A Scalable and Efficient Multi-label CNN-based License Plate Recognition On Spark

Weishan Zhang
Department of Software Engineering
China University of Petroleum
Qingdao, China
zhangws@upc.edu.cn

Bing Xue
Department of Software Engineering
China University of Petroleum
Qingdao, China
b199402030210@qq.com

Jiehan Zhou
Oulu University
Finland
jiehan.zhou@oulu.fi

Xin Liu
School Computer and Communication Engineering
China University of Petroleum
Qingdao, China
lx@upc.edu.cn

Hao Lv
College Computer and Communication Engineering
China University of Petroleum
Qingdao, China
lvhao.upc@gmail.com

***Abstract*—Surveillance cameras are being rapidly deployed for facilitating smart transportation. Recognizing the vehicle license plate from massive videos becomes a challenge in context of system scalability and efficiency. This paper proposes a novel algorithm for scalable and efficient license plate recognition (SELPR). The SELPR algorithm first locates the license plate using a YOLO (You Look Only Once) network and recognizes the license plate using multi-label convolutional neural network (Multi-label CNN). We deploy the SELPR algorithm to the Apache Spark framework to evaluate its scalability and efficiency in parallel processing. The results demonstrates that SELPR can achieve synthesized performance with 95% recognition accuracy, better processing efficiency and scalability on a Spark cluster.**

I. Introduction

More and more cameras are deployed for smart transportation, which contributes to the fact that video data becomes the majority of big data [1]. Efficiently mining of these data, e.g., recognizing license plate [2] from traffic video data is important for detecting the traffic status, tracking vehicles, and so on. License plate recognition (LPR) is one of the earliest application for smart transportation [3].

The main LPR steps consists of locating a license plate, segmenting the plate characters, and obtaining the complete plate information [3] [4]. There existed reports on LPR, for example, vertical edge detection [5], color feature extraction [6], and so on.

The deep learning based approaches promise better performance than traditional plate feature extraction in a complex background such as different angles, brightness, and weather, [7] proposed an approach based on a visual attention model and deep learning. He used a modified visual attention model to detect license plate. But this approach is not accurate for plate recognition in the above mentioned complex situations. Masood [8] proposed an end-to-end license plate recognition pipeline with a sequence of deep CNNs. The pipeline consists of three steps such as plate location, plate character segmentation and character recognition.Cheang [9] proposed a unified CNN-RNN model, which uses CNN to extract features and uses RNN for modeling the feature and label sequence to recognize plate. The idea is interesting to integrate plate segmentation and recognition. However, his work ignores the location of a plate. And we consider there are still rooms to improve CNN-RNN approach in the context of performance.

YOLO can be a fast and lightweight method suitable for locating a license plate [10]. YOLO directly takes the entire image as input in training and running. Therefore YOLO is able to encode the contextual information on objects. So YOLO has less background errors than other approaches. Multi-label CNN is suitable for plate recognition. A Multi-label CNN [11] can be trained by inputting multiple labels, where all different classes can share the convolution layer and only need one classification model, so that the classification of a plate does not need to repeat the calculation of the classification network, which leads to an improved recognition efficiency. This method is efficient for plate recognition with a same size and same format, like Chinese vehicle plates.

The underlying platform running those algorithms are equally important to themselves for effective LPR. There are widely used big data processing framework providing good chances for efficiently run LPR algorithms, like Apache Hadoop[1] [12] [13]. In this paper, we deploy our SELPR algorithm on the Apache Spark[2], which is a memory-based large data parallel processing framework. Apache Spark can greatly improve the processing speed through saving the intermediate result in memory rather than on a local disk.

