Journal of Imaging Science and Technology® 66(4): 040404-1–040404-9, 2022.

Semi-supervised Learning for YOLOv4 Object Detection in License Plate Recognition System

Cheng-Lung Chang
Department of Information and Telecommunications Engineering, Ming Chuan University, Taoyuan, Taiwan

Pi-Jhong Chen
Undergraduate Program in College of Electrical Engineering and Computer Science, Chung Yuan Christian University, Taiwan

Ching-Yi Chen
Department of Information and Telecommunications Engineering, Ming Chuan University, Taoyuan, Taiwan
E-mail: chingyi@mail.mcu.edu.tw

Abstract. *Deep learning is a research field with great application potential. However, training a deep learning model requires a large amount of paired data, which is time-consuming and expensive for real-world applications. In light of this problem, researchers have conducted methods to overcome the lack of training data, including data augmentation, data enhancement, transfer learning, and semi-supervised learning (SSL). Based on this concept, in this study, we conduct a self-training method to fine-tune a YOLOv4 license plate detection framework that requires only a limited amount of training datasets. The framework requires only a small amount of labeled data to complete the training process. Furthermore, this study utilizes Tesseract optical character recognition (OCR) on the detected license plates to achieve better performance. We propose a high-performance license plate recognition system for detecting different tilting angles and complex backgrounds.* © *2022 Society for Imaging Science and Technology.*
[DOI: 10.2352/J.ImagingSci.Technol.2022.66.4.040404]

1. INTRODUCTION

Automatic license plate recognition is one of the core technologies of the smart city concept. Applications can be developed through real-time object detection and image analysis when the city's traffic construction consists of networked monitors, including redline illegal parking billing, roadside parking charges, stolen car query system, etc. For example, on-street parking fees have always been one of the most important sources of government revenues, and on-street parking management is also closely related to the city's traffic congestion and vehicle emissions. However, to achieve automatic on-street parking fee management, there are some challenges to be solved. For instance, the variety of different street scenes, different heights and widths of vehicles, the short distance between two consecutive parking spaces, and parking violations being often on the streets. To achieve billing or charging with automatic license plate recognition technology, it is necessary to overcome the problems such as the extreme tilt of the captured image, high viewing angle, and small license plate area in the image. Compared with the license plate recognition task of a fixed camera at the gate of a parking lot, the license plate recognition of on-street parking is more challenging.

The license plate recognition system can be divided into two stages: license plate detection and character recognition. For license plate detection, the gray-scale intensity change of the license plate area should be the strongest in the entire vehicle image, as well as the shape of it has a fixed aspect ratio. Therefore, many researchers use edge-based methods to detect the location of license plates. Refs. [1] and [2] combine edge statistics and mathematical morphology to find the rectangular area that may be a license plate. However, although the edge-based methods have high execution speed, it is difficult to apply to complex images. On the other hand, the color-based method [3, 4] leverages the color feature of the license plate as the basis for the segmentation of the license plate and the background. Despite its simplicity and effectiveness, it may not be applicable in real-world scenarios since the color changes with different lighting conditions. The symmelet-based method [5] and texture-based method [6, 7] are also used to extract important features of license plates. Ref. [8] introduced a license plate detection method based on Gabor conversion. Although the performances of these methods are encouraging, being both time consuming and computational consuming makes it unsuitable for real-time applications. Moreover, Ref. [9] uses Haar-like features, which are invariant to the size, color, and brightness of the license plate, to detect objects. However, the performance of their methods are still very limited. In addition to the methods mentioned above, the template-matching method can also be effective for license plate detection tasks. However, it is not flexible and expandable for input images of different sizes [10].

Traditional object detection algorithms are usually divided into two steps: feature extraction and object classification. Designing such algorithms requires domain knowledge and real-world experiences. Moreover, a lot of tuning work is required to achieve better results. With the rise of deep learning, which is a powerful tool that learns knowledge by

Received Dec. 25, 2021; accepted for publication Mar. 29, 2022; published online Apr. 26, 2022. Associate Editor: Jia-Shing Sheu.
1062-3701/2022/66(4)/040404/9/$25.00

being trained with a large amount of training data, many studies have recently proposed using convolutional neural networks (CNN) for the location and feature extraction of license plates, which outperforms traditional license plate detection methods [11, 12]. Due to CNN's powerful feature-extracting capabilities, capturing the local details and overall features of the image layer by layer through CNN has become a popular method for object detection. CNN is often combined with candidate boxes and classifiers to perform object detection. YOLOv4 is a state-of-the-art object detector [13] based on the innovative CNN architecture. It is simple, fast, and robust background, which has large improvement on both accuracy and computing time.

