Two-Stage YOLOv2 for Accurate License-Plate Detection in Complex Scenes

Shohei Yonetsu
Graduate School of Information Sci. and Eng. Ritsumeikan University
Shiga, Japan
is0242se@ed.ritsumei.ac.jp

Yutaro Iwamoto
Graduate School of Information Sci. and Eng. Ritsumeikan University
Shiga, Japan
yiwamoto@fc.ritsumei.ac.jp

Yen Wei Chen
Graduate School of Information Sci. and Eng. Ritsumeikan University
Shiga, Japan
chen@is.ritsumei.ac.jp

***Abstract*—License-plate detection is an important preprocessing step in the vehicle license-plate recognition system, which is one of the key components of intelligent transport systems (ITSs). In this paper, we proposed a two-stage YOLOv2 method for accurate license-plate detection. The first stage detects cars and then the second stage detects the license-plate in the detected car region. The proposed method can detect license-plates even in complex scenes including nighttime scenes, blurry images, various sizes of cars, and various other objects. Another contribution of this work is that we constructed a database of Japanese license-plates. There is no public database of Japanese license-plate. Our experimental results have demonstrated that our proposed method improves the detection accuracy over the original YOLOv2.**

***Index Terms*—deep learning, detection, YOLOv2, license plate**

I. Introduction

Automatic license-plate recognition (ALPR) is one of the key components of intelligent transport systems (ITSs) for purpose of facilitating the surveillance, law enforcement, access control, etc [1]. Automatic license-plate detection (ALPD) is the fundamental component of ALPR. The performance of ALPD largely determines the overall accuracy of ALPR, thereby also affecting ITS. Many methods have been proposed for ALPD [2]–[5]. Though they are powerful and useful for ALPD, most would work only under specific conditions such as fixed illumination, simple backgrounds, or a single license-plate in the image. On the other hand, the state-of-the-art detection systems such as YOLOv2 [6] and Faster R-CNN [7], based on deep learning, have been developed for object detection. YOLOv2 has been applied to automatic license-plate detection and recognition [8]. But these methods are limited by the inability to detect small objects. In this paper, we proposed a two-stage YOLOv2 method for accurate ALPR. The first YOLOv2 is used to detect cars and then the second YOLOv2 is used to detect the license-plate in the detected car region. The proposed method can detect license-plates even in complex scenes including nighttime scenes, blurry images, various size of cars, and various other objects. We also constructed a database of Japanese vehicle license-plates to train our proposed model for the purpose of real applications in Japan. The remainder of this paper is follows. The rest of this paper is organized as follows. In Section 2, we describe the conventional YOLOv2. In Section 3, we describe the two-stage YOLOv2 and the utilized dataset. In Section 4, we describe the experimental results. In Section 5, we present our conclusions.

II. YOLOv2

Deep convolutional neural networks have been successfully applied to object detection. YOLOv2 [6], also known as YOLO9000(due to its ability to detect more than 9000 object categories) is an improve version of YOLO [9], which is a state-of-the-art, real-time object detection method based on deep learning. YOLOv2 achieves 76.8mAP at 67 FPS on PASCAL VOC 2007. YOLOv2 uses a fully convolutional network (FCN) that consists of 22 convolution layers and 5 pooling layers. By using the FCN for feature extractions, higher detection accuracy has been achieved than other deep learning based detection methods. The network of YOLOv2 is shown in Fig.1.

YOLOv2 divides the input image into S $\times$ S regions and predicts B bounding boxes (anchors) and C conditional class probabilities (one per class) for each region. Each bounding box contains five elements: (x, y, w, h, c), of which x and y are the center coordinates, w is the width, h is the height, and c is the confidence score(how likely the box contains an object). Thus, the YOLOv2 outputs S $\times$ S $\times$ (B $\times$ 5 + C) channels. Loss functions are shown in below.