This paper aims to propose a scalable and efficient LPR approach (SELPR). SELPR uses YOLO network to detect the location of a license plate. Most existing LPRs often divide license plate characters and character recognition into two steps and our proposed SELPR combines both steps into one

[1] http://hadoop.apache.org/
[2] http://Spark.apache.org/

using multi-label CNN [11]. Therefore SELPR simplifies LPR pipeline and improves LPR performance. We improve SELPR performance by feeding video frames into RDD (Resilient Distributed Datasets) [14] and running it in parallel. The contributions of this paper are as the follows:

- We propose a SELPR algorithm, which combines multi-label convolutional neural network and YOLO in order to improve the plate recognition accuracy.
- We deploy the SELPR algorithm on Apache Spark framework to improve SELPR performance.
- We verify the SELPR algorithm in terms of recognition accuracy, algorithm efficiency and scalability.

The remainder of the paper is organized as the follows: Section 2 presents the SELPR architecture. Section 3 designs and implements the SELPR algorithm. Section 4 evaluates the SELPR algorithm. Section 5 reviews the related work. Section 6 presents future work and draws a conclusion.

II. SELPR Architecture

The SELPR architecture is illustrated as Figure. 1. It is based on our previous work on intelligent video processing [12] and combines Hadoop based batch processing with Spark based in-memory processing.

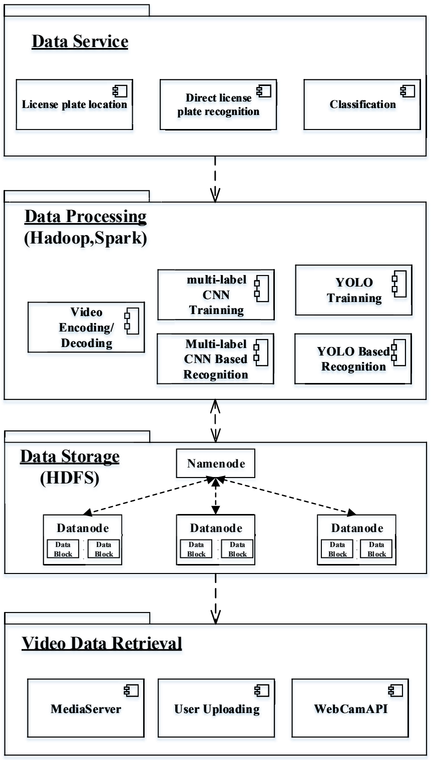

Figure 1: SELPR architecture

SELPR architecture consists of four layers. The bottom layer is responsible for collecting and generating traffic video data with the use of a video streaming server and a WebCam module.

The second layer is the data storage layer built on HDFS (Hadoop Distributed File System). HDFS as a distributed file system which is fault-tolerant. HDFS provides high throughput access to application data and is suitable for applications that have large data sets.

The third layer is the data processing layer which is built on Hadoop and Spark. In this layer, we also deploy Caffe[3] framework for data training and plate recognition. We use Hadoop MapReduce [15] to decode videos and train neural networks. We deploy the run-time plate recognition and memory computation tasks on Spark clusters.

The top layer is the data service layer, which visualizes results and provides the user with an interface for plate location, recognition and classification.

III. SELPR Algorithm and Implementation

A LPR algorithm usually consists of three steps: plate location, plate character segmentation, plate character recognition. Plate location determines the location of a license plate from a surveillance video frame. Plate character segmentation divides each character on the plate. Character recognition obtains the recognition results. Our SELPR is different from other LPR approaches. SELPR algorithm only consists of two steps: plate location and plate recognition.

A. *SELPR plate location*

SELPR uses a YOLO network for plate location. SELPR uses traffic video frames artificially marked with plate location as the YOLO training data set. The final output target class for YOLO is changed from twenty classes to one because our objection is only one license plate. We adjust YOLO network parameters in order to achieve better recognition results. Although YOLO can achieve a good recognition accuracy, there may still have multiple boxes for a same object. For reducing redundant boxes, we use the non-maxima suppression to select the best one. Figure. 2 illustrates SELPR plate location.