Although CNN-based deep neural network (DNN) models are the mainstream research direction of image recognition and object detection, it requires a large number of labeled training data for developing deep learning applications. The training data need to be representative and cover various changes, otherwise, the trained model may be biased or result in poor performance. Due to the drawbacks of being time-consuming and costly to label a large amount of data, many researchers have devoted themselves to proposing methods, such as weakly-supervised learning [14], mix-supervised learning, or using SSL [15], which is trained with a small amount of labeled data and a large amount of unlabeled data, that can solve the lack of training data. The methods mentioned above have gained lots of attention in recent years. Moreover, SSL combined with data augmentation can further improve the generalization and robustness of DNN models [16, 17]. The geometric and color space augmentation of the input image is a very important factor to improve the generalization ability [18]. However, most data augmentation techniques are often used in image classification tasks, while object detection and other tasks could also be improved with the augmented data. The complexity of data augmentation for object detection is much higher than that of the image classification problem since the geometric transformation of the image data will cause the position of the bounding boxes to also change, making the originally labeled data no longer applicable. In other words, although using data augmentation does not increase the complexity or slow down the speed of any model architecture, it is more difficult to design a reasonable data augmentation strategy suitable for object detection tasks.

Based on the above reasons, in this study, we propose a method that combines self-training and a data augmentation method suitable for object detection via YOLOv4 license plate detection architecture that requires only a limited training data set. Self-training is a wrapper method for SSL algorithm, which can reduce the need for labeled data during the training phase of DNN model. For data augmentation, since the bounding boxes of the labeled training data are of different numbers and sizes, it is difficult to match the bounding box labeled in the original image if the image is changed through disturbances such as rotation or distortion. Thus, this paper uses data augmentation methods such as Mosaic [13], Colour [19], and SMDWT [20] to maintain the corresponding relationship between the original image and the predicted bounding box coordinates of the disturbing image. To further identify the license plate numbers, we also process the detection results into the subsequent Tesseract OCR module [21] to create a system that can be applied to various license plate recognition systems with complex background.

2. RELATED APPROACHES

2.1 *Self-training Method*

Self-training is an effective way to strengthen the existing classifier model by using unlabeled data. It leverages a trained small data classifier model to give the unlabeled data soft labels to increase the training data [15]. For a labeled data set $\{(x^r, \hat{y}^r)\}_{r=1}^{R}$ and an unlabeled dataset $\{(x^u)\}_{u=R+1}^{R+U}$, the amount of data without the label should be much larger than of with label, that is, $U \gg R$. First, the labeled data is used to train a classifier model f^*. After that, f^* is applied to the unlabeled dataset in order to get $\{(x^u, y^u)\}_{u=R+1}^{R+U}$. Then, a part of the data can be removed from the unmarked data set, and added to the marked data set based on the confidence level or weight. Finally, the model can be retrained, and the

Self-training algorithm

Initialize:

Given: labelled dataset $= \{(x^r, \hat{y}^r)\}_{r=1}^{R}$, unlabeled dataset $= \{(x^u)\}_{u=R+1}^{R+U}$

While stopping criteria not met **do**

Train model f^* from labelled dataset

Apply f^* to the unlabeled dataset, and obtain $\{(x^u, y^u)\}_{u=R+1}^{R+U}$

Select confidence sample $\{(x^{conf}, y^{conf})\}$, $\{(x^{conf}, y^{conf})\} \in \{(x^u, y^u)\}_{u=R+1}^{R+U}$

Remove selected data from unlabeled dataset

Add the selected data into the labeled dataset

End while

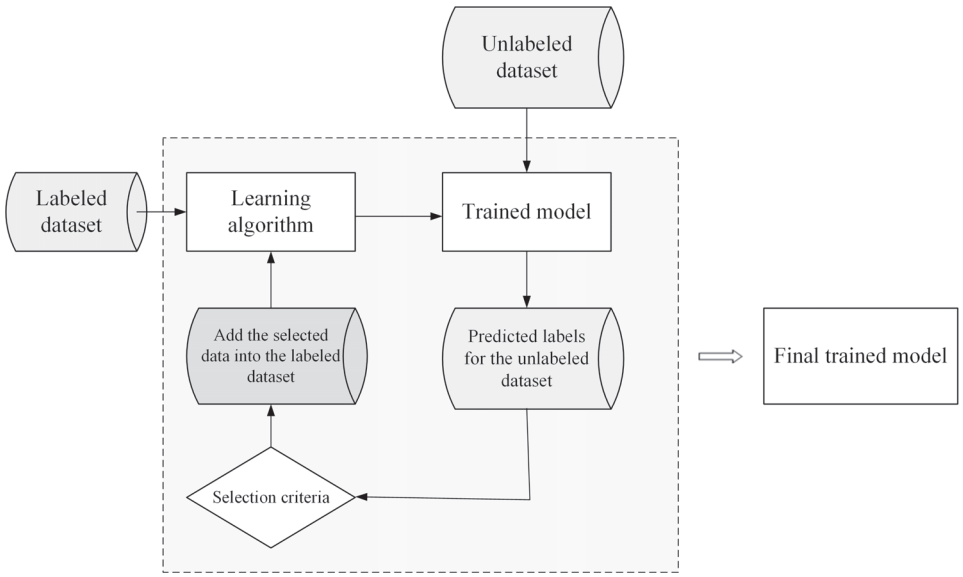

Figure 1. The flowchart of the self-training algorithm.

above steps are repeated until it converges. As shown in Figure 1, this method of using the trained classifier model to mark unlabeled data is also called pseudo-labeling, which effectively solves the problem of insufficient labeled data when training a DNN model.