$$
\begin{aligned}
Loss = \\
&\lambda_{obj}^{coord} \sum_{i}^{S^2} \sum_{j}^{B} l_{ij}^{responsible\_obj} \{(x_{ij}^{pred} - x_{ij}^{obj})^2 \\
&+ (y_{ij}^{pred} - y_{ij}^{obj})^2 + (w_{ij}^{pred} - w_{ij}^{obj})^2 + (h_{ij}^{pred} - h_{ij}^{obj})^2\} \\
&+ \lambda_{noobj}^{coord} \sum_{i}^{S^2} \sum_{j}^{B} l_{ij}^{no\_responsible\_obj} \{(x_{ij}^{pred} - x_{ij}^{anchor\_center})^2 \\
&+ (y_{ij}^{pred} - y_{ij}^{anchor\_center})^2 + (w_{ij}^{pred} - w_{ij}^{anchor\_default})^2 \\
&+ (h_{ij}^{pred} - h_{ij}^{anchor\_default})^2\} \\
&+ \lambda_{obj}^{conf} \sum_{i}^{S^2} \sum_{j}^{B} l_{ij}^{responsible\_obj} \{conf_{ij}^{pred} - iou(box_{ij}^{pret}, box_{ij}^{truth})\}^2 \\
&+ \lambda_{noobj}^{conf} \sum_{i}^{S^2} \sum_{j}^{B} l_{ij}^{no\_responsible\_obj} \{conf_{ij}^{pred} - 0\}^2 \\
&+ \sum_{i}^{S^2} \sum_{j}^{B} l_{ij}^{responsible\_obj} \{P_{ij}^{pred}(c) - P_{ij}^{truth}(c)\}^2
\end{aligned}
\tag{1}
$$

In the above formula, the following values are used.

$$
\begin{aligned}
&x_{ij}^{anchor\_center} = y_{ij}^{anchor\_center} = 0.5 \\
&w_{ij}^{anchor\_default} = h_{ij}^{anchor\_default} = 1.0 \\
&\lambda_{obj}^{coord} = 1.0 \\
&\lambda_{noobj}^{coord} = 0.1 \\
&\lambda_{obj}^{conf} = 10.0 \\
&\lambda_{noobj}^{conf} = \begin{cases} 0.0(iou > threshold) \\ 0.1(iou < threshold) \end{cases}
\end{aligned} \tag{2}
$$

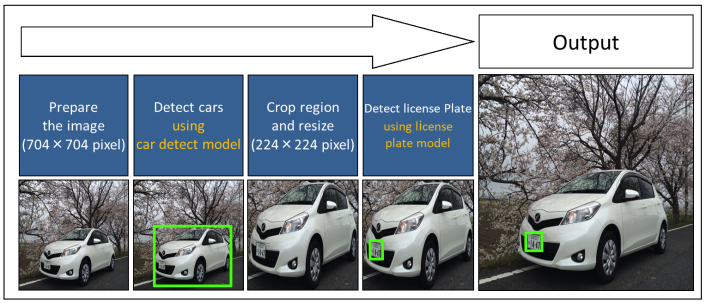

Fig. 2. Procedure of the proposed two-stage YOLOv2

Type	Filters	Size	Output
Convolutional	32	3 × 3	
MaxPooling		1/2	128 × 128
Convolutional	64	3 × 3	
MaxPooling		1/2	64 × 64
Convolutional	128	3 × 3	
Convolutional	64	1 × 1	
Convolutional	128	3 × 3	
MaxPooling		1/2	32 × 32
Convolutional	256	3 × 3	
Convolutional	128	1 × 1	
Convolutional	256	3 × 3	
MaxPooling		1/2	16 × 16
Convolutional	512	3 × 3	
Convolutional	256	1 × 1	
Convolutional	512	3 × 3	
Convolutional	256	1 × 1	
Convolutional	512	3 × 3	
MaxPooling		1/2	8 × 8
Convolutional	1024	3 × 3	
Convolutional	512	1 × 1	
Convolutional	1024	3 × 3	
Convolutional	512	1 × 1	
Convolutional	1024	3 × 3	
——			
Convolutional	1024	3 × 3	
Convolutional	1024	3 × 3	
Concatonation	3072		
Convolutional	1024	3 × 3	S × S × (B × 5+C)
Detection			

Fig. 1. Network of YOLOv2

III. Our Approach

YOLOv2 can detect medium-sized to large-sized objects, but struggles with small ones. The area of a license-plate is usually less than 1% in a scene image. (assuming the image size of 704 pixels × 704 pixels, with the area of the license-plate region occupying a maximum about 45 pixels × 40 pixel at most) Therefore, the detection accuracy is considered to be exceedingly low just using YOLOv2. Also, YOLOv2 tends to mis-detect signboards as the license-plates, which is a kind of false detection. Therefore, a method is needed to overcome both limitations.

A. Two-stage YOLOv2

We propose a two-stage YOLOv2 method, as shown in Fig.2, for accurate license-plate detection even for a complex scene images. We first use a YOLOv2 to detect cars. Since the size of a car is larger than that of a license plate, the detection rate for a car using YOLOv2 is high. Then we use the second YOLOv2 network to detect the license-plate of in the detected car region(the bounding box of the car). Since the input image for license-plate detection is limited to the car region, the input image is not a complex image and the size of the license-plate becomes relatively larger. Thus, we can significantly reduce the false detections and improve the detection rate.