B. *SELPR plate recognition*

SELPR doesn't separate plate character segmentation and character recognition. It uses multi-label CNN model to directly recognize the plate after locating a plate. The SELPR multi-label CNN model is illustrated in Figure. 3. SELPR simplifies LPR pipeline, which consists of the following tasks.

- Firstly, SELPR collects plate images as an original dataset. Then it makes a multi-label dataset, where each plate image has seven labels corresponding to seven characters. Finally, SELPR needs to modify Caffe to adapt to multi-label CNN.
- Different from single-label classification which only needs one loss function, multi-label CNN classification needs to set multiple loss function layers. Each loss function layer corresponds to a type of label.

[3] http://caffe.berkeleyvision.org/

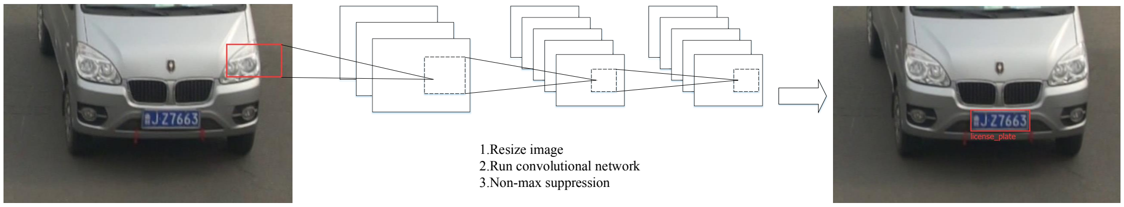

Figure 2: SELPR plate locating with YOLO

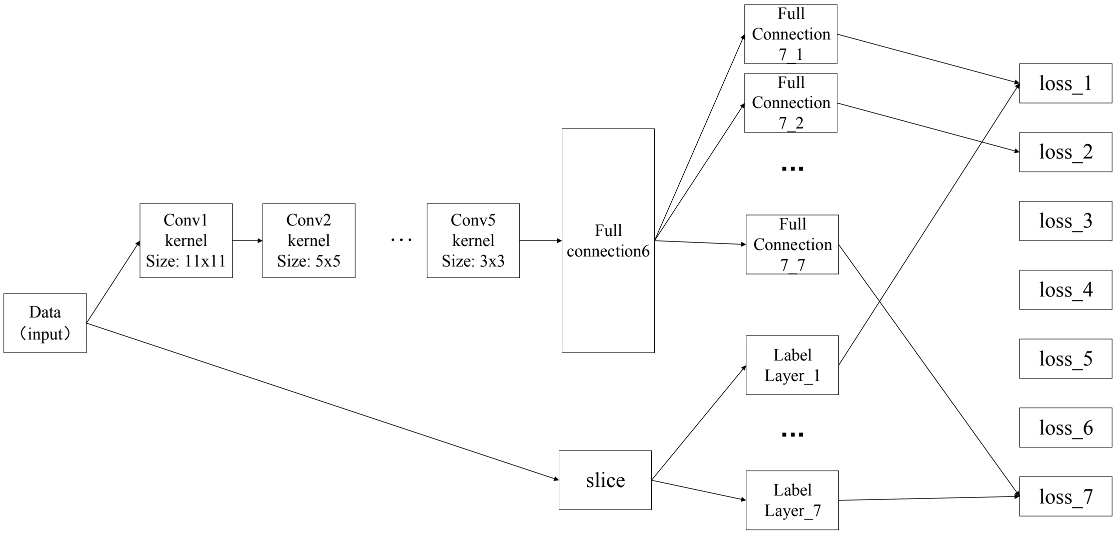

Figure 3: multi-label CNN model

- To calculate loss functions, SELPR needs to connect the full connection layer, and also to the corresponding label layer which is cut by the slice layer.
- SELPR set weight to 0.143 because a Chinese vehicle plate has seven characters with a same size.

C. SELPR

Figure. 4 illustrates the pipeline of SELPR plate recognition. As mentioned in Figure. 1, the video collection layer temporarily stores data collected by different methods in HDFS. In order to facilitate the sub-processing, SELPR decodes data into frames and converts frames to SequenceFile, which is in the form of a pair (key, value) stored in HDFS. SequenceFile is a small file solution from Hadoop to improve cluster utilization. SELPR stores data on images such as time stamp, location in the ’key’ and a binary numbers in the ’value’.

The core of SELPR algorithm is to parallelize SELPR plate location and plate recognition. Here SELPR uses Map transformation operation in RDD, that executes a function that can be programmed by developer on all elements in the RDD to produce a new RDD. The SELPR plate recognition on Spark is given in Algorithm. 1.

The SELPR plate location and recognition are realized in two Map operations (Map1 and Map2 respectively). Map1 is

Algorithm 1 License Plate Recognition

```
Input: Sequencefile of video data
Output: Text of recognition results
  RDD1 $\langle K_1, V_1 \rangle \leftarrow$ Sequencefile.AddRDD
  RDD2 $\langle K_2, V_2 \rangle$= RDD1.map1 (license plate locating)
    imageinfo $\leftarrow K_1$
    imagevalue $\leftarrow V_1$
    image $\leftarrow$ imagevalue.convert
    location $\leftarrow$ image.locate(YOLO model)
    $K_2 \leftarrow$ imageinfo+location
    $V_2 \leftarrow V_1$
  RDD3$\langle Text \rangle$ = RDD2.map2 (direct LPR)
    locationAndImageinfo $\leftarrow K_2$.getkey
    location $\leftarrow$ locationAndImageinfo.analyze
    imagevalue $\leftarrow$ V.getvalue
    plateimage $\leftarrow$ imagevalue.cut(location)
    characters $\leftarrow$ plateimage.recognize(multi-label CNN model)
    results $\leftarrow$ imageinfo+charactersOfPlate
  Text = RDD3.saveAsText
  return Text
```

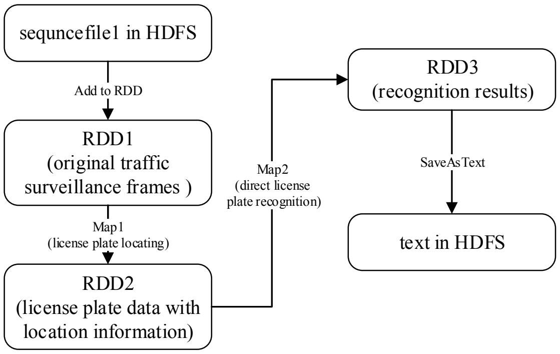

Figure 4: SELPR pipeline

for plate location, where the input data is the RDD which loads data from Sequencefile and processes this RDD.

Then SELPR converts binary data to Mat, and uses the packaged YOLO network model (after training) to extract the information on license plate in the underlying frame.

Finally, it returns the image information along with the license plate location. The Map operation (Map2) of SELPR plate recognition is similar to the license plate locating. Here the Mat data of a single license plate is handled by a multi-label CNN model and we get every plate character. At last it returns the image information along with every plate character. In the end of this algorithm, RDD is converted to a text file saved in HDFS.

IV. Evaluation

To verify our proposed SELPR approach, we collected real-time traffic surveillance videos and open source data set of Chinese vehicle plates. We evaluate SELPR in the terms of recognition accuracy, performance and scalability.

A. *Experiment environment*

Table. I presents the test bed configuration. The software packages include jdk1.8, OpenCV-2.4.9, Ant-1.7, Hadoop-2.6.4, Spark-1.6.3, Cuda-8.0 and Caffe.

