Deep Learning-based approach for Indian License Plate Recognition using Optical Character Recognition

Atharvraj Patil
Department of Electrical Engineering
College of Engineering, Pune
Pune, India

Jayesh Ingale
Department of Electrical Engineering
College of Engineering, Pune
Pune, India

Aditya Upadhye
Department of Electrical Engineering
College of Engineering, Pune
Pune, India

Sakshi Jaiswal
Department of Electrical Engineering
College of Engineering, Pune
Pune, India

Suhas Kakade
Department of Electrical Engineering
College of Engineering, Pune
Pune, India

Abhishek Bhatt
School of Data Science
Symbiosis Skills and Professional University
Pune, India

2023 IEEE Guwahati Subsection Conference (GCON) | 979-8-3503-3778-5/23/$31.00 ©2023 IEEE | DOI: 10.1109/GCON58516.2023.10183391

***Abstract*—The registration number (license number) of vehicles can be automatically read from digital images using a method called Optical Character Recognition (OCR). The enhanced license plate recognition method described in this paper uses a neural network trained on an alphanumeric dataset. The whole system can be categorized under two major modules, namely Plate Character Segmentation, and Plate Character Recognition. In this work, the pre-trained ResNet50 model is fine-tuned. The ResNet50 model is trained (fine-tuned) using the custom dataset after the top layer is removed and a new layer is added to the network. A Convolutional Neural Network (CNN) model is also trained on the same dataset and the results of both models are then compared. From the results, it can be inferred that the proposed CNN network with a validation accuracy of 95.60% outperformed the pre-trained ResNet50 network. Lastly, the lower computation time of the proposed network makes it extremely suitable for real-time applications.**

***Index Terms*—CNN, License plate recognition, Optical character recognition (OCR), ResNet50.**

I. Introduction

With the use of extracted license plate data, Optical Character Recognition technology will help in solving a variety of issues, such as access control, tolling, border, and traffic control, finding stolen vehicles, and effective parking management, among others, by automated means.

Machine learning algorithms have previously been used to monitor cars and license plates in a variety of different ways. However, these algorithms have limitations due to their processing complexity in unconstrained scenarios as discussed in [1] due to plate variations, environmental variations and camera mounting variations, etc, and mainly due to the technique/algorithm used for the License Plate Recognition system. Therefore, it is immediately necessary to create an autonomous system employing deep learning that will assist in accurately recognizing the car number plate details.

This paper proposes a recognition model in which the image of a license plate is precisely extracted and further given as input to the optical character recognition system. The algorithm segments the character on the plate and inputs them individually to the OCR technique for identification. Character recognition is done by two different models, CNN and fine-tuned RESNET-50, and their accuracy is compared. Finally, the model outputs the resultant string of characters. Python is used to develop and simulate this system, and real images are used for assessment.

Many studies have focused on finding various algorithms for OCR on license plate images. Conventional OCR techniques used for License Plate Detection usually fail on blurred/unclear or obstructing images of license plates and confusing characters. Earlier, this problem was categorized into (i) segmentation-free methods and (ii) segmenting first and then recognizing the segmented pictures. The former often provides the license plate (LP) to an OCR system or a convolutional neural network to accomplish the recognition task or uses license plate characteristics to extract plate characters directly to avoid segmentation. This also requires a fairly large dataset to train any model for recognition. The latter requires recognizing only a single character segmented out from the license plate. This segmented part needs to be accurate enough to hold only the character of our interest, and then pre-process this segmented image to be sent to OCR. [2] also shows how a two-stage process can be developed for the detection and recognition of license plates.

In recent years, some modern OCR techniques are being used. In paper [3] A modified You Only Look Once (YOLO) network is used for character segmentation and identification over the rectified LP. This study used synthetic and augmented data to deal with the LP characteristics of various geographic locations, thereby increasing the size of the training dataset. In paper [4], it uses Tesseract - OCR engine to predict the segmented characters from the license plate. While in the proposed method [5] For appraisal and comparison R-VGG16, R-SRCS, and R-CNN-L3 were used. They had identical output layers. Utilizing the fully connected layers and the softmax activation function, character prediction was done., and to predict the number of characters, another fully connected layer was constructed.

II. Dataset

We used an open-source public dataset [6] which consists of all numbers and capital alphabets, making a total of about 36 classes, categorically encoded as 0 to 35. There are about 37,200 images in this dataset with approximately 1,000 images per class. This dataset includes various challenging images which include blurred, tilted, rotated, and translated images. All images are of size 32x32 which has white characters on black background. Gray scaling is done on each of these images and Canny edge detection is applied on each of them in order to make it suitable for accurate and efficient prediction of character by recognition model. There are many similar characters and numbers which can be predicted wrongly by the model due to their similar features and sizes. Some of these pairs can be O and 0, 7 and 1, M and N, 5 and S, etc. To overcome this problem, highly similar images of these characters were excised from the dataset. This helped the model to learn only the prominent and dominant features of characters, thereby increasing the model's accuracy.

III. Methodology

In our proposed license plate recognition system, firstly license plate is extracted from a car image. Then extracted license plate is preprocessed and then character segmentation is carried out. Lastly, these segmented characters are passed on to the recognition model, which predicts the input characters. The details of each process are described further.

A. Image Acquisition

The license plate from a vehicle image was extracted using WPOD-NET presented by Sergio Silva et al. [7]. The extracted license plate image is further preprocessed before passing it on to the segmentation algorithm.

B. Image Preprocessing

Taking a localized license plate as input, various preprocessing is done on it for obtaining accurate results using OpenCV. Since the dataset consists of binarized images and for better separation of characters from the background, the color image is converted to a grayscale image as shown in figure 1. The image is then binarized using a threshold (1) in which pixels less than threshold value are converted to 0 whereas the ones greater than threshold values are converted to 1 [8]. As the segmentation algorithm uses Canny edge detection, morphological image processing is done on the image using erosion followed by dilation. To reduce the overall noise of the image, Gaussian blur (2) is applied.

$$\sigma_b^2(t) = w_1(t) * w_2(t)[u_1(t) - u_2(t)]^2 \tag{1}$$

where w_i are weights, u_i is mean and σ is the variance.

$$G(x, y) = \frac{1}{2\tau\sigma_2} e^{-(x_2+y_2)/2\sigma^2} \tag{2}$$

where σ is the standard deviation of the Gaussian distribution

Fig. 1. Image Preprocessing on the license plate.

C. Character Segmentation

Our next step is to find out the respective contours of characters in the image as shown in figure 2. One of the popular multistage edge detection algorithms is Canny edge detection. Edge detection is prone to noise, thus the first step is to remove it from the image. As a result, Gaussian Blur is applied. After this step, Canny edge detection is carried out.

Fig. 2. Character segmentation

After applying Canny edge detection on the image, contours and their respective bounding boxes are found. For shape analysis, object detection, and recognition, the contours are a useful tool. The CHAIN-APPROX-SIMPLE method is used as the contour approximation method. This method saves only boundary points of respective contours instead of taking all boundary points, as a result, memory is also saved. Also, for accurately specifying the hierarchy in the image, RETR-TREE is used as the contour retrieval mode to understand the parent and child relationship between the contours. After getting a list of all the contours, corresponding bounding boxes are drawn. Since there are only ten characters in the Indian number plate, we need to select only ten perfectly fitting bounding boxes across the characters. As a result, only those bounding boxes are selected whose height and width lie within certain limits of the defined percentage values. These defined percentage values were determined by the trial and error method.

IV. CLASSIFICATION AND RECOGNITION

A. RESNET-50

VGG and ResNet are the most frequently used algorithms for image classification. However as concluded in [9], the ResNet50 model proved to be better than both VGG16 and VGG19. A Residual Neural Network (ResNet) is a particular type of artificial neural network (ANN) that constructs a network by stacking residual blocks on top of one another. It is frequently used in many Computer Vision tasks as it proved to be successful in solving vanishing gradient problems. The ResNet50 variant of the ResNet model has 48 Convolution layers, one MaxPool layer, and one Average Pool layer. The ImageNet dataset, which contains roughly 1.4 million images and 1000 classifications, was used to train this network. By deleting the top layer and adding a new fully connected layer, the ResNet50 model is adjusted. The last layer consists of 36 nodes and the softmax activation function (4) is used as there are multiple classes for classification.

The architecture of fine-tuned ResNet50 is shown in figure 3. The above-mentioned dataset is split into training and validation sets in an 80:20 ratio. The ResNet50 model was compiled and fitted to the training dataset. A validation accuracy of 90.11% and training accuracy of 92.6% is obtained. The graph of accuracy against the number of epochs is plotted as shown in figure 4. From the graph, it seems that as the number of epochs increases, accuracy also increases. The training of the model was stopped using an early stopping callback at 100th epoch.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 resnet50 (Functional)       (None, 2048)              23587712

 flatten (Flatten)           (None, 2048)              0

 dense (Dense)               (None, 512)               1049088

 dense_1 (Dense)             (None, 36)                18468

=================================================================
Total params: 24,655,268
Trainable params: 1,067,556
Non-trainable params: 23,587,712
_________________________________________________________________
```

Fig. 3. ResNet50 architecture

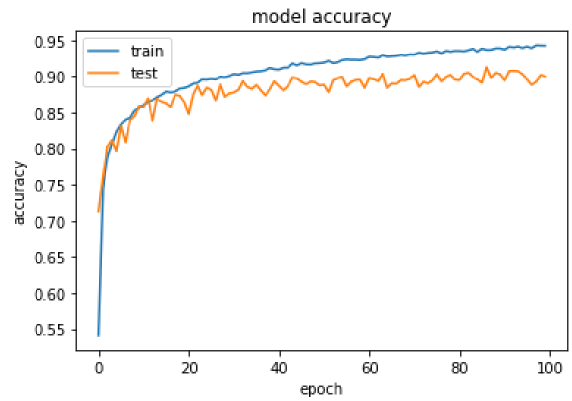

Fig. 4. ResNet50 model accuracy

B. Convolutional Neural Network

A model with simple architecture is used. Four convolutional layers are added initially, with "ReLu" (eq. 5) as an activation function, with the Max pooling layer having a window size of 4*4 to reduce its dimensionality. Furthermore, a dropout layer with a 0.2 dropout rate is incorporated to minimize overfitting. As a result, 80% of nodes are retained. This data is passed to the flattened layer, which converts the data to a single dimension. In the end, the first dense layer is added with the dimensionality of output space as 512, and a second dense layer is added at the output which uses Softmax as an activation function as this is a multi-class classification problem and has 36 outputs. This model is trained on the above dataset for 40 epochs. The loss function used here is sparse categorical cross-entropy:

$$L_{CE} = -\sum t_i log(p_i) \tag{3}$$

For n classes, where t_i is the truth label and p_i is the Softmax probability for i_{th} class.

$$softmax(z_i) = \frac{exp(z_i)}{\sum_j exp(z_j)} \tag{4}$$

where z = input vector
$exp(z_i)$ = standard exponential function for input vector
K = number of classes in the multi-class classifier
$exp(z_j)$ = standard exponential function for output vector

The equation (4) is used as an activation function in the output layer.

$$Relu = f(x) = max(0, x) \tag{5}$$

Figure 5 represents the architecture of the proposed CNN model. The validation accuracy obtained after compiling and fitting the model on the custom dataset is 95.4%. Also, the plot obtained for accuracy against a number of epochs is shown

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 32, 32, 16)        23248

 conv2d_1 (Conv2D)           (None, 32, 32, 32)        131104

 conv2d_2 (Conv2D)           (None, 32, 32, 64)        131136

 conv2d_3 (Conv2D)           (None, 32, 32, 64)        65600

 max_pooling2d (MaxPooling2D  (None, 8, 8, 64)         0
 )

 dropout (Dropout)           (None, 8, 8, 64)          0

 flatten_1 (Flatten)         (None, 4096)              0

 dense_2 (Dense)             (None, 512)               2097664

 dense_3 (Dense)             (None, 36)                18468

=================================================================
Total params: 2,467,220
Trainable params: 2,467,220
Non-trainable params: 0
_________________________________________________________________
```

Fig. 5. CNN architecture

in figure 6. From the graph, we can conclude that the test accuracy slightly increases with an increase in the number of epochs.

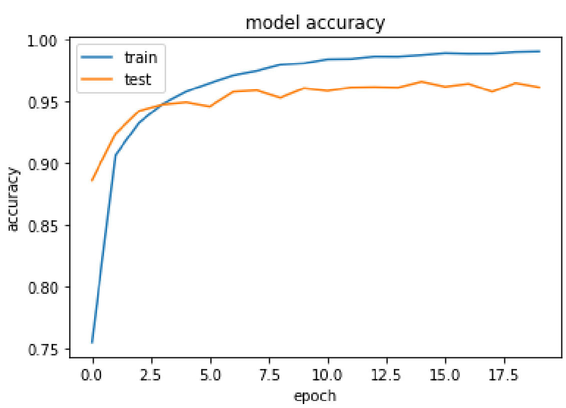

Fig. 6. CNN model accuracy

V. Results and Comparison

In this work, we presented a system for Optical Character Recognition(OCR) for character recognition of license plates. Two separate datasets were used, one which consisted of images preprocessed using canny edge detection which gave better results as compared to the original dataset.

We have used two different models, namely the proposed CNN model and fine-tuned ResNet50 model which are trained on custom datasets. We can conclude that the proposed CNN model outperformed the pre-trained fine-tuned ResNet50 model. A comparison table of both models can be seen in table I. The custom CNN is a better choice for optical character recognition because it has less complex architecture than the pre-trained ResNet50 model. Also, the ResNet50 model is more susceptible to overfitting, which results in them performing worse than our custom CNN model. The Custom CNN model doesn't require the addition of skip-level connections, which is required in ResNet50 models.

TABLE I
Accuracy Comparison

Sr.No	Model Used	Training Accuracy	Validation Accuracy
1	ResNet50	92.60%	90.11%
2	CNN	96.80%	95.60%

For every class, precision can be found out, the amount of true positives is compared to the sum of true positives and false positives. F1 scores per class can also be calculated, where 1.0 is the best score and 0 is the worst. F1 score and precision for each class are calculated for both the models and their comparison has been depicted in figure 7.

Class	Precision		f1 score	
	ResNet50	CNN	ResNet50	CNN
0	0.8	0.82	0.66	0.69
1	0.98	0.98	0.89	0.98
2	0.94	0.96	0.93	0.98
3	0.98	0.99	0.93	0.99
4	0.94	0.99	0.96	0.99
5	0.87	0.93	0.9	0.96
6	0.89	0.99	0.92	0.97
7	0.98	0.99	0.96	0.99
8	0.94	0.97	0.91	0.96
9	0.9	0.97	0.93	0.98
10	0.96	0.96	0.93	0.97
11	0.94	0.98	0.86	0.97
12	0.88	0.97	0.9	0.98
13	0.92	0.99	0.95	0.99
14	0.81	0.96	0.85	0.97
15	0.9	0.99	0.91	0.98
16	0.95	0.98	0.86	0.97
17	0.91	0.95	0.93	0.97
18	0.92	0.96	0.88	0.96
19	0.97	0.97	0.95	0.98
20	0.99	0.98	0.97	0.98
21	0.95	0.99	0.95	0.98
22	0.94	0.98	0.92	0.97
23	0.88	0.98	0.91	0.98
24	0.66	0.71	0.73	0.79
25	0.91	0.98	0.94	0.98
26	0.87	1	0.9	0.99
27	0.76	0.98	0.82	0.95
28	0.91	0.95	0.89	0.94
29	0.89	0.96	0.91	0.97
30	0.97	1	0.94	0.98
31	0.84	0.97	0.88	0.97
32	0.85	0.97	0.9	0.97
33	0.92	0.97	0.92	0.98
34	0.94	0.98	0.91	0.98
35	0.97	0.99	0.94	0.98

Fig. 7. Accuracy Metrics Comparison

VI. Conclusion

OCR has been thoroughly studied in the past. However, the use of OCR for character recognition in license plates has been a slightly difficult task due to many varying conditions. The main work of this paper is to compare the accuracy metrics of fine-tuned pre-trained ResNet50 model and the proposed CNN model, both trained on the same dataset. Since the weighted average of the F1 score of the proposed CNN model is greater as compared to that of ResNet50, we can conclude that the proposed CNN model outperforms fine-tuned pre-trained ResNet50 for character recognition. For future work, we aim to broaden our solution for recognizing characters on more challenging license plates.

References

[1] P. Mukhija, P. K. Dahiya and P. Priyanka, "Challenges in Automatic License Plate Recognition System: An Indian Scenario," 2021 Fourth International Conference on Computational Intelligence and Communication Technologies (CCICT), Sonepat, India, 2021, pp. 255-259, doi: 10.1109/CCICT53244.2021.00055.

[2] