INDIAN LICENSE PLATE DETECTION AND RECOGNITION USING YOLOV8 AND LPRNET

Shashwat Bansal
Department of Computer Science and Engineering
Amity University
Noida, India
shashwat.bansal@s.amity.edu

Pulkit Rathi
Department of Computer Science and Engineering
Amity University
Noida, India
pulkit.rathi@s.amity.edu

Vibha Nehra
Department of Computer Science and Engineering
Amity University
Noida, India
vnehra@amity.edu

***Abstract*— Several new uses of ANPR technology have emerged in last few years. Because of this, the latest Automatic Number Plate Recognition (ANPR) systems face challenges in accurately identifying license plates due to variations in their orientation, size, and language. This research paper introduces YOLOv8 and LPRNet as methods for automatically detecting and recognizing Indian license plates. This Paper shows that by making some changes to the LPRNet an accuracy of 99% can be achieved in license plate recognition. This paper also shows that YOLOv8 achieved the highest mAP of 99% for license plate detection for Indian Number plates and also compares the other methods by running the models on different datasets which are Media Lab and AOLP. The experiments were conducted on NVIDIA GeForce RTX 3060, 12288MiB GPU, and a 12th Gen Intel(R) Core(™) i7-12700F 2.10GHz CPU. The achieved latency for detection and recognition averaged 18ms and 28ms, respectively.**

***Keywords*— Automatic License Plate Recognition (ANPR), Vehicle Detection, Computer Vision, Deep Learning, Artificial Intelligence**

I. INTRODUCTION

Controlling traffic, monitoring digital security, identifying vehicles, and managing parking in large cities makes automatic license plate recognition difficult thus, it becomes crucial work. Due to various issues, including fuzzy photos, bad lighting, variation in license plate numbers (including unique characters), physical impact (deformations), and weather conditions, this task is difficult. For an Automatic Number Plate Recognition (ANPR) system to be considered robust, it should be capable of operating effectively and maintaining high accuracy across diverse situations and environments. In other words, the system should demonstrate its ability to perform reliably under varying conditions.

A typical License Plate (LP) recognition system consists of four processing stages (fig. 1). The first stage involves capturing an image or video using a camera. In the second stage, the system detects the license plate within the captured image. The next stage involves extracting the individual characters from the license plate. Finally, in the last stage, the system identifies the extracted characters using multiple classifiers. These four stages are typically implemented by combining various deep-learning technologies or image-processing techniques to achieve accurate license plate recognition.

In this paper, algorithms for the Automatic Number Plate Recognition (ANPR) system are presented. The license plate detection task is addressed using the YOLOv8 model, which is a state-of-the-art object detection model [7]. On the other hand, for the recognition of license plate characters, the paper employs the LPRNet model [8], specifically designed for Indian license plate recognition. These algorithms form the core components of the ANPR system described in the paper.

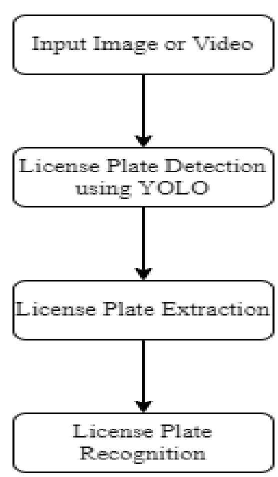

Fig. 1 Steps in an LPR System

II. LITERATURE REVIEW

In the paper [1], an improved model combining YOLOv5m and LPRnet is proposed. This model achieves high accuracy in recognizing license plates in various conditions such as frontal, tilted, low-light, and strong-light environments. The reported results include an accuracy of 99.49%, recall of 98.79%, and mAP (mean Average Precision) of 998.26. Additionally, the license plate recognition speed reaches 42 pictures per second.

Another paper [2] presents a method to detect the speed of a vehicle along with its license plate. This approach involves detecting the contours in the image and finding the bounding rectangle that encompasses all the contours. Then, image segmentation with Optical Character Recognition (OCR) using the K-Nearest Neighbors (KNN) and Support Vector Classifier (SVC) models is applied to recognize the characters on the license plate.

Paper [3] introduced a four-step method using a bounding box to detect license plates. It removes noise using filters and images are converted to grayscale. Edges in the images are found and then segmented for vehicle plate recognition. Template matching is used for character recognition. This Paper reached an extraction accuracy of 93.33%, a segmentation accuracy of 86.67% and a recognition accuracy is 93.33%.