Table I: Experimental hardware

Node No.	**Hardware**	**number**
1	i7 6800K, 16G, NVIDIA 1070	1
2, 3, 4, 5, 6, 7	i7 4790, 16G, NVIDIA 745	6

The Hadoop cluster mainly comprised of two parts. Part 1 is HDFS (Hadoop Distribute File System) consisting of a NameNode and six DataNodes. Part 2 is MapReduce consisting of a NodeManager and six ResourceManagers in this Hadoop cluster. Here a Hadoop cluster is mainly responsible for data storage and video encoding/decoding. The Spark cluster comprises of a master node and six slave nodes. The master node manages tasks, and six slave nodes process video, which installed SELPR algorithm. Table. II presents detailed allocations for each roles.

Table II: Experimental configurations

Node No.	**Role**
1	NameNode, ResourceManager, master
2, 3, 4, 5, 6, 7	DataNode, NodeManager, slave

B. *SELPR recognition accuracy*

We first train YOLO net work for plate location. We used 7587 traffic surveillance images with license plates as training sets and 734 images as test sets. After the training, the plate location accuracy reaches 99%.

Next, we select 6941 separate plate images as the training sets for SELPR plate recognition, 592 images as testing sets. With SELPR, plate character recognition accuracy reaches 96%.

Figure. 5 shows overall SELPR process in this experiment with 1000 images as test data. First SELPR detects plate location through YOLO network. Then it runs multi-label CNN network to extract plate characters. The overall SELPR accuracy reaches 95% by processing a frame. We also conduct a continuous plate recognition through 5 frames, and the overall SELPR accuracy reaches close to 100%.

C. *Comparison*

We compare SELPR algorithm with open OpenALPR[4], with the use of the same test dataset for accuracy experiment.

As shown in Figure. 6, YOLO based approach is much better than OpenALPR on Chinese vehicle plate location.

OpenALPR doesn't support Chinese characters that represent the abbreviation of Chinese provinces. We have to program OpenALPR for Chinese character recognition. Chinese vehicle plate usually contains seven characters. Its first character is the abbreviation of a Chinese provinces. Its second one is an English letter that represents a city in the province. Its remaining five characters are random combinations of numbers and English letters. We use this plate structure to train identifiers for plate character recognition. Figure. 6 presents the comparison of SELPR with traditional LPRs and OpenALPR. The result shows that Multi-label CNN-based SELPR outperforms others.

D. *Performance*

For performance evaluation, we choose different sizes of traffic videos (the resolution is 1280*720) as test data (Table. III), each of them is tested eight times (see Table. IV, time as second). The results are given in Figure. 7, X-axis represents input data size, and Y-axis represents processing time. We can see that processing time is increasing almost in linear with input data. We also test plate recognition time using GTX 1070. And the average time is 30ms per frame. That shows SELPR has an outstanding performance in plate recognition.

