2018 24th International Conference on Pattern Recognition (ICPR)
Beijing, China, August 20-24, 2018

A robust and efficient method for license plate recognition

Ajin Meng[1], Wei Yang[1,*], Zhenbo Xu[1], Huan Huang[1,2], Liusheng Huang[1], Changchun Ying[2]
[1]University of Science and Technology of China, Hefei, China
[2]Hefei Xingtai Financial Holdings (Group) Co., Ltd.
[*]Corresponding Author. E-mail: qubit@ustc.edu.cn

***Abstract*—License plate recognition is an essential step in automatic license plate recognition since it is a key technology to recognize detected license plates. Though there are extensive researches on license plate recognition, it is still challenging to recognize license plates under conditions like great tilt angles, uneven illuminations, and distortions. Based on the observation that an accurate shape correction can significantly improve the recognition accuracy on these images, this paper proposes a robust methodology named LCR for license plate recognition free of conventional image analysis operations. This approach is based on three neural networks for three different purposes: (i) predicting the locations of four vertices; (ii) predicting cutting locations; (iii) character classification. To the best of our knowledge, LCR is the first to address shape correction by designing neural networks to accurately predict the coordinates of license plates vertices. Experiments on over 250,000 unique images show that LCR significantly outperforms several state-of-the-art license plate recognition approaches. Moreover, in evaluations, the application of shape correction significantly improve the recognition accuracy.**

I. Introduction

Automatic license plate recognition (ALPR) plays an important role in Intelligent Transportation System and has numerous real-life applications, such as traffic law enforcement, vehicle tracking [1], highway toll station, car park entrance and exit management. In recent years a large number of approaches for ALPR have been proposed (e.g. [2] [3] [4] [5] [6] [7] [8] [9]). However, it is still challenging to deal with license plates (LPs) under various real-life conditions.

Generally, ALPR algorithms can be divided into two stages: (i) LP detection. (ii) license plate recognition (LPR). On the one hand, with the development of deep convolutional neural network [10] and some novel structures based on it like Faster-RCNN [11], the accuracy of object detection easily surpasses humans. On the other hand, extensive researches (e.g. [12] [13] [14] [15]) have been done to make LPR methods faster or more accurate. However, most current researches achieve high LPR accuracy only under some controlled conditions like small tilt angles (about 20° downwards), high-resolution, even illumination, and etc. In other words, most LPR algorithms might not perform well in some real-life scenarios, as shown in Fig. 1, where LPs exhibit strong variations on tilt angles, illuminations, distortions and etc. These algorithms (e.g. [12] [13] [15]) only consider LPs with a regular shape and evaluate their algorithms on LPs with little variations.

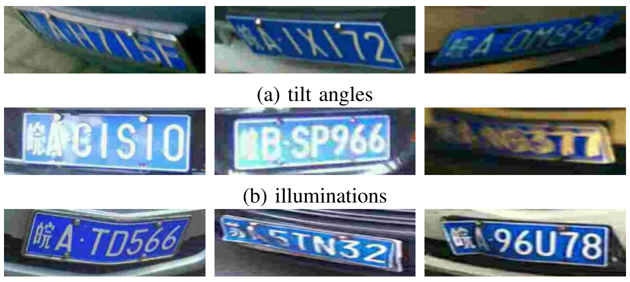


Fig. 1. Challenging images for license plate recognition

However, conditions like great tilt angles, uneven illuminations, distortions are common in daily life. Under these conditions, shape correction [6] [14] is essential because there are countless variations such as the possible shapes of LPs. Such a great number of variations make directly training robust neural networks for LPR in a natural environment very difficult and resource-consuming. On the contrary, if all LPs can be corrected into a regular size, LPs become much easier to be recognized. The importance of shape correction is also demonstrated by evaluations in Table II.

In this paper, we propose a robust and efficient LPR method named LCR (Locate, Cut and Recognize). To the best of our knowledge, LCR is the first method that designs neural networks to accurately predict the vertices coordinates of LPs for shape correction and the cutting locations for precise segmentation. For each image containing a complete LP, LCR first locates four vertices coordinates by a novel convolution neural network named LocateNet. Then, a regular LP is obtained by applying an affine transformation on the original image. After that, the obtained LP is reduced to 328 dimensions and a four-layer neural network named CutNet is exploited for predicting cutting locations. Finally, segmented LP character images are fed into a well-trained convolution neural network for LP character classification. Through extensive evaluations we demonstrate that LCR is robust under various conditions and LCR outperforms state-of-the-art LPR approaches on different datasets.