B. Models and Dataset

Our proposed method consists of two models, of which the first (car detection) is for detection of cars in the complex scene images and the second one (license-plate detection) is for the license plates in the car regions.

The car detection model was trained with a Microsoft-COCO dataset [10], which has 80 object categories, of which five categories were related to vehicles (bicycles, cars, motorcycles, buses, trucks). ”Car”, ”bus” and ”track”, which have license plates, were included in the car category (positive samples), while ”bicycle” and ”motorcycle” as well as the other categories were excluded (negative samples). If the detected object category was ”car”, we used the second model to detect the license-plate in the car region (the bounding box of the car). For license-plate detection (the second model), our targets were the three colors used for license plates ”white”, ”yellow” and ”green” that are used generally in Japan. As there is no public Japanese license plate database for training, we construct an original Japanese license plate database including 747 annotated Japanese license plates (white plates:416, yellow plates: 172, green plates: 159). Some examples are shown in Fig.3 It should be noted that other colors, such as pink and black are not included because such color plates are rarely used.

IV. Experiments

To validate the effectiveness of our proposed method, we experimented with real complex street scene images with that include nighttime scenes, blurry images, various sizes of cars, and various other objects. Our experimental environment consisted of Ubuntu 14.04 LTS, CUDA=8.0 and CuDNN=6.0.21 on. PC equipped. PC equipped with GeForce GTX TIAN X and Intel Core i7 5960-X. The framework we used was Chainer 1.24.0. [11].

Fig. 3. Examples of our Japanese license plate database

A. Training

1) Car detection model: The car detection model was trained using an SGD with a weight decay of 0.005 and momentum of 0.9. The learning rate is set to 0.001 for the first 10 iterations, 0.0001 for 11-60 iterations, 0.00001 for 61-90 iterations, and 0.000001 for 91-160 iterations. The batch size was 4 and the training image size was set to one of the values in{320, 352, 384, ..., 608} multiples of 32 for each batch.

We used the microsoft-COCO datasets to train the car detection model (the first stage). Data argumentation by ”contrast adjustment” and ”horizontal flip” was used to expand the training dataset.

2) License-plate detection model: The license-plate detection model was also trained using a SGD with a weight decay of 0.005 and momentum of 0.9. The learning rate is set to 0.00001 for 800 iterations. The batch size is 4 and the training image size is set to one of the values in{64, 96, 128, ..., 448} multiples of 32 for each batch.

We used our original license plate dataset as mentioned in previous section to train the second model. Data argumentation was used and we expanded the training images to 7880 images (about 10 times of the original dataset) by using ”contrast adjustment (0.2 and 2.0)”, ”horizontal flip ”and ”noise ”. It should be noted that ”vertical flip ” was not adopted because it will reduce the detection rate. Examples of data augmentations are shown in Fig.4

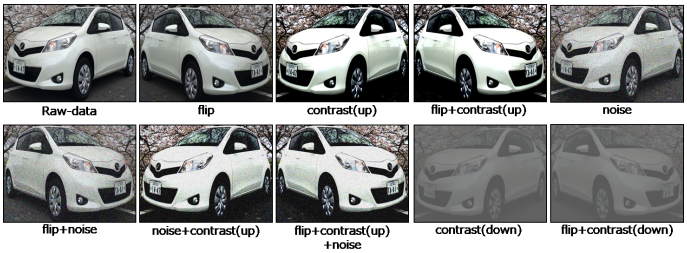

Fig. 4. Examples of data augmentation

B. Results

For the evaluation, we used 160 real street scene images (704 pixels × 704 pixels) with 300 annotated license-plates. Among them, 114 images with 200 annotated license-plates have various sizes and readable numbers (minimum size is about 15 pixels × 15 pixels), of which 48 images are taken in the daytime and clear weather, and 66 images are taken at nighttime including rainy. The other 46 images with 100 annotated license plates have non-readable numbers because of blur and distant distance. The detection criterion is defined as more than 30% of Intersection over Union (IoU) of the predicted results and ground truth. The detection results for various complex scene images are shown in Fig.5. The images in the left column images are ground truth, while the middle and right column images are detection results of a conventional YOLOv2 [6], [8] and our proposed two-stage YOLOv2, respectively.

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

Fig. 5. Left column: ground truth results. Middle column: detection results of conventional YOLOv2. Right column: detection results of our proposed two-stage YOLOv2. Green and red boxes show the results of the first-stage car and the second-stage license-plate detection models, respectively. The numbers on the boxes signify the IoUs of the predicted and ground truth results.