[4] https://github.com/openalpr/openalpr

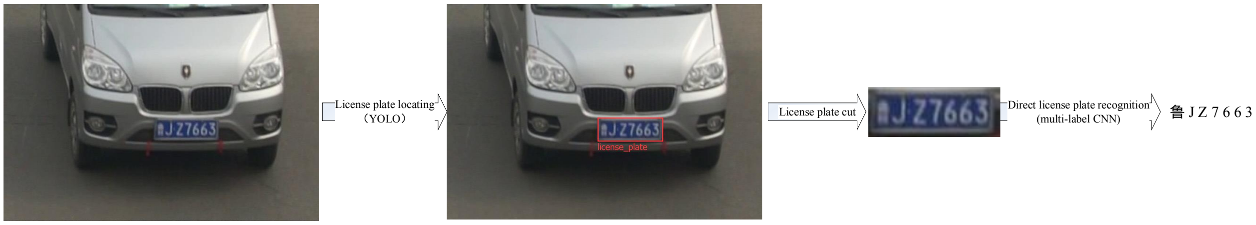

Figure 5: Process of LPR

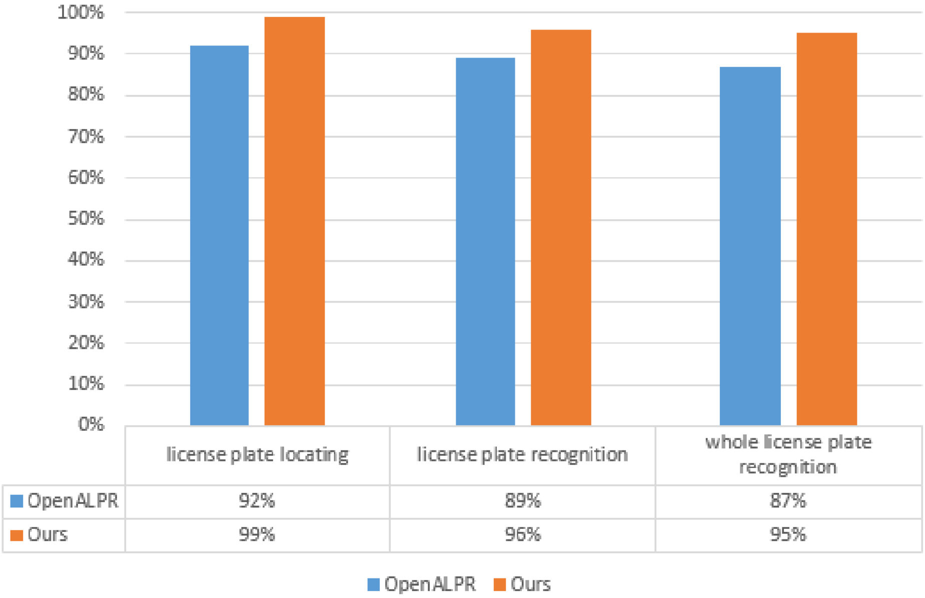

Figure 6: Comparison of SELPR with other method experiment

Table III: Input SequenceFile size

Experiment No.	A	B	C	D	E
Size(Gb)	3.2	6.1	8.6	11.3	14.5

Table IV: Performance experiment of SELPR based on Spark

Exp.No.	1	2	3	4	5	6	7	8	average
A	751	785	781	755	783	764	752	771	767
B	1464	1484	1483	1519	1456	1443	1523	1535	1488
C	2073	2084	2072	2082	2054	2061	2060	2081	2071
D	2750	2805	2806	2785	2808	2779	2802	2777	2789
E	3537	3521	3497	3492	3533	3491	3494	3469	3467

E. Scalability

We test scalability for SELPR by changing Spark executor numbers. In the experiment environment configuration, each executor runs only one slave node. In order to ensure the experiment accuracy, we unified 3.2 Gb SequenceFile file as input data. Table. V and Figure. 8 present the scalability experiment.

Figure. 8 shows that with the growth of slave nodes, the proposed SELPR algorithm runs faster as expected. It also shows time decreasing gradually slows down. That means that the current computing resources are enough and this is consistent with the fact because we only deal with 3.2Gb data in this experiment.

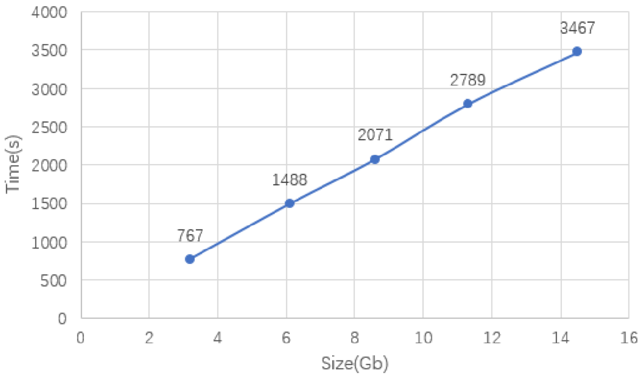

Figure 7: Performance experiment of SELPR based on Spark

Table V: Scalability experiment of SELPR based on Spark

Exp.No.	1	2	3	4	5	6	7	8	average
1	4074	4017	4039	4064	4045	4024	4070	4117	4056
2	2341	2334	2318	2367	2344	2340	2321	2347	2339
3	2341	2334	2318	2367	2344	2340	2329	2339	2339
4	1150	1177	1185	1171	1197	1150	1195	1200	1178
5	943	888	956	933	924	934	908	917	925
6	751	785	781	755	783	764	752	771	767

