Light-YOLOv3: License Plate Detection in Multi-Vehicle Scenario

Yuchao SUN\(^{(a)}\), Qiao PENG\(^{1}\), Nonmembers, and Dengyin ZHANG\(^{1}\), Member

SUMMARY With the development of the Internet of Vehicles, License plate detection technology is widely used, e.g., smart city and edge sensor monitor. However, traditional license plate detection methods are based on the license plate edge detection, only suitable for limited situation, such as, wealthy light and favorable camera’s angle. Fortunately, deep learning networks represented by YOLOv3 can solve the problem, relying on strict condition. Although YOLOv3 make it better to detect large targets, its low performance in detecting small targets and lack of the real-time interactively. Motivated by this, we present a faster and lightweight YOLOv3 model for multi-vehicle or under-illuminated images scenario. Generally, our model can serves as a guideline for optimizing neural network in multi-vehicle scenario.

key words: plate detection, YOLOv3, multi-vehicle scenario

1. Introduction

With the continuous development of Internet of Vehicles technology and the continuous expansion of traffic volume, traffic management in complex traffic scenarios has become a major problem. The license plate is the only identification mark of the car’s identity and the automatic detection of the license plate \cite{1} plays an important role in intelligent transportation. Traditional license plate detection \cite{2} technology is based on graphic segmentation and graphic recognition theory to determine the position of the license plate in the picture to complete the detection. This technology can achieve satisfactory results in a good image acquisition environment.

Tian Ying et al. \cite{3} introduced the method of slack variables and penalty function combined with the concept of optimal hyperplane to build a feature extractor that can automatically synthesize a license plate feature from a large number of positive and negative samples of license plates, effectively improving the accuracy of license plate detection and positioning. Yuan Yu et al. \cite{4} extracted the color features and texture features of license plates by histogram back projection method and Sobel operator, and then combined the two features to obtain the connected regions as license plate candidate regions. Even if the convolution network is not good enough to locate the license plate, it can detect the position of the license plate by using the neural network.

Most of the existing methods analyze the license plate features from the license plate samples, extract the commonness, and get the location of the license plate in the image. Therefore, they only have a good performance in some specific scenarios. When the license plate background color is similar to the license plate color, Their performance is a bit unsatisfactory. Moreover, they cannot meet the high real-time requirements in scenarios with dense traffic. In recent years, deep neural networks have become a research hotspot. There are many high-precision target detection methods based on deep learning, which are mainly divided into one-stage networks and two-stage networks. The two-stage network first generates a series of sample region proposals by an algorithm, and then performs granular target detection. The representative algorithms are R-CNN\cite{5}, Fast-RCNN \cite{6}, and Faster-RCNN \cite{7}. However, the two-stage network detection speed is slow, which cannot meet the real-time performance in traffic scenarios. One-stage network directly converts the problem of target frame positioning into regression problem processing, which greatly improves the speed of image detection, while also providing good accuracy. Representative algorithms include SSD \cite{8} and YOLO \cite{9}–\cite{11}.

You only look once (YOLO) \cite{9} as a representative of the one-stage detection model. It takes object detection as a regression problem to solve. The input image can get the position of all the objects in the image, their category and the corresponding confidence probability after one input. The detection and training are completed in a separate network. Therefore, YOLO can achieve fast detection and high accuracy at the same time. YOLO has achieved good results in target detection in many fields, but it has not directly applied YOLO to license plate detection in traffic scenarios. This paper proposes a license plate detection model based on YOLO.

2. License Plate Detection System Model

2.1 YOLOv3

YOLOv3 proposed Darknet53 as the backbone convolutional network. Darknet53 contains a total of 53 convolutional layers. A large number of residual modules composed of 3×3 and 1×1 convolutions are stacked on top of each other. In the middle of the network, 3×3 convolutional layer with stride of 2 is used to scale the feature map.

As shown in Fig. 1, YOLOv3 extracts feature maps of
different sizes in the convolution process for feature fusion, and then divides it into three scales for output prediction. YOLOv3 extracts the feature maps of different sizes during the convolution process for feature fusion, and then outputs them in three scales. First, the input of the network is an image of 416×416. When passing through the darknet53 network structure, it will be scaled five times to obtain 208×208, 104×104, 52×52, 26×26, 13×13 feature map. Because the feature maps of the first two sizes do not contain accurate feature information, the last three feature maps of 52×52, 26×26, and 13×13 are taken as large-scale, medium-scale, and small-scale predictions. Second, the small feature map and the large feature map are merged by up-sampling the feature map and output three predicted features. Finally, Using the probability score as a threshold to filter out most anchors with lower scores. Then use non-maximum suppression (NMS) [12] for post-processing, leaving a more precise box.

2.2 Improved License Plate Detection Model

We redesigned the backbone network structure of YOLOv3 in Fig. 2. We reduced the computational parameters in the model by replacing the normal convolution in the backbone with depthwise separable convolution. Considering that the reduction of the backbone network layer will negatively affect the accuracy of feature extraction, We propose a new network unit structure. Firstly, we use the Channel Shuffle method to reorganize the feature information stream of different groups of convolutions. Secondly, We use the Spatial Pyramid Pooling module to improve the expressive ability of feature maps. Finally, we introduce the scale conversion module to enhance feature extraction. Scale conversion module is composed of Pooling and Scale-transfer Layer. Among them, Pooling is mainly used to obtain small-scale feature maps, while Scale-transfer Layer is used to replace the up-sampling layer in the original FPN network. The large-scale is obtained by reducing the number of feature maps. For the standard map, no parameters are added in the whole process.

2.2.1 Network Unit in Backbone Network

We change the standard convolution to depthwise separable convolutions according to Google “Xception” [13]. The standard volume integral is solved into two layers: The first layer is depthwise convolution, which decomposes a single filter of size N×N×N in the standard convolution into N filters, each of which has a size of N×1, and depthwise convolution applies a single filter to each input channel. The second layer is 1×1 convolution, which is called point-wise convolution. Point-wise convolution applies 1×1 convolution to combine the output of depth convolution, and the main function is to expand the depth. Here, we define the input feature map size as F×F×M(Here we define the number of input channels as M, and the feature map size as F×F), the output feature map size as H×H×N.(Here we define the number of output channels as N and the feature map size as H×H)

Here we define the number of output channels as N and the feature map size as H×H:

\[
\text{cost}_1 = H \times H \times M \times N \times F \times F
\] (1)

The computational cost of depthwise separable convolution is:

\[
\text{cost}_2 = H \times H \times M \times F \times F \times M \times N \times F \times F
\] (2)

The ratio of computational cost required for depthwise separable convolution and standard convolution is:

\[
\frac{\text{cost}_2}{\text{cost}_1} = \frac{1}{N} + \frac{1}{H^2}
\] (3)

Then, the backbone network unit uses the residual connection method shown in the Fig. 3. In this network, each
residual network contains 3 layers. The three layers are $1 \times 1$, $3 \times 3$, and $1 \times 1$ convolutions. We define $C_i$ as the input channel and $C_o$ as the output channel. We use $1 \times 1$ convolution to amplify the input to $K \times C_i$ and filter it through a lightweight $3 \times 3$ depth separable convolution. Finally, we perform a dimensionality reduction operation on it through $1 \times 1$ convolution to restore the original dimension.

However, as shown in Fig. 4(a), if multiple groups of convolution are superimposed together, the feature extraction ability of the network will be reduced. Here we add channel shuffle after a point-wise convolution group as shown in (b). The information flow between the amplified convolutional layers can be reorganized across groups, thereby improving the feature extraction ability of the network.

Finally, we use the spatial pyramid pooling module in SPPNet [14] network for reference. At the end of the network, spatial pyramid pooling module is added to enrich the expression ability of feature graph. As shown in the Fig. 5, SPP module is composed of maximum pooling with kernel size of $5 \times 5$, $9 \times 9$, $13 \times 13$ and a jump connection. After the fusion of local features and global features, the expression ability of feature map is enriched, which is conducive to the situation that the target size in the image to be detected is greatly different, and the detection accuracy is improved.

### 2.2.2 Scale Conversion Module

As shown in Fig. 6, we use the feature map scaling method in STOD [15] to replace the up-sampling method in the original FPN network, which destroys the original data and has a huge computational load. Scale-transfer is composed of a pooling layer and a scale-transfer layer. Among them, the Mean pooling layer is used to obtain low-resolution feature maps; Scale-transfer mainly reduces the number of channels by decreasing the number of channels and enlarge the width and height of the transmission layer.

First of all, the $13 \times 13$ and $26 \times 26$ feature maps are scaled up by the method of fusion migration, where our up-sampling factor is set to 2. The convolutional dimension reduction operation is performed through $1 \times 1$ convolution kernel with depth of 64. Then the feature is extracted by convolution operation of $3 \times 3$ convolution kernel with depth of 64. Finally, the dimension is increased by using $1 \times 1$ convolution kernel with depth of 64 and the prediction feature is added with the upper feature.