TABLE I
COMPARISON OF OUR PROPOSED METHOD TWO-STAGE YOLOV2 WITH CONVENTIONAL YOLOV2 IN COMPLEX SCENE WITH READABLE NUMBER PLATES

Environment	CLEAR WEATHER		NIGHTFALL, NIGHT	
Metrics	Conventional YOLOv2 [8]	Two-stage YOLOv2	Conventional YOLOv2 [8]	Two-stage YOLOv2
LP detection rate (car detection rate)	58% (-)	87% (99%)	44% (-)	74% (85%)
False detection	7	4	4	3
Average IoU	0.38	0.65	0.28	0.51
Average precision	0.41	0.77	0.32	0.51
Average recall	0.49	0.70	0.37	0.56
Average F number	0.44	0.73	0.33	0.59

TABLE II
COMPARISON OF OUR PROPOSED METHOD TWO-STAGE YOLOV2 WITH CONVENTIONAL YOLOV2 IN COMPLEX SCENE WITH NON-READABLE NUMBER PLATES

Environment	Non-readable License Plate	
Metrics	Conventional YOLOv2 [8]	Two-stage YOLOv2
LP detection rate (car detection rate)	1% (-)	53% (87%)
False detection	4	3
Average IoU	0.005	0.33
Average precision	0.005	0.44
Average recall	0.007	0.37
Average F number	0.006	0.40

The latter was able to detect the license-plates of both distant and near cars accurately, even in complex scenes, such as nighttime and blurry images, as shown in Fig.5 As shown in Fig.5(c), the conventional YOLOv2 mis-detected a signboard as a license-plate, while the two-stage YOLOv2 did not detected such false positives.

Fig.5(d) are results for cars with distant distance. It can be seen that the conventional YOLOv2 can not detect the distant license-plate (middle), while our proposed two-stage YOLOv2 can detect it accurately (right).

The quantitative comparisons between the proposed two-stage YOLOv2 and the conventional YOLOv2 (one-stage YOLOv2) are shown in Table I and II. As shown in Table I and II, significant improvements were achieved with our two-stage YOLOv2. Especially, the conventional YOLOv2 can not detect the non-readable license-plates, while our proposed method still can detect it though the detection accuracy is not high enough.

V. CONCLUSION

We proposed a two-stage YOLOv2 for accurate license plate detection in complex scene images. We also constructed an original Japanese license plate database for real applications in Japan. Experiments with our database demonstrated that proposed two-stage YOLOv2 did improve the detection rate and significantly reduce the number of false positives even in complex scene images, which included nighttime scenes, blurry images, various sizes of cars and various other objects.

For future work, we will develop a detection method incorporating with a super-resolution technology to improve the detection rate for distant license plates (non-readable license-plates). We will also attempt to reduce the computation time.

REFERENCES

[1] C.N. Anagnostopoulos, et al. “A license plate-recognition algorithm for intelligent transportation system applications”, IEEE Trans. On Intelligent Transportation Systems, Vol.7, pp.377-392, 2006.

[2] Y. Yuan, et al. “A robust and efficient approach to license plate detection”, IEEE Trans. On Image Processing, Vol.26, pp.1102-1114, 2017.

[3] K.-H. Lin, et al. “Robust license plate detection using image saliency”, In 2010 17th IEEE International Conference on Image Processing (ICIP2010), pp.3945-3948, 2010.

[4] K. Deb, and K.-H. Jo, “A vehicle license plate detection method for intelligent transportation system applications”, Cybernetics and Systems: An International Journal, Vol.40, pp.689-705, 2009.

[5] S. Z. Masood, et. al., “License plate detection and recognition using deeply learned convolutional neural networks”, arXiv: 1703.07330, 2017.

[6] J. Redmon, and A. Farhadi, “YOLO9000: Better, Faster, Stronger”, In 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 6517-6525, 2017.

[7] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: towards real-time object detection with region proposal networks”, IEEE transactions on pattern analysis and machine intelligence, Vol.39, pp.1137- 1149, 2017.

[8] R. Laroca, “A robust real-time automatic license plate recognition based on the YOLO detector”, arXiv:1802.09567, 2018.

[9] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, “You only look once: Unified, real-time object detection”, In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 779- 788, 2016.

[10] T. Y. Lin, et al. “Microsoft coco: Common objects in context”, In European conference on computer vision, pp. 740-755, 2014..

[11] A. Chainer: A flexible framework for neural networks,” https://chainer.org”