features of the block of a license plate are defined, with which the common characteristics of license plates can be represented.

3.1. Gradient Density

According to our observation, regions that contain a license plate tend to have a high density of edge information. For algorithm simplicity purpose, the gradient information is investigated rather than edge information in this paper because an efficient general-purpose edge detector is usually difficult to obtain in practice.

The gradient density in a block is used to describe the edge density of the block using

$$D_G = \frac{1}{N}\sum_i \sum_j G(i,j), \tag{1}$$

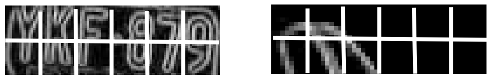
where $G(i, j)$ represents the gradient magnitude at location (i, j) and N is the number of pixels in the block.

The Sobel gradient operator is employed to produce gradient map, where the resulted gradient magnitudes are normalized by the maximum gradient strength in the image.

During the training procedure, the size of the block is fixed to the size of the sample images, which is 48*16. During the testing procedure, the size of the block is changed depending on the scale of the searching block.

3.2. Density Variance

Besides the abundant edge information, note that the foreground characters in a license plate are usually distributed with relatively even interval. As its consequence, the gradient in the block of a license plate is distributed more evenly in space with similar strength [10], compared to most of the areas with simple structures. Fig. 2 gives such an example.



(a) License Plate (b) Background

Fig. 2. Areas with different gradient distributions

Therefore, we modify and redefine the density variance feature [10] in order to discriminate license plates from background regions.

To obtain the feature, a block is divided into 12 equal-sized sub-blocks, as shown in Fig. 2. Let g_i denote the mean value of the gradient strength at sub-block i, and g denote the mean value of the gradient strength of the whole block. Then, the density variance of the block, denoted as V_G, is defined as

$$V_G = \frac{\sum_{i=1}^{n} |g_i - g|}{n \cdot g}, \qquad (2)$$

where n is the number of the sub-blocks, e.g., $n = 12$ in above example.

The above defined density variance, which takes value from 0 to 1, is a ratio to the mean gradient strength of the block. The density variance keeps low as long as there are *similarly* strong or weak gradient distributed evenly through the block.

4. Local Haar-like Feature and AdaBoost

The Haar-like features originate from Haar basis functions. They consist of a number of rectangles covering adjacent image regions (see Fig. 3). The value of a Haar-like feature is the difference between the average of the pixel values (in our algorithm, the gradient magnitude) in white rectangles and grey rectangles.

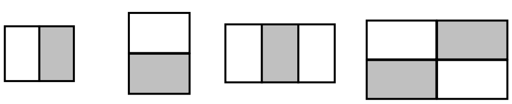


Fig. 3. Four types of Haar-like features

A Haar-like feature is determined by its type, the size and the position of the rectangles. The size and the position can be any as long as the feature is in the image block. Such Haar-like features dictionary can capture the interior structure of objects that are invariant to certain transformations. However the number of the features is too large in this features dictionary, e.g. there are hundreds of thousands features in a 48*16 image block. It is prohibitively time-consuming to compute all the features.

AdaBoost algorithm [11] is a good choice to select a small number of features from a very large number of potential features. The classifier trained through AdaBoost algorithm is the combination of a set of simple classifiers (called weak classifier). Each weak classifier uses one feature. The construction of weak classifier is independent of AdaBoost algorithm. In our algorithm, perceptron is selected as the weak classifier, in which the classifying threshold is determined by the given detection rate.

The basic idea of the AdaBoost algorithm is as follows. After constructing a weak classifier, the samples are re-weighted in order to emphasize those which are incorrectly classified. Then the next weak classifier is trained with the re-weighted samples. A number of weak classifiers are trained in this way till the given false positive rate is reached. The final classifier (called strong classifier) is constructed by combining these weak classifiers using a set of weights. These weights are determined by classification error of each weak classifier.

5. Experiments

In the experiments, 460 images containing license plates and 500 images without license plate are used. Among these positive images, 300 images are taken as training images, in which there are 305 visible license plates; the other 160 images are testing images, in which there are 169 visible license plates. All the negative samples are extracted from the 500 negative images. The images used in our experiments were taken in various circumstances with various illuminations and view angles. Some examples of the license plates are shown in Fig. 4.