2.2 *YOLOv4 Object Detection*

A YOLO algorithm views object detection as a regression problem. From image input to output prediction, this method is implemented by only a single CNN model to predict the coordinates of multiple bounding boxes and calculates the probability of the object category for each bounding box. Such end-to-end training method increases the execution speed while solving problems like the traditional object detection frameworks. The process of YOLO is to normalize an image and then slice it into $S \times S$ grid cells. If the center of an object falls within a certain grid, the grid is responsible for the detection task of the object. During training and testing, each grid performs the prediction of B bounding boxes and the probability of the category to which it belongs. YOLOv2 removed the FC layer of the YOLOv1 network and introduced the anchor box to predict the bounding box. Although YOLOv3 does not have revolutionary innovations, it refers to the concepts of networks such as ResNet and feature pyramid network (FPN) to further improve performance.

Researchers including Wang et al. published the Cross-stage partial network (CSPNet) [22] with low hardware resource requirements and high accuracy in 2019. This architecture increases gradient combinations and maximizes the diversity of gradients, which can effectively solve the problem of gradient vanishing of a convolutional network. The YOLOv4 algorithm uses CSPDarkNet53 as the backbone to strengthen the learning ability of the convolutional network, and the neck part adds the spatial pyramid pooling (SPP) and path aggregation network (PANet). The SPP block is used to reinforce the receptive field, and PANet replaces the FPN of YOLOv3 to do parameter aggregation in different levels of detectors. When the input image is scaled to a size of 416×416 and sampled by CSPDarkNet53, three feature maps of different scales will be obtained. The last feature map will be subjected to 5×5, 9×9, and 13×13 maximum pooling operations, and then merge the results to get a 13×13 feature map. These three feature maps of different scales are sent to PANet for regression and classification operations, and the bounding box coordinates and confidence level of the target object can be obtained. The YOLOv4 algorithm achieves the best balance between accuracy and speed, which obtained preferable performance than that of other R-CNN networks [13]. The main network structure of YOLOv4 is shown in Figure 2.

2.3 *Data Augmentation*

When the size of the training dataset is insufficient to train a DNN model, unlike SSL, which changes the learning strategy to improve performance, data augmentation directly solves the problem from the root of the problem by generating more training data from existing training data. Since preparing similar samples for training in advance for all test samples that need to be inferred is costly and hard to obtain, data augmentation techniques improve the training dataset by expanding it, which also increases the diversity and robustness of training samples. These techniques improve and scale up the training set to expose the model to unseen scenarios, allowing DNN models to learn from situations with more diversities.

2.3.1 *Mosaic Data Augmentation*

Geometric augmentation includes random scaling, cropping, flipping, and rotating, etc. However, these types of augmentation are not easy to be applied to the training tasks of object detection models, since the image generated by geometric

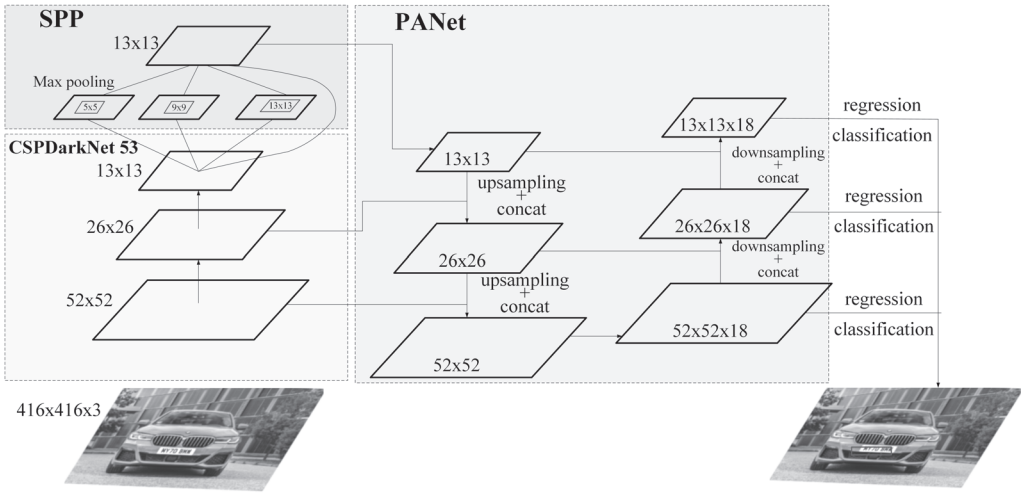

Figure 2. Main structure of YOLOv4 [23].