In paper [4], a method is presented to detect license plates under varying illumination conditions. The process involves converting the image to grayscale, detecting the edges, and then applying dilation. The horizontally and vertically projected images are generated, followed by thresholding using the OTSU method to identify the license plate.

In paper [5], Gabor filtering is introduced as a preprocessing step that can be applied before segmentation. Additionally, template matching is introduced as a technique that can be utilized prior to character recognition. These approaches aim to enhance the performance of the overall license plate detection and recognition system.

III. DATASETS

A. Datasets Sources

The datasets are taken from three different sources. [9] The First source is the Indian vehicle license plate dataset from Kaggle. This dataset contains license plates from different states divided into three directories: State-wise_OLX, google_images, and video_images.

The second dataset, referred to as the AOLP dataset [10], comprises 2,049 images captured under diverse conditions such as different locations, times, traffic situations, and weather conditions. The entire dataset is divided into three subsets, each representing one of the major applications: access control, law enforcement, and road patrol, thereby offering a comprehensive range of samples for each application.

The third dataset, known as the Media Lab Dataset [6], consists of 706 images that exhibit various environmental conditions.

B. Datasets Preparation

To ensure compatibility with the algorithms for license plate detection and recognition, the datasets undergo preprocessing to obtain the necessary format. The preprocessing step for the YOLO format aligns the datasets with the specific requirements depicted as detailed in the reference [11].

Images for LPRNet need to be cropped such that only license plates are seen and each image should be named with its license Number.

After converting images to the required format, a dataset of 2785 images for license plate recognition and 4199 images for license plate localization is created. This dataset contains 1253 license plates from the AOLP dataset and 559 license plates from Media Lab Dataset. Also, it contains 1799 images for license plate localization from the AOLP dataset and 706 images for license plate localization from Media Lab Dataset.

IV. LICENSE PLATE DETECTION

Joseph Redmon and Ali Farhadi from the University of Washington developed YOLO (You Only Look Once)[12], a widely recognized model for object detection and image segmentation. YOLOv8 represents the state-of-the-art (SOTA) model, building upon the achievements of previous YOLO versions and incorporating new features and improvements to enhance performance and versatility.

A. YOLOv8 Model

On the COCO dataset, the YOLOv8x model achieves the highest mAP (mean Average Precision) of 53.9% among all models tested. This particular model has 68.2 million parameters. However, the YOLOv8l model utilized for this paper has 43.7 million parameters. The YOLOv8l model achieved an mAP of 52.9% on the COCO dataset.

B. Training Details

The model used in this paper, YOLOv8l, was trained from scratch for 50 epochs on an NVIDIA GeForce RTX 3060 GPU with 12288MiB memory. The training was conducted using images of width 640 pixels and height 640 pixels, with a batch size of 16 images. The training dataset consisted of 3628 valid images, while the validation dataset contained 370 valid images.

C. Results Analysis

After training, this method gets the best mAP50 of 0.99 with a precision of 0.99 and recall of 0.973 on the validation dataset. The model is of size 85MB. Its inference time is 19.3ms.

For the AOLP dataset, this paper gets the best mAP50 of 0.989 with a precision of 0.987 and recall of 0.989 with an inference time of 18.5ms.

For the Media Lab dataset, this paper gets the best mAP50 of 0.994 with a precision of 0.984 and recall of 0.977 with an inference time of 19.1ms.

Figure 2 shows predicted vs true bounding boxes and Table 1 shows the results.

Table 1 Detection Results with respect to Different Datasets

Dataset	**mAP50**	**Precision**	**Recall**
AOLP	0.989	0.987	0.989
Media Lab	0.994	0.984	0.977

True Bounding Boxes　　Predicted Bounding Boxes

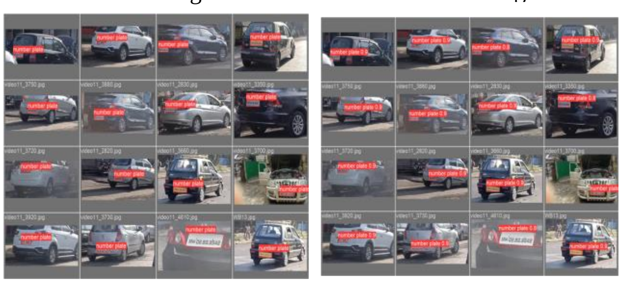

Fig. 2 True Bounding Boxes vs Predicted Bounding Boxes

V. LICENSE PLATE RECOGNITION

The Paper [8] describes an architecture to detect characters in a license plate for Chinese license plates. This Paper took that architecture and rebuilt it to recognize Indian license plates.