### 3. Experiment Results and Analysis

#### 3.1 Environmental Configuration and Preparation

The configurations of Environment are as follows: (1) CPU: Intel Xeon E5-1620 v4 @ 3.50 GHz; (2) Graphics Card: NVIDIA RTX 2080 Ti, 8 GB GDDR5; (3) Memory: 64 GB DDR4; (4) Operating system: Linux(Ubuntu 16.04.5); (5) Frame: TensorFlow

First, we make datasets. We use the images captured...
by surveillance cameras in traffic scenes as tagging images. There are 2269 images, including green cards, blue cards, yellow cards and mixed color cards. Our dataset annotation tool is LabelImg. Among them, 1869 images were used to train the model and 400 images were used to test the model. Then, we use our network to train the prepared data set. Some key experimental configuration parameters of the network are shown in Table 1.

### Table 1: Experimental configuration parameters.

| Parameter    | Value |
|--------------|-------|
| NMS_score    | 0.4   |
| IOU          | 0.5   |
| Confidence_score | 0.001 |
| Leaky_relu   | 0.1   |

3.2 Performance Evaluation

We choose Frames Per Second (FPS) and mAP(mean Average Precision) to evaluate the performance of the model. FPS indicates the average detection time of each picture when detecting a batch of pictures per unit time. mAP can be expressed as the average progress average when detecting the license plate category. In our experiment, We call a positive sample if the confidence result is greater than or equal to 0.5. FPS is one of the important indicators to measure the real-time detection rate of the model, defined as the number of detected pictures in period time.

In order to evaluate the accuracy, we define mAP as follows:

\[
Precision = \frac{TP}{TP + FP}
\]

\[
recall = \frac{TP}{TP + FN}
\]

where TP in Eq. (4) represents true positive samples. FP indicates false positive samples, meanwhile FN in Eq. (5) represents false negative samples.

\[
mAP = \frac{\sum_{i=1}^{\text{classes}} \int_{0}^{1} p_i(r_i)dr_i}{\text{Num(classes)}}
\]

where \(P_i\) and \(R_i\) represents the signal precision and recall respectively. \(\text{Num(classes)}\) is the length of classified species.

3.3 Results and Analysis

In order to verify the accuracy of the proposed model in complex traffic situations, we compared the improved model with the original YOLOv3 version. We use a self-made data set to train the model. In the test set, we selected 400 pictures for testing.

We test and classify the license plates of different types in 400 images. As shown in Table 2, we select different color license plates to test the model and find that the mAP of detecting mixed color card is lower than that of monochrome card. Because the mixed color card adopt the license plate splicing method, and the background overlap causes the confidence to be too low, which ultimately reduces the detection rate. In addition, it can be seen that the mAP of our proposed model can reach 0.95 when checking monochrome license plates, which means that our method has good performance.

We compare the FPS of Light-YOLOv3 and YOLOv3 in Fig. 7. The figure shows the FPS of the two model detection images in one minute. Under the same environment configuration, We use batch images to test the FPS of both models. Experimental results show that Light-YOLOv3’s FPS can reach 26-27 frames/s, while YOLOv3 can only reach 22-23 frames/s. Therefore, Light-YOLOv3 has better real-time performance on detection images.

As can be seen from Table 3, in complex traffic scenarios, the mAP value of our Light-YOLOv3 network is not different from that of the original YOLOv3 network, but it was 20 percent faster than the original YOLOv3 detection. The main reason is that we replace the ordinary convolution with the deep separable convolution in the backbone network, which greatly reduces the calculation parameters and improves the detection rate of the model. In order to compensate for the negative impact of data loss caused by convolution layer replacement, We add channel shuffle to recombine the feature information in different convolution groups to supplement the depth information of feature image. At the end of the network, SPP module is added to enrich the expression ability of feature map. Finally, the scale-up module is used to replace the original upsampling method in FPN network, which makes full use of the visual information provided by different channels on the basis of avoiding the original data being destroyed.

Figure 8 shows the detection effect of Light-YOLOv3
Table 3 The values of mAP and FPS

| Method      | mAP  | FPS     |
|-------------|------|---------|
| YOLOv3      | 0.9563 | 22.3174 |
| Light-YOLOv3 | 0.9582 | 26.6271 |

model and YOLOv3 model. It can be seen from 1(a) that YOLOv3 model will not detect accurately when detecting mixed color license plate. The detection effect of our method is shown in 1(b). We can detect the mixed color license plate completely in the image. It can be seen from 2(a) that in the multi vehicle traffic scenario, YOLOv3 model will miss the inspection. In contrast, as shown in 2(b), our method can detect all license plates in the image in the multi vehicle scene. Therefore, the model has a good detection effect.