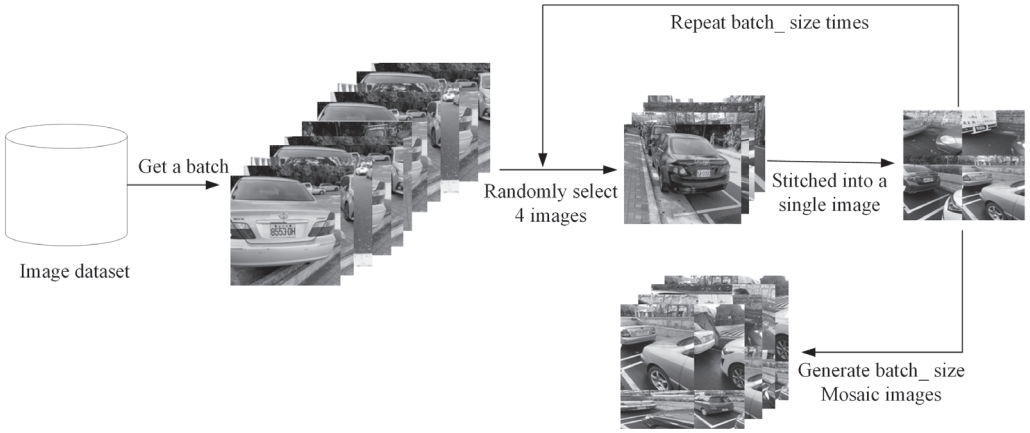

Figure 3. Mosaic data augmentation.

distortion will cause the position of the target object to change, thus affecting the coordinates of its bounding box. Different from traditional data augmentation methods, which involves distorting, flipping, or changing the tone of an image, Mosaic data augmentation crops four image crops randomly and then stitch them into one image as additional training data. As shown in Figure 3. Figure 4 shows some examples of mosaic data augmentation on vehicle images.

2.3.2 *Colour Data Augmentation*

Colour data augmentation including changing the brightness, contrast, saturation, HSV jittering, color channel dropping uses randomly distorted color channels without impacting the locations of the bounding boxes. Since there will have a light deviation in the actual image, it is necessary to enhance the light during training. Figure 5 shows the result of Colour data augmentation of the license plate image.

2.3.3 *2D-SMDWT Data Augmentation*

The 2D-symmetric mask wavelet-based discrete wavelet transform (2D-SMDWT) architecture used in this system was proposed by Hsia et al. [20]. It retains the advantages of lifting-based 2D-DWT and can increase the computing speed, also reducing the overall computing complexity at the same time. The overall architecture is shown in Figure 6. Through derivation and simplification, 2D-SMDWT can obtain four subbands through the calculation of four masks. Since the low-low (LL) band of the image can remove some high-frequency noise and retain important characteristic information, in this study, we only capture the LL band feature for data augmentation. Figure 7 shows an example of data increase using the LL band.

2.4 *Tesseract OCR Module*

Tesseract is an OCR engine developed by HP Labs [21]. In 2006, Google restarted it as an open-source project. This project currently supports mainstream operating systems such as Windows, Linux, and Mac OS. The architecture of Tesseract OCR engine is shown in Figure 8. To perform the Tesseract OCR, Otsu's thresholding is used to binarize the image. After that, page layout analysis completes the extraction of text blocks in the document and divides the image into text and non-text areas. Finally, the model detects

Figure 4. Example of Mosaic data augmentation.

Figure 5. Colour data augmentation (brightness = 0.5, saturation = 0.6, contrast = 0.7, hue = 0.8).

the baselines of each line of text, then cuts the content of the text into words.

3. SELF-TRAINING BASED LICENSE PLATE RECOGNITION SYSTEM

The license plate recognition system in this research contains two parts: license plate detection and license plate character recognition. The license plate detection is implemented through YOLOv4 and uses methods such as data augmentation and self-training to improve performance. The license plate character recognition system is constructed using Tesseract OCR. The architecture is described in detail below.

3.1 *Self-training Based YOLOv4 License Plate Detection*

The main difference between YOLOv4 object detection and general image classifiers is that the output of the object detection model includes the coordinates of the object's bounding box and the probability of its category. When applying the self-training method to train the license plate detection system, the YOLOv4 model we trained with small data in advance is used to output the license plate bounding box coordinates of the unlabeled test data. Test elements, whose confidence score is greater than the threshold λ will be moved from the test data set to the labeled training data set as the augmentation data for retraining the YOLOv4 model. After repeating the above steps several times, a high-accuracy license plate detection model can be obtained, as shown in Figure 9.

3.2 *License Plate Character Recognition*

After obtaining the correct boundary frame coordinates of the license plate through YOLOv4, the result of character recognition will be output through Tesseract OCR. Finally, the license plate area and the recognized license plate number will be displayed on the vehicle image, as shown in Figure 10.