The architecture of the paper [8] does not work for the two-liner license plate. Code for the Paper [8] can be found here [13]. This Paper modifies architecture to detect those license plates also.

A. LPRNet Architecture

A Basic Block is created which is used in the main architecture of LPRNet. See Table 2 for the architecture. Table 3 shows the backbone of deep learning architecture for LPRNet. Table 4 is the container of the architecture. This paper took the maximum number of characters to be recognized as 13. A 2d array consisting of 38 rows each for a character from A-Z and 0-9 and two extra characters and 13 columns for the position of each character is created. The model predicts this 2d array and then gives the result.

Table 2 Basic Block

Type of Layer	**Parameters or Dimensions**
Input Layer	C_in × H × W C_out feature map
Convolution 2d	# C_out/4 1x1 stride 1
Convolution 2d	# C_out/4 3x1 strideh=1, padh=1
Convolution 2d	# C_out/4 1x3 stridew=1, padw=1
Convolution 2d	# C_out1x1 stride 1
Convolution 2d	C_out× H × W feature map

Table 3 Backbone Network Architecture

Type of Layer	**Parameters or Dimensions**
Input Layer	90x35 RGB image
Convolution 2d	#64 5x3 with stride 1x3
Batch Normalization 2d	features 64
Relu activation	
MaxPooling 3d	#64 1x5x3 with stride 1
Basic block	#128 3x3 with stride 1
Batch Normalization 2d	features 128
Relu activation	
MaxPooling 3d	#64 1x5x3 with stride (2, 1, 2)
Basic block	#256 3x3 with stride 1

Batch Normalization 2d	features 128
Relu activation	
Basic block	#512 3x3 with stride 1
Batch Normalization 2d	features 512
Relu activation	
Basic block	#512 3x3 with stride 1
MaxPooling 3d	#128 1x5x3 with stride (4, 1, 2)
Dropout	0.2 ratio
Convolution 2d	#512 5x4 with stride 1
Batch Normalization 2d	features 512
Relu activation	
Dropout	0.2 ratio
Convolution	# class number 3x3
Batch Normalization 2d	features class number
Relu activation	

Table 4 Container Network Architecture

Layer Type	Parameters/Dimensions
Convolution 2d	#class number 1x1
Sigmoid Activation	

B. Training Details

All experiments are done in PyTorch on NVIDIA GeForce RTX 3060, 12288MiB with 12th Gen Intel(R) Core(™) i7-12700F 2.10GHz.

This paper removed Chinese characters from the CHARS list in file load_data.py and added all uppercase alphabets with all numbers and a full stop and dash. Since the dataset can have up to 13 characters this paper took a maximum of 13 characters to predict. The model in this paper was trained using the Adam optimizer and employed the BCELoss as the chosen loss function. The training was conducted with a batch size of 1024 and up to 800 epochs. Data augmentation techniques were applied during the experiments, including image resizing to a size of 90x35.

C. Results Analysis

To analyze the results this paper took three measures. The first one tells how many characters have been predicted correctly which is accuracy (see Algorithm 1).

The second measure is edit distance which tells how much edit is required to reach the correction label (see Algorithm 2). The third measure is license plate accuracy which tells how many license plates have been predicted correctly (see Algorithm 3). See Table 5 for further results.

Table 5 in this paper illustrates that the accuracy of license plate recognition for the entire dataset is 82%. However, the accuracy is relatively low due to the average character length of 6 in the AOLP dataset, which is less than half of the average character length in the full dataset. One of the reasons is also inaccurate bounding boxes of license plates. This makes learning difficult for the model. But after fine-tuning the model for AOLP and Media Lab dataset the accuracies increased and edit distances decreased. For fine-tuning, this paper retrained both datasets for 50 epochs. See Figures 3 and 4 for loss and accuracy and Figure 5 for predictions.

Algorithm 1

Algorithm 1 Accuracy Calculation

Input: Actual and predicted labels.

Output: Accuracy.

```
SET t_acc TO 0
FOR k IN range(13):
    IF actual[k]==predicted[k]:
        t_acc+=1
SET t_acc TO t_acc/13
RETURN t_acc
```

Algorithm 2

Algorithm 2 Edit Distance Calculation

Input: Actual and predicted labels.