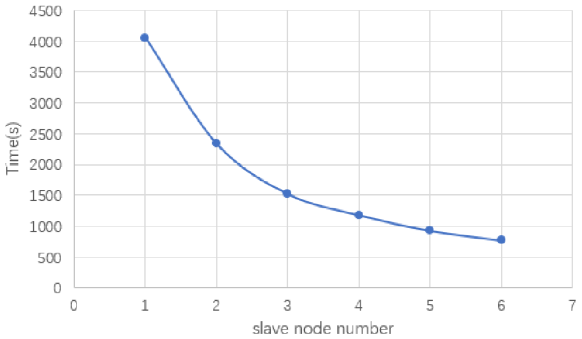

Figure 8: Scalability experiment of SELPR based on Spark

V. Related Work

Many researches have been done on LPR. Qadri [4] proposed an approach involving a series of steps, including searching for yellow pixels from plates with yellow background, filtering image, plate region extraction, line separation, characters extraction and optical character recognition. Those image processing techniques can be grouped into license plate location, character segmentation and character recognition. Chakraborty [16] proposed an improved template matching algorithm for LPR and achieved a good recognition accuracy. But his experiment only focused on character segmentation and recognizing or Indian vehicle license plates. Mai [17] proposed an approach for LPR by combining edge detection, image subtraction, mathematical morphology, radon transform, interpolation and specific characteristics for Vietnam vehicle license plates. They trained a MLP neural network and used BP algorithm for character recognition. The method achieved a high accuracy on Vietnam vehicle plate recognition, but its performance declines when plate edge is not clear due to view angle. In addition, other traditional LPR processing methods are such as vertical-edge-based detection [5] and pattern recognition [18].

With the development of deep learning, there grows in popularity on deep learning based LPR. Masood. [8] proposed an end-to-end license plate detection and recognition pipeline which is with a sequence of deep CNNs. In order to ensure system robustness, they used different conditions and types of license plates as training data sets. They used convolution neural network in plate location, character segmentation and character recognition. In contrast, we simplify their three steps and use multi-lable CNN, and target the improvement of recognition speed and accuracy for Chinese vehicle plate recognition.

Cheang [9] proposed a unified CNN-RNN model, which uses CNN to extract features and use RNN for sequencing, to recognize license plates. This method combines feature learning and joint image/label embedding of CNN and RNN to model the feature and label sequence for LPR. This idea is same as ours and integrates character segmentation and recognition. However, we used multi- label CNN, which is faster than CNN-RNN model. Also they did not consider plate location.

Zang. [7] proposed an approach based on a visual attention model and deep learning. They used a modified visual attention model to detect license plate. The attention model takes into account of plate color, brightness, orientation. They utilized a projection method for character segmentation and recognized plate characters through combining CNN and SVM. They used method based SSD in license plate location. SELPR also uses similar method to fit different angles of license plate to achieve high accuracy with fast locating speed.

VI. Conclusion and Future Work

In this paper, we propose a SELPR approach for recognizing vehicle plate using Apache Spark. A SELPR algorithm is designed based on deep learning for Chinese vehicle plate recognition. The SELPR algorithm is deployed on Spark framework to cope with massive traffic surveillance data in parallel. Extensive experiments verify that SELPR is effective. The evaluation shows that the SELPR solution is scalable and efficient.

In the future, we will work on enhancing the robustness of SELPR in extreme weather situations such as rainstorm, snow, haze and so on.