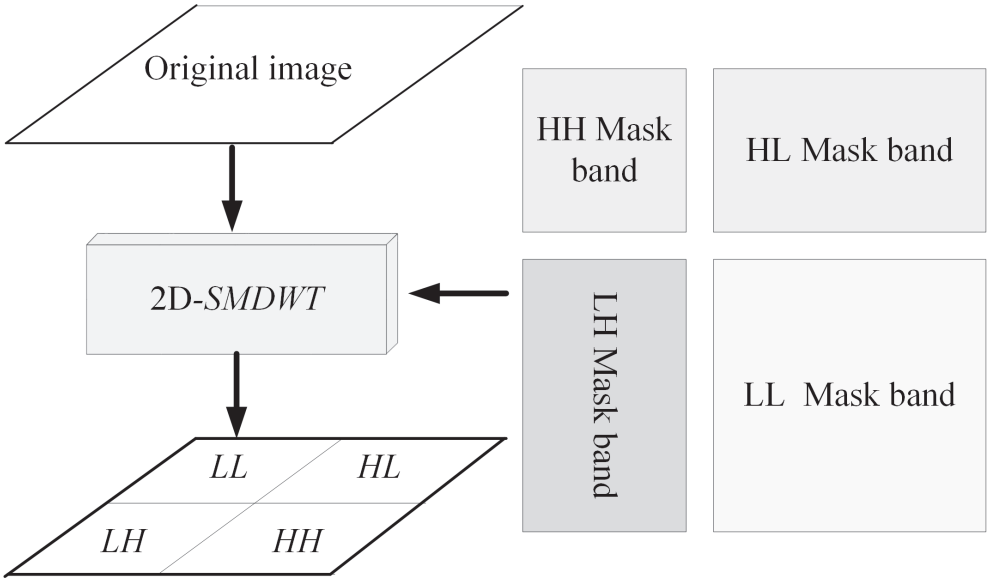

Figure 6. The architecture diagram of SMDWT.

Figure 7. Data augmentation of 2D-SMDWT.

4. EXPERIMENTAL RESULTS AND DISCUSSION

In this section, we use a small number of images to train 6 different slightly different YOLOv4 detection models, and then input 1000 unlabeled test images into the models trained by each model to analyze the difference in performance.

4.1 *Datasets Self-training Based YOLOv4 License Plate Detection Model*

The images of the vehicles used in this study were taken from typical roadsides including outdoor parking lots, under the shade of trees, roadside parking spaces, and slopes. In order

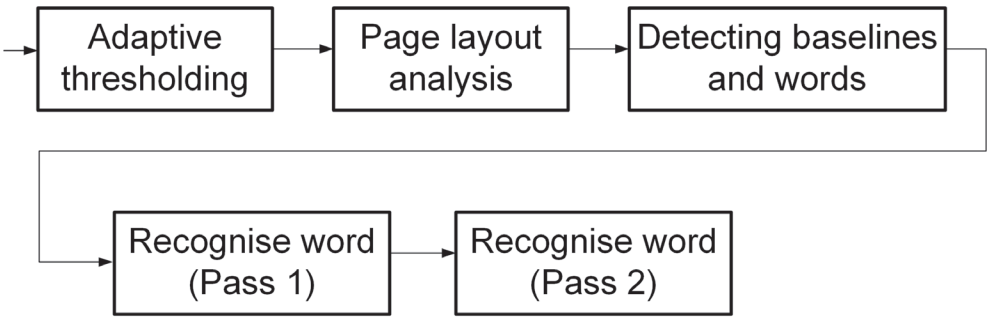

Figure 8. Engine block diagram of Tesseract OCR.

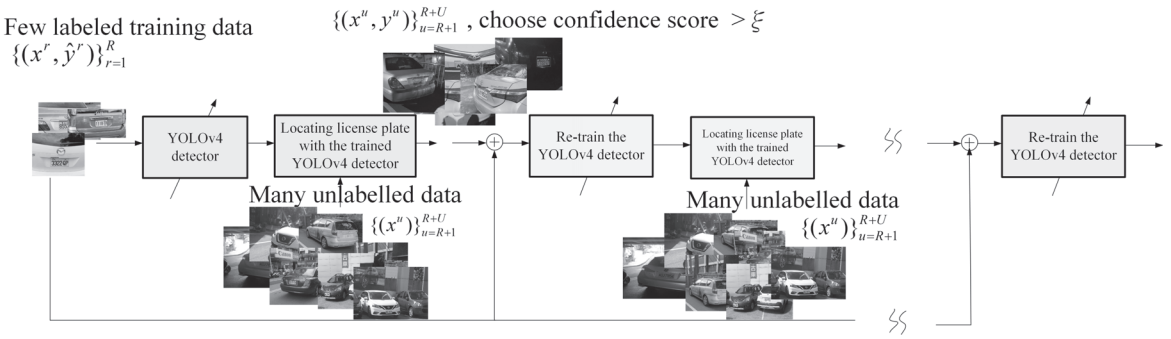

Figure 9. The architecture diagram of proposed plate detection method.