Output: Edit Distance.

```
SET actual TO actual.replace("-","")
SET predicted TO predicted.replace("-","")
SET s1 TO actual
SET s2 TO predicted
IF len(s1) > len(s2):
    SET s1, s2 TO s2, s1
SET distances TO range(len(s1) + 1)
FOR i2, c2 IN enumerate(s2):
  SET distances_ TO [i2+1]
  FOR i1, c1 IN enumerate(s1):
   IF c1 EQUALS c2:
     distances_.append(distances[i1])
   ELSE:
     distances_.append(1 + min((distances[i1], distances[i1 + 1], distances_[-1])))
 SET distances TO
 distances_distance=distances[1]
 RETURN distance
```

Algorithm 3

Algorithm 3 Number Plate Accuracy Calculation
Input: Actual and predicted labels.
Output: Number Plate Accuracy.

```
1: SET correct_predicted TO 0
2: IF actual==predicted:
3:    correct_predicted+=1
4: RETURN   correct_predicted
```

Table 5 Training Results with respect to different datasets

Dataset	**Accuracy**	**Edit Distance**	**Number Plate Accuracy**
Full Dataset	0.97	0.3	0.82
Indian Number Plate Dataset	0.99	0.01	0.99
AOLP Dataset	0.97	0.26	0.82
Media Lab Dataset	0.99	0.01	0.99

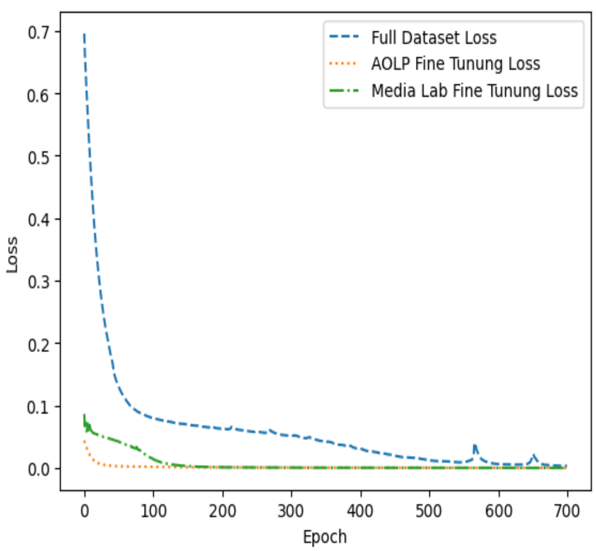

Fig. 3 Loss Curves

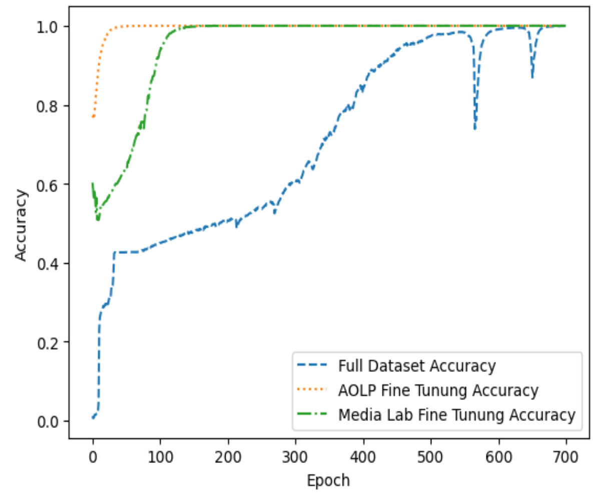

Fig. 4 Accuracy Curves

Fig. 5 Predictions

VI. EXPERIMENTAL RESULTS AND COMPARISONS

Here, the results of the experiments were compared with previous work conducted in other papers. See Tables 6 and 7 for comparative analysis.