To summarize, this paper makes the following contributions:

- LocateNet is the first neural network designed to locate LP vertices, which contributes a lot to LCR in recognizing LP images under various conditions with high accuracy and efficiency.

- LCR does not rely on specific LP layouts. With enough training data, LCR is applicable to different styles of LPs.
- Extensive experiments on different datasets demonstrate that shape correction is essential for LPR tasks and LCR outperforms other state-of-the-art LPR approaches.

The rest of paper is organized as follows. Related work is discussed in Section 2. The workflow of our method is presented in Section 3. Training details and evaluations are summarized in Section 4. Finally, conclusions are drawn in Section 5.

II. Related Work

State-of-the-art LPR approaches can be divided into two main classes: (i) segmentation-based method. (ii) segmentation-free method.

A. Segmentation-based method

Most previous works (e.g. [16] [17] [18]) utilized image processing operations to segment characters in the LP, and then recognized each segmented character. In [16] [17], the binary LP image was projected horizontally to determine the top and bottom boundaries of LPs, and then vertically to find the gap between characters for further segmentation. [12] [18] exploited image processing operations and connected component analysis to extract LP character images and fed them into a well-trained CNN for classification. [13] utilized an optical character recognition (OCR) system to decompose each character into features like lines, closed loops, line direction, and line intersections and compare image features with these extracted features to recognize characters.

Current only a few papers (e.g. [6] [14]) applied shape correction to LPs before segmenting LP characters because, in most papers, the LPs to be recognized are basically rectangular shapes. Moreover, these shape correction related papers [19] usually assumed a good binary LP image can be obtained and utilized LP layout info and image analysis methods like hough transform to correct the shape of LPs. However, conventional image analysis methods are not robust enough to acquire suitable binary images for LPs in any possible environments. This work is the first to design neural networks to predict the locations of LP vertices for shape correction.

B. Segmentation-free method

Segmentation-free LPR methods can be classified into two categories: (i) utilize features around LP characters to extract LP characters directly so as to avoid segmentation. (ii) deliver the whole LP image to a convolutional Neural Network to perform the recognition task. [20] exploited scale invariant feature transform algorithms to extract and recognize LP characters directly. Li *et al.* [21] proposed a framework free of segmentation by utilizing deep convolutional neural networks and bi-directional recurrent neural network with long short-term memory, where a sliding window strategy is applied to extract feature sequence. Spanhel [15] presented a methodology based on a Convolutional Neural Network named Holistic-CNN which exploited the feature maps output by the last CNN layer and fed these feature maps into different branches of the net with fully connected layers to recognize LPs in a holistic way.

These segmentation-free methods avoid the challenging task of shape correction can be more efficient, but might not achieve high recognition accuracy on challenging LPs.