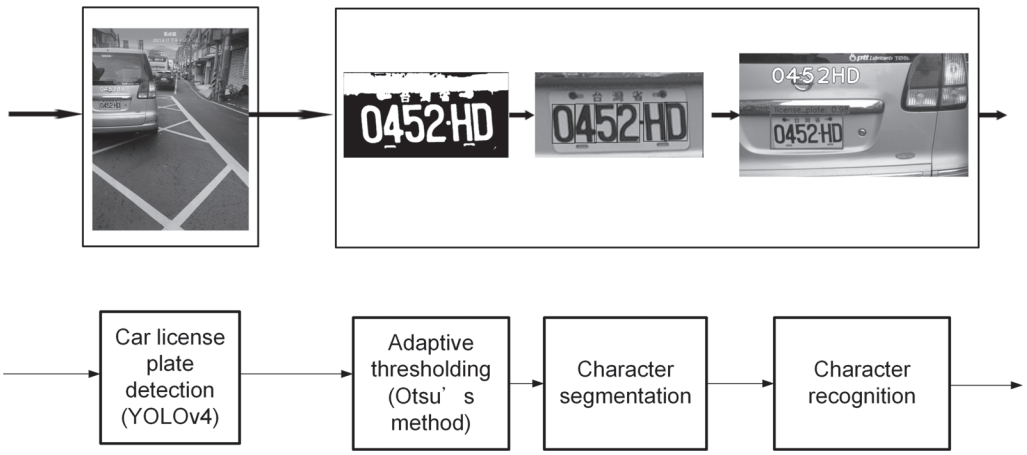

Figure 10. License plate character recognition module.

to obtain various situations in the training data, there are no restrictions on the direction and angle of shooting. The vehicle image database used in the experiment has a total of 1100 images, of which 100 are labeled images and the remaining 1000 are all unlabeled test images.

4.2 *Performance of the Self-training Based YOLOv4 License Plate Detection Model*

To evaluate the performance of different self-training based YOLOv4 license plate detection models, we conduct an ablation study by training 6 different YOLOv4 detection models using 100 labeled images. The 6 different models for ablation study are: the YOLOv4 detection model without any data augmentation, GrayScale data augmentation only, SMDWT data augmentation only, Colour data augmentation, Mosaic data augmentation, and all the data augmentation methods combined. For training details, we test 1000 unlabeled test images on the YOLOv4 detection model trained by each method to analyze the difference in performance, as shown in Table I. After obtaining the pseudo label for the unlabeled data, we use self-training to retrain the YOLOv4 model with the best mAP performance (i.e., 85.4%). As shown in Table I, our model achieves 94.6% on the detection success rate. Table II shows the statistical table of detection success and detection failure of the test samples.