Table 6 Performance Comparison between various License Plate Detection Methods

Algorithm	Type of LP Dataset	Images Count	LP detection Rate
CIP[14]	proprietary	80	91.25%
FCC[15]	proprietary	150	95.3%
Y. Yuan[16]	Caltech+ proprietary	3977	96.62%
J. Yepez[17]	Media-lab	741	98.45%
Two pass[18]	proprietary	9026	97.16%
RELIP[19]	proprietary	100	97%
SHGA[20]	Media-lab	336	98.5%
VEDA[21]	proprietary	664	91.65%
ESM[22]	proprietary	9825	99.6%
SCW[23]	Media-lab and proprietary	1334	96.5%
Tiny YOLOv4[7]	proprietary	10000	96.6%
Tiny YOLOv3[7]	proprietary	10000	93.0%
YOLOv2[7]	proprietary	10000	93.4%
YOLOv3[7]	proprietary	10000	97.6%
YOLOv4[7]	proprietary	10000	97.4%
Proposed YOLOv8l	Media-lab	705	99.4%
Proposed YOLOv8l	AOLP	1799	98.9%

Table 7 Performance Comparison of Different Methods for Character Recognition

Method	Number of Images	Dataset Type	Recognition Rate
Lee[24]	1248 characters	Proprietary	95.7%
Hsu[25]	2049	AOLP dataset	94%
Mail[26]	5689 characters	Proprietary	97.5%
Nijhuis[27]	10000	Proprietary	98.51%
Xie[28]	2049	AOLP dataset	99.25%
Anagnostopoulo[29]	741	Media Lab Dataset	85.50%
Proposed LPRNet	1253	AOLP dataset	97%
Proposed LPRNet	559	Media Lab Dataset	99%

VII. CONCLUSION

This study introduces a technique for detecting and recognizing Indian license plates and includes a comparative analysis of its performance with two other datasets. This Paper presented LPRNet for character recognition and achieved 99% accuracy for Indian license plates and for license plate detection this paper used the latest YOLOv8 and also achieved 99% accuracy. There is no question in the minds that LPRNet might operate in real-time on more specialized embedded low-power devices. LPRNet state-dict is saved which is nearly 8MB for further usage. For YOLOv8 since the large model is only 88MB this paper can easily deploy both LPRNet and YOLOv8 on the cloud for real-time detection.

References

[1] S. Luo and J. Liu, "Research on Car License Plate Recognition Based on Improved YOLOv5m and LPRNet," in IEEE Access, 2022, doi: 10.1109/ACCESS.2022.3203388.

[2] P. Devi Mahalakshmi | Dr. M. Babu "Automated License Plate detection and Speed estimation of Vehicle Using Machine Learning - Haar Classifier Algorithm" Published in International Journal of Trend in Scientific Research and Development (ijtsrd), ISSN: 2456-6470, Volume-4 | Issue-6, October 2020, pp.487-491.

[3] K. M. Babu and M. V. Raghunadh, "Vehicle license plate detection and recognition using bounding box method," 2016 International Conference on Advanced Communication Control and Computing Technologies (ICACCCT), 2016, pp. 106-110, doi: 10.1109/ICACCCT.2016.7831610.

[4] R. Boliwala and M. Pawar, "Automatic license plate detection for varying illumination conditions," 2016 International Conference on Communication and Signal Processing (ICCSP), 2016, pp. 0658-0661, doi: 10.1109/ICCSP.2016.7754224.

[5] A. Firasanti, T. E. Ramadhani, M. A. Bakri and E. A. Zaki Hamidi, "License Plate Detection Using OCR Method with Raspberry Pi," 2021 15th International Conference on Telecommunication Systems, Services, and Applications (TSSA), 2021, pp. 1-5, doi: 10.1109/TSSA52866.2021.9768252

[6] C. E. Anagnostopoulos, I. E. Anagnostopoulos, I. D. Psoroulas, V. Loumos, and E. Kayafas, Medialab lpr database, multimedia technology laboratory, national technical university of athens.

[7] Usama, M., Anwar, H., Shahid, M. M., Anwar, A., Anwar, S., & Hlavacs, H. (2022). Vehicle and License Plate Recognition with Novel Dataset for Toll Collection. arXiv preprint arXiv:2202.05631.

[8] Zherzdev, S., & Gruzdev, A. (2018). Lprnet: License plate recognition via deep neural networks. arXiv preprint arXiv:1806.10447.