Acknowledgment

This work was supported in part by the Key Research Program of Shandong Province under Grant 2017GGX10140 and in part by the Fundamental Research Funds for the Central Universities(15CX08015A), National Natural Science Foundation of China (No. 61309024).

References

[1] T. Huang, “Surveillance video: The biggest big data,” *Computing Now*, vol. 7, no. 2, pp. 82–91, 2014.

[2] S. Du, M. Ibrahim, M. Shehata, and W. Badawy, “Automatic license plate recognition (alpr): A state-of-the-art review,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 23, no. 2, pp. 311–325, 2013.

[3] R. Baran, T. Rusc, and P. Fornalski, “A smart camera for the surveillance of vehicles in intelligent transportation systems,” *Multimedia Tools and Applications*, vol. 75, no. 17, pp. 10 471–10 493, 2016.

[4] M. T. Qadri and M. Asif, “Automatic number plate recognition system for vehicle identification using optical character recognition,” in *International Conference on Education Technology and Computer*, 2009, pp. 335–338.

[5] A. M. Al-Ghaili, S. Mashohor, A. R. Ramli, and A. Ismail, “Vertical-edge-based car-license-plate detection method,” *IEEE transactions on vehicular technology*, vol. 62, no. 1, pp. 26–38, 2013.

[6] A. H. Ashtari, M. J. Nordin, and M. Fathy, “An iranian license plate recognition system based on color features,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 4, pp. 1690–1705, 2014.

[7] D. Zang, Z. Chai, J. Zhang, D. Zhang, and J. Cheng, “Vehicle license plate recognition using visual attention model and deep learning,” *Journal of Electronic Imaging*, vol. 24, no. 3, pp. 033 001–033 001, 2015.

[8] S. Z. Masood, G. Shu, A. Dehghan, and E. G. Ortiz, “License plate detection and recognition using deeply learned convolutional neural networks,” *arXiv preprint arXiv:1703.07330*, 2017.

[9] 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.

[10] J. Redmon and A. Farhadi, “Yolo9000: Better, faster, stronger,” *arXiv preprint arXiv:1612.08242*, 2016.

[11] Y. Wei, W. Xia, J. Huang, B. Ni, J. Dong, Y. Zhao, and S. Yan, “Cnn: Single-label to multi-label,” *arXiv preprint arXiv:1406.5726*, 2014.

[12] W. Zhang, L. Xu, Z. Li, Q. Lu, and Y. Liu, “A deep-intelligence framework for online video processing,” *IEEE Software*, vol. 33, no. 2, pp. 44–51, 2016.

[13] C. Ryu, D. Lee, M. Jang, C. Kim, and E. Seo, “Extensible video processing framework in apache hadoop,” in *Cloud Computing Technology and Science (CloudCom), 2013 IEEE 5th International Conference on*, vol. 2. IEEE, 2013, pp. 305–310.

[14] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. Mccauley, M. J. Franklin, S. Shenker, and I. Stoica, “Resilient distributed datasets: a fault-tolerant abstraction for in-memory cluster computing,” in *Usenix Conference on Networked Systems Design and Implementation*, 2012, pp. 2–2.

[15] J. Dean and S. Ghemawat, “Mapreduce: simplified data processing on large clusters,” *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[16] S. Chakraborty and R. Parekh, “An improved template matching algorithm for car license plate recognition,” *International Journal of Computer Applications*, vol. 118, no. 25, 2015.

[17] V. Mai, D. Miao, and R. Wang, “Vietnam license plate recognition system based on edge detection and neural networks,” *Journal of Information and Computing Science*, vol. 8, no. 1, pp. 27–40, 2013.

[18] R. Azad, F. Davami, and B. Azad, “A novel and robust method for automatic license plate recognition system based on pattern recognition,” *Advances in Computer Science: an International Journal*, vol. 2, no. 3, pp. 64–70, 2013.