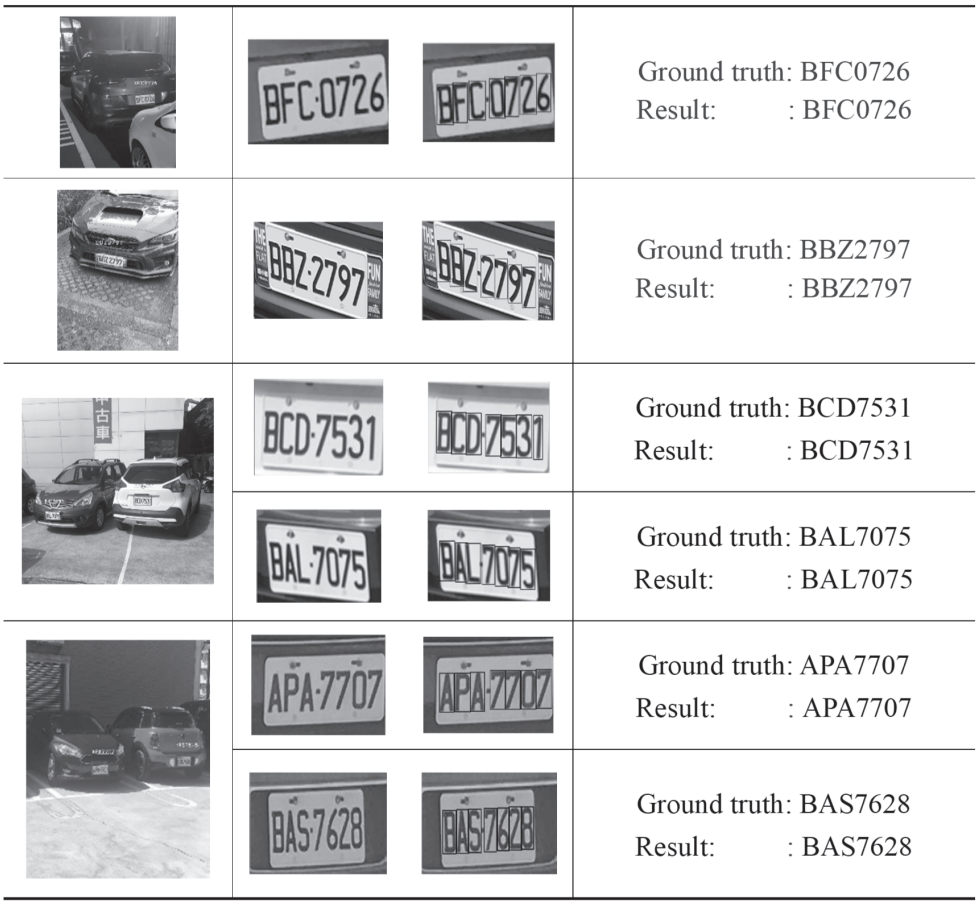

Figure 11. Successful cases of license plate positioning and character recognition.

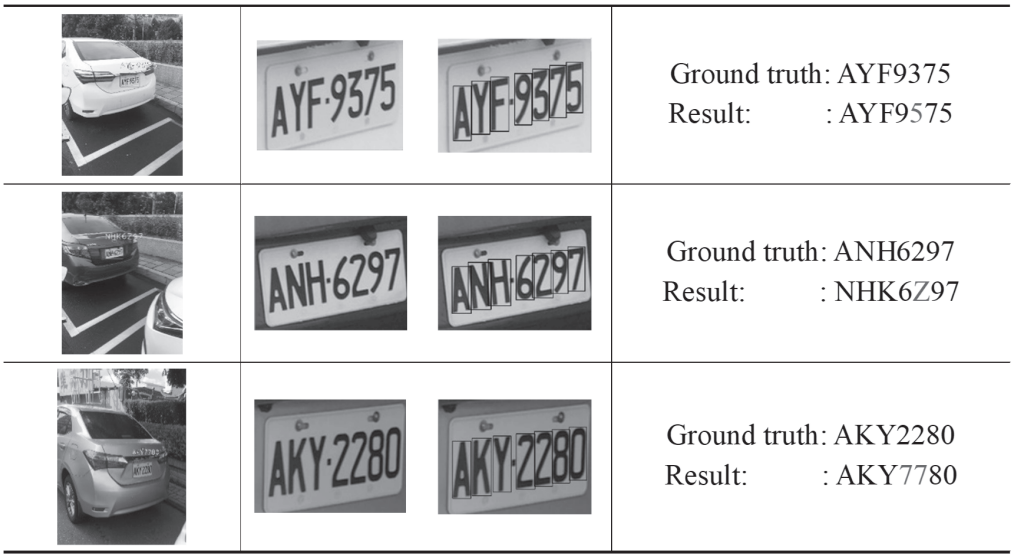

Figure 12. A case where the license plate is successfully located, but there are some errors in character recognition.

4.3 *License Plate Recognition Result*

Figures 11 and 12 shows the experimental results of some of the examples from the testing dataset detected by the proposed license plate recognition models. The four images in Fig. 11 contain front and inclined license plates with one to two vehicles in the images. According to the test results, our system successfully detects the license plates of all vehicles in these images, as well as completes the character cutting and

Table I. Training results of different Data Augmentation methods for YOLO v4 model.

	Original	GrayScale	SMDWT	Colour	Mosaic	ALL	After Self-training
mAP	71.0%	71.7%	74.0%	74.6%	79.4%	85.4%	94.6%
Precision	65.0%	72.0%	76.0%	72.0%	76.0%	83.0%	92.0%
Average IOU	45.6%	53.8%	55.4%	56.6%	61.5%	73.8%	82.5%

Table II. YOLOv4's detection success rate of test data.

	YOLOv4 model with small data	YOLOv4 model with small data + Data augmentation	Our method
Successfully detected	710 photos	854 photos	946 photos
Detection failed	290 photos	146 photos	54 photos
Detection rate	71.0%	85.4%	94.6%

license plate number recognition. The last recognized license plate number will be displayed in white characters above the license plate in the photo.

Fig. 12 shows some examples of incorrect recognition of license plate numbers. The viewing angle of the license plate in these three images has a severe distortion and is in extreme angles. Although the license plate detection model we built can still correctly detect the location of the license plate, the subsequent character recognition error occurs.

5. CONCLUSION

In this study, we proposed a license plate detection method suitable for training on limited datasets. Self-training algorithms and data augmentation methods are adapted to train the YOLOv4 detection model, which can effectively increase the size of the training dataset, and train a high-performance license plate detection model based on the generated training data. After performing object detection, we use Tesseract OCR as character recognition module. The character recognition module predicts the license plate number after receiving the image of the located license plate area. The experimental results show that the framework proposed in this study can indeed use a small amount of additional data to train an object detection model with competitive performance.

REFERENCES

[1] H. Bai and C. Liu, "A hybrid license plate extraction method based on edge statistics and morphology," *Proc. 17th Int'l. Conf. on Pattern Recognition, Cambridge, UK* (IEEE, Piscataway, NJ, 2004), pp. 831–834.