4. Conclusion

In this paper, we use the method of depthwise separable convolution and channel shuffle to improve the backbone network of YOLOv3 and add the SPP module to the last layer of the network to improve the expressive ability of feature maps. Then we use scale-up module to further optimize the FPN network of YOLOv3. The experimental results show that the mAP of different license plate detection in complex traffic scenes proposed in this paper can reach 0.9582, and has higher real-time performance than the original YOLOv3.

In the future work, we hope to further improve the speed and accuracy of the algorithm by creating a heterogeneous model combining YOLOv3 and image defogging algorithm.

5. Acknowledgements

This work is financially supported by National Natural Science Foundation of China (No.61872423), Industry Prospective Primary Research & Development Plan of Jiangsu Province (No.BE2017111), the Scientific Research Foundation of the Higher Education Institutions of Jiangsu Province (No.19KJA180006), the Postgraduate Research & Practice Innovation Program of Jiangsu Province (No. KYCX18_0912).

References

[1] V.J. Montero and Y.J. Jeong, “Development of license plate recognition on complex scene with plate-style classification and confidence scoring based on km,” IEICE Trans. Inf. & Syst., vol. E101.D, no.12, pp. 3181–3189, Dec. 2018.
[2] S.W. Park, S.C. Hwang, and J.W. Park, “The extraction of vehicle license plate region using edge directional properties of wavelet sub-band,” IEICE Trans. Inf. & Syst., vol. 86, no. 3, pp. 664–669, March 2003.
[3] T. Ying, L. Xin, and L. Wanxiang, “License plate detection and localization in complex scenes based on deep learning,” in 2018 Chinese Control And Decision Conference (CCDC), pp. 6569–6574, June 2018.
[4] Y. Yu, Q. Zhang, H. Wu, and Z. Jiao, “License plate location based on combination of deep learning and feature fusion,” in 2019 10th Int. Conf. Information Technology in Medicine and Education (ITME), pp. 684–687, 2019.
[5] H. Shin, H.R. Roth, M. Gao, L. Lu, Z. Xu, I. Nogues, J. Yao, D. Mollura, and R.M. Summers, “Deep convolutional neural networks for computer-aided detection: Cnn architectures, dataset characteristics and transfer learning,” IEEE Trans. Medical Imaging, vol. 35, no. 5, pp. 1285–1298, May 2016.
[6] R. Girshick, “Fast r-cnn,” in 2015 IEEE Int. Conf. Comput. Vis. (ICCV), pp. 1440–1448, 2015.
[7] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 39, no. 6, pp. 1137–1149, June 2017.
[8] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A.C. Berg, “Ssd: Single shot multibox detector,” Computer Vision–ECCV 2016, Lecture Notes in Computer Science, vol. 9905, Springer, Cham, 2016.
[9] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in 2016 IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), pp. 779–788, 2016.
[10] J. Redmon and A. Farhadi, “Yolo9000: Better, faster, stronger,” in 2017 IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), pp. 6517–6525, 2017.
[11] J. Redmon and A. Farhadi, “Yolov3: An incremental improvement,” arXiv e-prints, 2018.
[12] A. Neubeck and L. Van Gool, “Efficient non-maximum suppression,” in 18th Int. Conf. Pattern Recognit. (ICPR’06), vol. 3, pp. 850–855, 2006.
[13] F. Chollet, “Xception: Deep learning with depthwise separable convolutions,” in 2017 IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), pp. 1800–1807, 2017.
[14] K. He, X. Zhang, S. Ren, and J. Sun, “Spatial pyramid pooling in deep convolutional networks for visual recognition,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 37, no. 9, pp. 1904–1916, Sept. 2015.
[15] P. Zhou, B. Ni, C. Geng, J. Hu, and Y. Xu, “Scale-transferrable object detection,” in 2018 IEEE/CVF Conference on Comput. Vis. Pattern Recognit., pp. 528–537, 2018.
Yuchao Sun received the post graduation degree in information network from the Nanjing University of Posts and Telecommunication, Nanjing, China. His current research interests include Internet of Vehicle, Computer Vision and Image defogging.

Qiao Peng received the post graduation degree in electronic information from the Nanjing University of Posts and Telecommunication, Nanjing, China. His current research interests include Internet of Vehicle, Computer Vision and Image defogging.

Dengyin Zhang received the B.S. degree, M.S. degree and Ph.D. degree in Nanjing University of Posts and Telecommunications, Nanjing, China, in 1986, 1989 and 2004 respectively. He is currently a Professor of the School of Internet of Things, Nanjing University of Posts and Telecommunications, Nanjing, China. He was in Digital Media Lab at Umea University in Sweden as a visiting scholar from 2007 to 2008. His research interests include signal and information processing, networking technique, and information security.