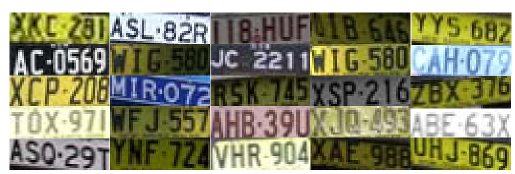
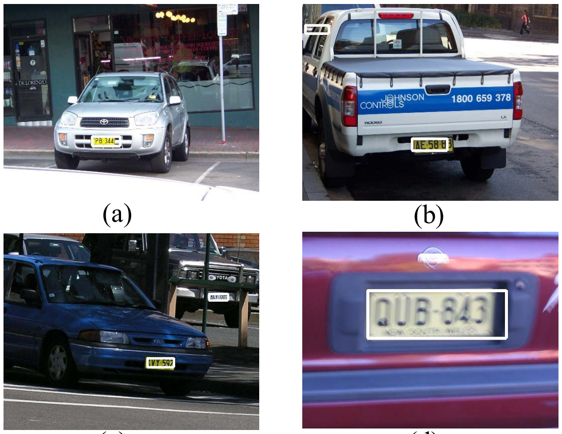


Fig. 4. Some examples of the license plates

The negative samples used to train the classifiers based on global features are collected by randomly selecting 28,000 sub-windows from part of the negative images (50 images in our experiments). The negative samples used in AdaBoost learning procedure are the false positive samples obtained from the previous layers of the cascade classifier.

In the experiments, a six-layer cascade classifier is obtained. Each of the first two layers uses one of the global features defined in Section 3. On the last four layers, the numbers of the features in the strong classifiers are 19, 34, 47 and 58 respectively. So our final cascade classifier has 6 layers and 160 features. This classifier is much simpler than Viola's classifier which has 38 layers and 6060 features [8].

In our experiment, among the 169 visible license plates in 160 testing images, 158 license plates are detected, with detection rate 93.5%. At the same time, there are only 8 false positive regions. On a PC with Pentium 2.8GHz CPU, the detector can process a 648*486 image in about 80ms.



(a) (b) (c) (d)

Fig. 5. Detection results of some vehicle images

Fig. 5 shows some of the detection results, where the license plates are circled by white boxes. From the examples, we may see that our algorithm can detect the license plates with various sizes, positions and colors from various backgrounds. Fig. 5(a) is an example with cluttered background; Fig. 5(b) shows the license plate detection against interference characters; Fig. 5(c) is the result of detecting multiple license plates in one image; Fig. 5(d) shows that the algorithm can detect large size and blurred license plate.

6. Conclusions

In this paper, we construct a cascade classifier for license plate detection using both global and local features. The classifiers on the first two layers are based on global features. They can exclude more than 70% background regions from further training or detecting. The classifiers on the following layers are based on local Haar-like features. The final classifier is invariant to the brightness, color, size and position of license plates. 93.5% detection rate and very low false positive rate are both obtained when the license plate detection algorithm works in various complex environments. Moreover, the detection procedure is very fast in our algorithm.

7. Acknowledgement

This work was sponsored by the Australian Research Council under Large ARC-Discovery Grant DP0451666.

8. References

1. H. Bai and C. Liu, *A Hybrid License Plate Extraction Method Based on Edge Statistics and Morphology*, in *Proc. of the International Conference on Pattern Recognition*, p.831-834 vol.2, 2004.
2. S. K. Kim, D. W. Kim and H. J. Kim, *A Recognition of Vehicle License Plate Using a Genetic Algorithm Based Segmentation*, in *Proc. of International Conference on Image Processing*, p.661-664 vol.2, 1996.
3. S. Kim, D. Kim, Y. Ryu and G. Kim, *A Robust License-plate Extraction Method under Complex Image Conditions*, in *Proc. of International Conference on Pattern Recognition*, p.216-219 vol.3, 2002.
4. W. Jia, H. Zhang, X. He and M. Piccardi, *Mean Shift for Accurate License Plate Localization*, in *Proc. of International Conference on Intelligent Transportation Systems*, p.566-571, 2005.
5. S.-L. Chang, L.-S. Chen, Y.-C. Chung and S.-W. Chen, *Automatic License Plate Recognition.* IEEE Transactions on Intelligent Transportation Systems, 2004. **5**(1): p. 42-53.
6. J. Matas and K. Zimmermann, *Unconstrained License Plate and Text Localization and Recognition*, in *Proc. International Conference on Intelligent Transportation Systems*, p.572-577, 2005.
7. T. Mita, T. Kaneko and O. Hori, *Joint Haar-like Features for Face Detection*, in *Proc. of the International Conference on Computer Vision*, p.1619-1626, 2005.
8. P. Viola and M. J. Jones, *Robust Real-Time Face Detection.* International Journal of Computer Vision, 2004. **57**(2): p. 137-154.
9. X. Chen and A. L. Yuille, *Detecting and Reading Text in Natural Scenes*, in *Proc. of the International Conference on Computer Vision and Pattern Recognition*, p.366-373 Vol.2, 2004.
10. W.-Y. Chen and S.-Y. Chen, *Adaptive Page Segmentation for Color Technical Journals' Cover Images.* Image and Vision Computing, 1998. **16**(12-13): p. 855-877.
11. Y. Freund, *An Adaptive Version of the Boost by Majority Algorithm.* Machine Learning, 2001. **43**(3): p. 293-318.