[2] D. Zheng, Y. Zhao, and J. Wang, "An efficient method of license plate location," Pattern Recognit. Lett. **26**, 2431–2438 (2005).

[3] T. H. Wang, F. C. Ni, K. T. Li, and Y. P. Chen, "Robust license plate recognition based on dynamic projection warping," *IEEE Int'l. Conf. on Networking, Sensing and Control* (IEEE, Piscataway, NJ, 2004), pp. 784–788.

[4] F. Wang, L. Man, B. Wang, Y. Xiao, W. Pan, and X. Lu, "Fuzzy-based algorithm for color recognition of license plates," Pattern Recognit. Lett. **29**, 1007–1020 (2008).

[5] D. S. Kim and S. I. Chien, "Automatic car license plate extraction using modified generalized symmetry transform and image warping," *2001 IEEE Int'l. Symposium on Industrial Electronics Proc.* (IEEE, Piscataway, NJ, 2001), pp. 2022–2027.

[6] C. N. E. Anagnostopoulos, I. E. Anagnostopoulos, V. Loumos, and E. Kayafas, "A license plate-recognition algorithm for intelligent transportation system applications," IEEE Trans. Intell. Transp. Syst. **7**, 377–392 (2006).

[7] H. Caner, H. S. Gecim, and A. Z. Alkar, "Efficient embedded neural network-based license plate recognition system," IEEE Trans. Veh. Technol. **57**, 2675–2683 (2008).

[8] F. Kahraman, B. Kurt, and M. Gökmen, "License plate character segmentation based on the Gabor transform and vector quantization," *Int'l. Symposium on Computer and Information Sciences* (Springer, Cham, 2003), pp. 381–388.

[9] H. Zhang, W. Jia, and X. He, "Learning-based license plate detection using global and local features," *18th Int'l. Conf. on Pattern Recognition* (IEEE, Piscataway, NJ, 2006).

[10] A. C. Roy, M. K. Hossen, and D. Nag, "License plate detection and character recognition system for commercial vehicles based on morphological approach and template matching," *2016 3rd Int'l. Conf. on Electrical Engineering and Information Communication Technology* (IEEE, Piscataway, NJ, 2016).

[11] Z. Xu, W. Yang, A. Meng, N. Lu, H. Huang, C. Ying, and L. Huang, "Towards end-to-end license plate detection and recognition: A large dataset and baseline," *15th European Conf., Munich, Germany* (Springer, Cham, 2018), pp. 261–277.

[12] J. Wang, H. Huang, X. Qian, J. Cao, and Y. Dai, "Sequence recognition of Chinese license plates," Neurocomputing **317**, 149–158 (2018).

[13] A. Bochkovskiy, C. Y. Wang, and H. Y. Mark Liao, "YOLOv4: Optimal speed and accuracy of object detection," arXiv:2004.10934 (2020).

[14] Z. Jie, Y. Wei, X. Jin, J. Feng, and W. Liu, "Deep self-taught learning for weakly supervised object localization," *2017 IEEE Conf. on Computer Vision and Pattern Recognition* (IEEE, Piscataway, NJ, 2017), pp. 1377–1385.

[15] Q. Xie, M. T. Luong, E. Hovy, and Q. V. Le, "Self-training with noisy student improves ImageNet classification," *2020 IEEE/CVF Conf. on Computer Vision and Pattern Recognition* (IEEE, Piscataway, NJ, 2020), pp. 10687–10698.

[16] D. Berthelot, N. Carlini, E. D. Cubuk, A. Kurakin, K. Sohn, H. Zhang, and C. Raffel, "ReMixMatch: Semi-supervised learning with distribution matching and augmentation anchoring," *2020 Int'l. Conf. on Learning Representations*, arXiv:1911.09785 [cs.LG] (2020).

[17] Q. Xie, Z. Dai, E. Hovy, M. T. Luong, and Q. V. Le, "Unsupervised data augmentation for consistency training," arXiv:1904.12848 (2019).

[18] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. V. Le, "Autoaugment: Learning augmentation strategies from data," *2019 IEEE/CVF Conf. on Computer Vision and Pattern Recognition* (IEEE, Piscataway, NJ, 2019).

[19] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," J. Big Data **6** (2019).

[20] C. H. Hsia and C. F. Lai, "Embedded vein recognition system with wavelet domain," Sens. Mater. **32**, 3221–3234 (2020).

[21] TesseractUserManual.

[22] C. Y. Wang, H. Y. Mark Liao, I. H. Yeh, Y. H. Wu, P. Y. Chen, and J. W. Hsieh, "CSPNet: A new backbone that can enhance learning capability of CNN," arXiv:1911.11929 (2019).

[23] J. Lyu, Y. Hu, S. Ren, Y. Yao, D. Ding, Q. Guan, and L. Tao, "Extracting the tailings ponds from high spatial resolution remote sensing images by integrating a deep learning-based model," Remote Sens. **13**, 743 (2021).