[9] Kant Gaur. (n.d.). Indian vehicle license plate dataset. Retrieved March 16, 2023, from https://www.kaggle.com/datasets/saisirishan/indian-vehicle-dataset/code

[10] G. Hsu, J. Chen, and Y. Chung, Application-oriented license plate recognition, IEEE Transactions on Vehicular Technology, vol. 62, pp. 552561, Feb 2013

[11] Kathuria. (n.d.). How to Train YOLO v5 on a Custom Dataset. Retrieved March 16, 2023, from https://blog.paperspace.com/train-yolov5-custom-data/

[12] What is YOLOv8? The Ultimate Guide. (2023, January 11). Roboflow Blog. https://blog.roboflow.com/whats-new-in-yolov8/

[13] S. A. (2019, June 2). GitHub - sirius-ai/LPRNet_Pytorch: Pytorch Implementation For LPRNet, A High Performance And Lightweight License Plate Recognition Framework. GitHub. https://github.com/sirius-ai/LPRNet_Pytorch

[14] Lee, E. R., Kim, P. K., & Kim, H. J. (1994, November). Automatic recognition of a car license plate using color image processing. In *Proceedings of 1st international conference on image processing* (Vol. 2, pp. 301-305). IEEE.

[15] Yang, Y. Q., Bai, J., Tian, R. L., & Liu, N. (2005, August). A vehicle license plate recognition system based on fixed color collocation. In *2005 International Conference on Machine Learning and Cybernetics* (Vol. 9, pp. 5394-5397). IEEE.

[16] Li, H., Wang, P., & Shen, C. (2018). Toward end-to-end car license plate detection and recognition with deep neural networks. *IEEE Transactions on Intelligent Transportation Systems*, *20*(3), 1126-1136.

[17] Yepez, J., & Ko, S. B. (2018). Improved license plate localisation algorithm based on morphological operations. *IET Intelligent Transport Systems*, *12*(6), 542-549.

[18] Wen, Y., Lu, Y., Yan, J., Zhou, Z., von Deneen, K. M., & Shi, P. (2011). An algorithm for license plate recognition applied to intelligent transportation system. *IEEE Transactions on intelligent transportation systems*, *12*(3), 830-845.

[19] Haneda, K., & Hanaizumi, H. (2012, March). A flexible method for recognizing four-digit numbers on a license-plate in a video scene. In 2012 IEEE International Conference on Industrial Technology (pp. 112-116). IEEE.

[20] Abo-Samra, G. E. (2018). Application independent localisation of vehicle plate number using multi-window-size binarisation and semi-hybrid genetic algorithm. The Journal of Engineering, 2018(2), 104-116.

[21] Al-Ghaili, A. M., Mashohor, S., Ramli, A. R., & Ismail, A. (2012). Vertical-edge-based car-license-plate detection method. IEEE transactions on vehicular technology, 62(1), 26-38.

[22] Hongliang, B., & Changping, L. (2004, August). A hybrid license plate extraction method based on edge statistics and morphology. In Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004. (Vol. 2, pp. 831-834). IEEE.

[23] Anagnostopoulos, C. N. E., Anagnostopoulos, I. E., Loumos, V., & Kayafas, E. (2006). A license plate-recognition algorithm for intelligent transportation system applications. IEEE Transactions on Intelligent transportation systems, 7(3), 377-392.

[24] Lee, H. J., Chen, S. Y., & Wang, S. Z. (2004, August). Extraction and recognition of license plates of motorcycles and vehicles on highways. In Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004. (Vol. 4, pp. 356-359). IEEE.

[25] Hsu, G. S., Chen, J. C., & Chung, Y. Z. (2012). Application-oriented license plate recognition. IEEE transactions on vehicular technology, 62(2), 552-561.

[26] Mai, V. D., Miao, D., & Wang, R. (2012, August). Building a license plate recognition system for Vietnam tollbooth. In Proceedings of the 3rd Symposium on Information and Communication Technology (pp. 107-114).

[27] Nijhuis, J. A. G., Ter Brugge, M. H., Helmholt, K. A., Pluim, J. P., Spaanenburg, L., Venema, R. S., & Westenberg, M. A. (1995, November). Car license plate recognition with neural networks and fuzzy logic. In Proceedings of ICNN'95-international conference on neural networks (Vol. 5, pp. 2232-2236). IEEE.

[28] Xie, L., Ahmad, T., Jin, L., Liu, Y., & Zhang, S. (2018). A new CNN-based method for multi-directional car license plate detection. IEEE Transactions on Intelligent Transportation Systems, 19(2), 507-517.

[29] Anagnostopoulos, C. N. E., Anagnostopoulos, I. E., Loumos, V., & Kayafas, E. (2006). A license plate-recognition algorithm for intelligent transportation system applications. IEEE Transactions on Intelligent transportation systems, 7(3), 377-392