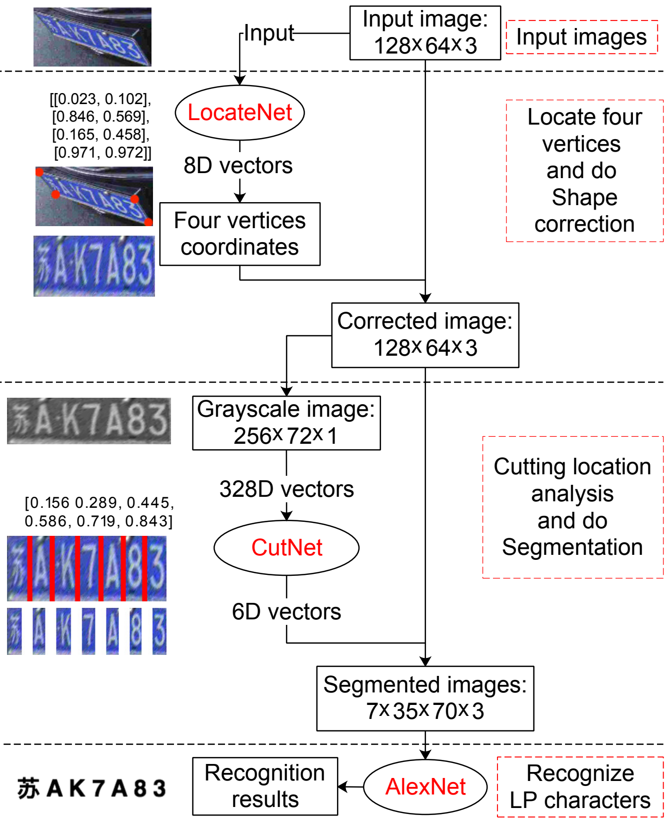


Fig. 2. The flowchart of LCR (Locate, Cut and Recognize)

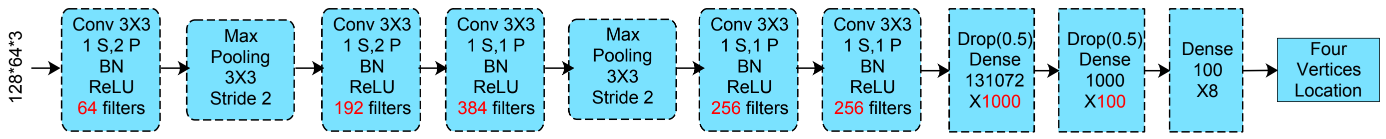
III. Proposed Method

In this section we introduce our proposed LCR method. The flowchart of LCR is presented in Fig. 2. The input of LCR is a LP image of any size captured by LP detection algorithms. LCR can be generally divided into three stages: (i) Locate four vertices and do shape correction. (ii) Cutting location analysis and do character segmentation. (iii) Character classification.

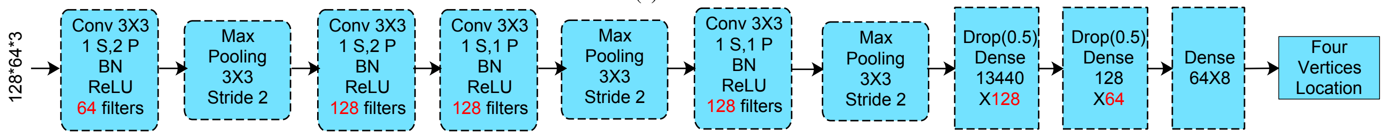
A. Locate

As shown in Fig. 2, we need to train a neural network which can analyze the input LP image and predict the four vertices coordinates. After obtaining the locations of four vertices, LCR can correct the shape of the LP.

We design a neural network model named LocateNet with 10 layers for predicting the four vertices coordinates, as shown in Fig. 3. The LocateNet model includes five convolutional layers (each convolutional layer with ReLU and Batch Normalization), two max pooling layers and three fully connected layers.


128*64*3
Conv 3X3 1 S,2 P BN ReLU 64 filters
Max Pooling 3X3 Stride 2
Conv 3X3 1 S,2 P BN ReLU 192 filters
Conv 3X3 1 S,1 P BN ReLU 384 filters
Max Pooling 3X3 Stride 2
Conv 3X3 1 S,1 P BN ReLU 256 filters
Conv 3X3 1 S,1 P BN ReLU 256 filters
Drop(0.5) Dense 131072 X1000
Drop(0.5) Dense 1000 X100
Dense 100 X8
Four Vertices Location

(a) LocateNet


128*64*3
Conv 3X3 1 S,2 P BN ReLU 64 filters
Max Pooling 3X3 Stride 2
Conv 3X3 1 S,2 P BN ReLU 128 filters
Conv 3X3 1 S,1 P BN ReLU 128 filters
Max Pooling 3X3 Stride 2
Conv 3X3 1 S,1 P BN ReLU 128 filters
Max Pooling 3X3 Stride 2
Drop(0.5) Dense 13440 X128
Drop(0.5) Dense 128 X64
Dense 64X8
Four Vertices Location

(b) Smaller LocateNet

Fig. 3. Schematic figure of CNN. The network contains five convolutional layers with ReLU and Batch Normalization, which extract a feature sequence from the input image. The features are then fed into three sequences of three fully connected layers predicting eight float numbers (x_0/W,y_0/H,...,x_3/W,y_3/H), that is, the coordinates of the four vertices of the LP.

All input images are resized to $128(Width)\times64(Height)\times3(Channels)$ for being fed into LocateNet. For each input image, 256 feature maps of size 32×16 are extracted after the fifth convolutional layer. The 256 feature maps are then concatenated together into one feature vector with length 131072, which are fed into the fully connected layers. Suppose W and H are the width and the height of the input image. $(x_i, y_i), (X_i, Y_i)$ $0 \leq i \leq 3$ denotes the predicted vertices coordinates and the ground truth vertices coordinates respectively. The final output is eight float numbers (x_0/W,y_0/H,...,x_3/W,y_3/H), which are the horizontal and vertical locations of four vertices.

During learning, we exploit back propagation and SGD with momentum to minimize the following loss:

$$L = \sum_0^3 (|X_i/W - x_i/W| + |Y_i/H - y_i/H|),$$

where X_i, Y_i are the ground truth vertices coordinates respectively.

We also tested a smaller network, named smaller LocateNet, which contains 4 convolutional layers (the fifth convolutional layer is replaced with a max pooling layer) with fewer filters (64/128/128/128 instead of 64/192/384/256/256) and fewer neurons in fully connected layers. The smaller LocateNet contains 21M parameters in contrast with 133M parameters in LocateNet. Evaluations demonstrate that, as shown in Fig. 6, the smaller LocateNet achieves slightly lower accuracy as LocateNet, but predicts faster and consumes fewer resources.

B. Cut

As shown in Fig. 2, after shape correction, we obtain an $128(Width)\times64(Height)\times3(Channels)$ image. Then, we resize the corrected image into $256(Width)\times72(Height)$ and convert it into grayscale. Each grayscale image is projected to the x and y plane to sum their grayscale value in each column and each row. Because the width and the height of the grayscale image are 256 pixels and 72 pixels respectively. After concatenation, we obtain a 328-dimension (328D) vectors. Next, we need to train a neural network which can analyze the 328D vectors, as shown in Fig. 4, extracted from the grayscale image and predict the cutting locations. After obtaining these cutting locations, we can segment the LP into several LP character images for character classification.

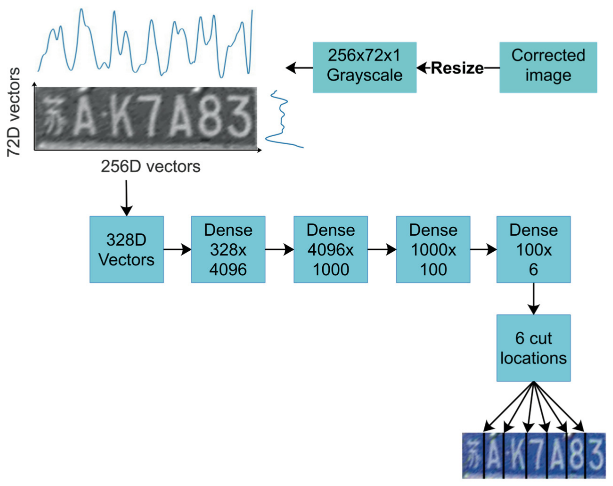


Fig. 4. The network architecture of CutNet. CutNet is consisted of four fully connected layers.

We design another neural network model named CutNet with four fully connected layers for cutting location prediction. CutNet does not exploit convolutional layers due to the following observation. As the brightness of LP characters is always higher than that of the inter-character area, the sum of the gray values of each row or each column varies between character area and inter-character area. The network architecture is shown in Fig. 4.

Suppose W is the width of the input image. z_i, Z_i $0 \leq i \leq 5$ denotes the predicted cutting locations and the ground truth cutting locations respetively. The final output is six float numbers(z_0/W,z_1/W,...,z_5/W) representing cutting locations.

During learning, we exploit back propagation and SGD with momentum to minimize the following loss:

$$L = \sum_0^5 (|Z_i/W - z_i/W|)$$

C. Recognize

After obtaining all LP character images, we exploit an AlexNet pre-trained on ImageNet [10] for character classification by changing its first convolutional layer and its dense layers and re-training only these layers. The details are omitted for simplicity.

IV. Data acquisition and accuracy metric

In this section, we first describe the way we collected data from several publicly available datasets [22] [23] [24] and augmented data for training and testing. Then, before evaluating LCR, we define accuracy metrics.

A. *Data acquisition and augmentation*

We collect over 250,000 unique LP images from CCPD [22], the largest LP dataset to date. CCPD contains tens of thousands of LPs under uncommon real-life conditions. For each image in CCPD, the ground truth coordinates of four vertices of each LP and the corresponding LP number are labeled. Recognition of images in CCPD is more challenging because these images exhibits strong variations in tilt angles, illuminations, and blurriness (see Fig. 5). For other two datasets Zemirs [23] and Medialab [24], as these images are not well annotated, we prepare training and test data in a semi-automatic way. We first use a small part of data to train a cascade classifier for extracting candidate for LP regions and manually select the correct LP regions. To determine the ground truth vertex coordinates of LPs, we develop a python program. This program shows one LP a time and can record vertex coordinates by manually clicking those vertices. By the hand of this annotation program, we label the LP number and the positions of four vertices.

As the data volume of Zemirs and Medialab is limited and images in these three datasets further comprise background regions around LPs, we augment each LP dataset by shifting bounding boxs several times in the following way: (i) determine the minimal bounding box of each LP. (ii) randomly shift four edges outward to obtain a wider bounding box and compute new vertex coordinates according to the corresponding shift. Moreover, the performance of LocateNet and CutNet is evaluated only on CCPD because the data volume in Zemirs and Medialab, though it reaches 5,000 images, is still not enough to accurately evaluate LocateNet and CutNet.

B. *Accuracy metric of LocateNet*

Suppose that A is the region (four vertices) given by the LocateNet and B is the ground truth region. $PA_i(0 \leq i \leq 3)$ denotes the four vertices coordinates (x, y) of A and $PB_i(0 \leq i \leq 3)$ denotes the four vertices coordinates (x, y) of B. The width of the LP is W and the height of the LP is H. A is an accurate prediction if and only if

$$(PA_i[x] - PB_i[x])/W < 0.05, \forall 0 \leq i \leq 3 \tag{1}$$

$$(PA_i[y] - PB_i[y])/W < 0.05, \forall 0 \leq i \leq 3 \tag{2}$$

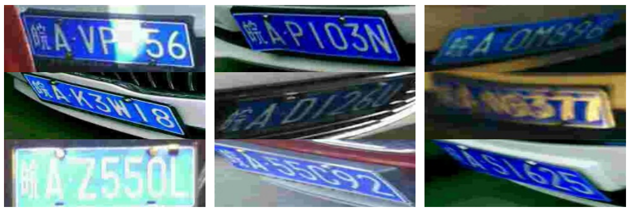


Fig. 5. License plates in CCPD

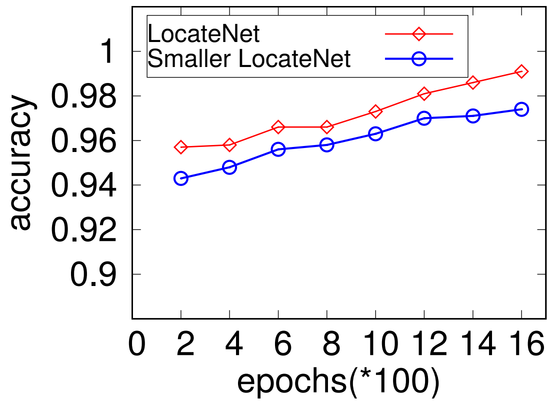


Fig. 6. The performance of LocateNet and smaller LocateNet

C. *Accuracy metric of CutNet*

Suppose that C is the cutting locations predicted by CutNet for segmenting character images. C is correct if and only if the recognition result of all character images, which are produced by segmenting the LP according to C, is the same with the ground truth LP number.

D. *Accuracy metric of LPR*

Suppose that R is the LP number predicted by LCR. R is correct if and only if R is the same with the ground truth LP number.

V. Evaluations

In this section, we first examine the performance of our designed LocateNet and CutNet on the largest dataset CCPD. Then, we coarsely introduce the way we implemented several state-of-the-art approaches for evaluation. Finally, these approaches and LCR are evaluated on three different datasets. All our training and test tasks are finished on desktop PCs with eight 3.40 GHz Intel Core i7-6700 CPUs, 24GB RAM and one Quadro P4000 GPU.

A. *The performance of LocateNet*

Following the accuracy metric of LocateNet, we evaluate LocateNet and the smaller version. For each dataset, training and testing tasks were conducted with 75% and 25% of the total data respectively. We train LocateNet using stochastic gradient descent (SGD) with momentum. The momentum factor value is 0.9 and the learning rate is 0.001. The accuracy on the test set is shown in Fig. 6. With the number of training epochs increases, LocateNet and the smaller version

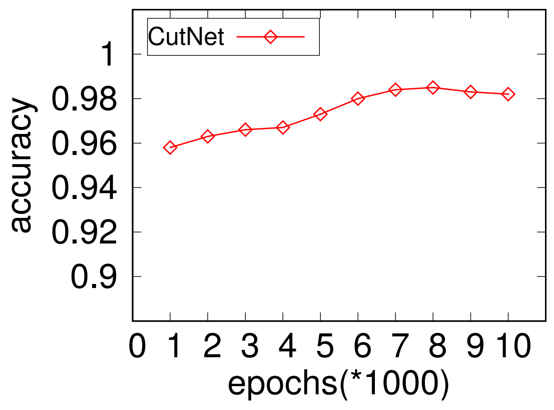


Fig. 7. The performance of CutNet

both achieves high locate accuracy on all these three datasets. Though the accuracy of LocateNet is slightly higher than small LocateNet, in experiments, the average time smaller LocateNet consumes to handle one image is a quarter of LocateNet.

B. The performance of CutNet

Before we present the accuracy metric for CutNet, as mentioned before, we train an AlexNet-like deep neural network to classify LP characters by transfer learning. The training data for character classification is provided in CCPD. For Zemirs and Medialab, as LPs includes only common letters and numbers, we exploit a publicly available LP character dataset [25] for training. The process of finetuning is omitted here for simplicity.

To prepare data for training and testing CutNet from CCPD, we first apply shape correction to all images and store these corrected LP images. Then we try to segment each LP image using image processing operations according to prior knowledge like general LP characters distribution. If the LP number recognized by the well-trained AlexNet is the same with the ground truth LP number, we believe these cutting locations are correct and store them as the correct cutting locations.

Following the accuracy metric of CutNet, we evaluate CutNet on these three datasets. For each dataset, training and testing were conducted with 75% and 25% of the total data respectively. We train LocateNet using stochastic gradient descent (SGD) with momentum. The momentum factor value is 0.9 and the learning rate is 0.0001. As it can be seen from Fig. 7, before the number of epochs reaches 8000, the accuracy is proportional to the number of epochs. After that, the accuracy starts to decrease. The underlining reason might be that the training data is not accurate as an accurate recognition can also be achieved by a coarse segmentation.

C. Implementation of state-of-the-art methods

After exact 2000 epochs training on each dataset, we combine the trained LocateNet and CutNet as our proposed LCR method for evaluation. Moreover, we implement and evaluate other three state-of-the-art methods on these datasets for comparisons.

Accuracy(%)	CCPD [22]	Zemirs [23]	Medialab [24]
LCR	95.9	97.3	97.2
OCR	45.3	93.2	91.8
IP+CNN	58.1	94.2	92.1
Holistic-CNN	83.9	97.1	96.5

TABLE I
ACCURACY OF LCR AND OTHER STATE-OF-THE-ART ALGORITHMS ON THREE PUBLICLY AVAILABLE DATASETS

Accuracy(%)	CCPD [22]	Zemirs [23]	Medialab [24]
LCR	96.2	97.3	98.3
OCR	69.6	95.1	93.3
IP+CNN	71.9	94.7	94.5
Holistic-CNN	96.4	97.7	96.9

TABLE II
ACCURACY OF LCR AND OTHER STATE-OF-THE-ART ALGORITHMS ON THREE PUBLICLY AVAILABLE DATASETS AFTER SHAPE CORRECTION

Speed(ms)	CCPD [22]	Zemirs [23]	Medialab [24]
LCR	6.9	6.3	7.2
OCR	19.0	16.7	17.2
IP+CNN	114.6	94.5	92.7
Holistic-CNN	1.1	0.91	0.88

TABLE III
SPEED OF LCR AND OTHER STATE-OF-THE-ART ALGORITHMS ON THREE PUBLICLY AVAILABLE DATASETS

1) OCR [13]: OCR algorithms decompose each character into features like lines, closed loops, line direction, and line intersections and compare image features with these extracted features to recognize characters. We evaluate LP character recognition on a famous online OCR service [26].

2) image processing (IP) [12] and CNN: Image processing related methods often exploit image processing operations to find regions with LP characters. After connected component analysis, LP characters are extracted and are fed into a well-trained CNN for classification. For each dataset, we fine-tune image processing parameters like the color range of LPs, the minimum spacing between characters, etc. to achieve the highest accuracy. Moreover, we exploit the same AlexNet in LCR for LP character classification.

3) Holistic CNN [15]: Follow the idea in [15], we implement and train the Holistic-CNN which holistically processes the whole image without the segmentation. Holistic-CNN exploits the feature maps output by the last CNN layer and feed these feature maps into several branches of the net with fully connected layers to predict characters on respective positions.

D. LPR Accuracy

As shown in TABLE I, LCR outperforms other methods on CCPD and achieves high accuracy on Zemirs and Medialab the same as Holistic-CNN. It's worth noting that, in contrast with Zemirs and Medialab, CCPD provides large quantities of images and also these images exhibit strong variations. Thus the recognition accuracy on CCPD is more convincing. From analyzing those images Holistic-CNN fails to recognize, we found Holistic-CNN can not handle images with great tilt angles well. Therefore, to further highlight the importance of shape correction, we produce a dataset with regular LPs by correcting the shape of all LPs and re-evaluate all methods. The results are shown in TABLE II. These three methods achieve significant higher accuracy. This increment of accuracy demonstrates the importance of shape correction.

E. LPR Speed

As shown in TABLE III, LCR is very efficient in comparison with IP methods and OCR. Though, the results show Holistic-CNN is faster than LCR because the shape correction in LCR is computed by CPU and thus consumes more time. But it should be noted that avoiding the shape correction, Holistic-CNN fails to achieve the same high accuracy as our proposed LCR. As the shape correction is essential for a robust LPR approach, it is worthy to sacrifice some recognition time.

VI. CONCLUSION AND FUTURE WORK

In this work, we present LCR, a robust and efficient method for LPR. LCR is the first to design neural networks to learn and predict the LP vertices coordinates. LCR outperforms other LPR algorithms in three different aspects: (i) LCR can accurately and quickly recognize LP images under various conditions. (ii) LCR does not rely on specific LP layouts and thus can be easily extended to different styles of LPs as long as enough training data is available. (iii) Evaluated on Large-scale different datasets, LCR outperforms state-of-the-art LPR approaches in both accuracy and speed. Moreover, through comparative experiments, we highlight the importance of shape correction in LPR.

ACKNOWLEDGMENTS

This work was supported by the National Natural Science Foundation of China (Nos. 61572456, 61672487), the Anhui Province Guidance Funds for Quantum Communication and Quantum Computers and the Natural Science Foundation of Jiangsu Province of China (No. BK20151241).

REFERENCES

[1] M. Donoser and H. Bischof, "Efficient maximally stable extremal region (mser) tracking," in *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, vol. 1. IEEE, 2006, pp. 553–560.

[2] S. Rasheed, A. Naeem, and O. Ishaq, "Automated number plate recognition using hough lines and template matching," in *Proceedings of the World Congress on Engineering and Computer Science*, vol. 1, 2012, pp. 24–26.

[3] I. Giannoukos, C.-N. Anagnostopoulos, V. Loumos, and E. Kayafas, "Operator context scanning to support high segmentation rates for real time license plate recognition," *Pattern Recognition*, vol. 43, no. 11, pp. 3866–3878, 2010.

[4] L. Zheng, X. He, B. Samali, and L. T. Yang, "Accuracy enhancement for license plate recognition," in *Computer and Information Technology (CIT), 2010 IEEE 10th International Conference on*. IEEE, 2010, pp. 511–516.

[5] T. K. Cheang, Y. S. Chong, and Y. H. Tay, "Segmentation-free vehicle license plate recognition using convnet-rnn," *arXiv preprint arXiv:1701.06439*, 2017.

[6] Y. Wen, Y. Lu, J. Yan, Z. Zhou, K. M. von Deneen, and P. Shi, "An algorithm for license plate recognition applied to intelligent transportation system," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 3, pp. 830–845, 2011.

[7] Y. Jing-hua, "License plate recognition technology research," *Netinfo Security*, no. 4, pp. 0–0, 2011.

[8] V. Jain, Z. Sasindran, A. Rajagopal, S. Biswas, H. S. Bharadwaj, and K. Ramakrishnan, "Deep automatic license plate recognition system," in *Proceedings of the Tenth Indian Conference on Computer Vision, Graphics and Image Processing*. ACM, 2016, p. 6.

[9] H.-H. Kim, J.-K. Park, J.-H. Oh, and D.-J. Kang, "Multi-task convolutional neural network system for license plate recognition," *International Journal of Control, Automation and Systems*, vol. 15, no. 6, pp. 2942–2949, 2017.

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[11] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91–99.

[12] G. A. Smara and F. Khalefah, "Localization of license plate number using dynamic image processing techniques and genetic algorithms," *IEEE transactions on evolutionary computation*, vol. 18, no. 2, pp. 244–257, 2014.

[13] K. M. A. Yousef, M. Al-Tabanjah, E. Hudaib, and M. Ikrai, "Sift based automatic number plate recognition," in *Information and Communication Systems (ICICS), 2015 6th International Conference on*. IEEE, 2015, pp. 124–129.

[14] S. Azam and M. M. Islam, "Automatic license plate detection in hazardous condition," *Journal of Visual Communication and Image Representation*, vol. 36, pp. 172–186, 2016.

[15] J. Špaňhel, J. Sochor, R. Juránek, A. Herout, L. Maršík, and P. Zemčík, "Holistic recognition of low quality license plates by cnn using track annotated data," in *Advanced Video and Signal Based Surveillance (AVSS), 2017 14th IEEE International Conference on*. IEEE, 2017, pp. 1–6.

[16] J.-M. Guo and Y.-F. Liu, "License plate localization and character segmentation with feedback self-learning and hybrid binarization techniques," *IEEE Transactions on Vehicular Technology*, vol. 57, no. 3, pp. 1417–1424, 2008.

[17] S. Qiao, Y. Zhu, X. Li, T. Liu, and B. Zhang, "Research of improving the accuracy of license plate character segmentation," in *Frontier of Computer Science and Technology (FCST), 2010 Fifth International Conference on*. IEEE, 2010, pp. 489–493.

[18] S.-L. Chang, L.-S. Chen, Y.-C. Chung, and S.-W. Chen, "Automatic license plate recognition," *IEEE transactions on intelligent transportation systems*, vol. 5, no. 1, pp. 42–53, 2004.

[19] S. Saha, S. Basu, M. Nasipuri, and D. K. Basu, "A hough transform based technique for text segmentation," *arXiv preprint arXiv:1002.4048*, 2010.

[20] M. Zahedi and S. M. Salehi, "License plate recognition system based on sift features," *Procedia Computer Science*, vol. 3, pp. 998–1002, 2011.

[21] X. Shi, W. Zhao, and Y. Shen, "Automatic license plate recognition system based on color image processing," in *International Conference on Computational Science and Its Applications*. Springer, 2005, pp. 1159–1168.

[22] CCPD, "China City Parking Dataset," https://github.com/ccpd2018/ccpd.

[23] Zemris, "Zemris License Plate Dataset," http://www.zemris.fer.hr/projects/LicensePlates/hrvatski/rezultati.shtml.

[24] C. N. E. Anagnostopoulos, I. E. Anagnostopoulos, I. D. Psoroulas, V. Loumos, and E. Kayafas, "License plate recognition from still images and video sequences: A survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 9, no. 3, pp. 377–391, Sept 2008.

[25] R. Rawat, M. Manry, and F. Martinez, "Character classification data for license plates," 3 2016. [Online]. Available: https://figshare.com/articles/Character_classification_data_for_license_plates/3113449

[26] ABBYY, "ABBYY Online OCR service," http://www.abbyy.cn/.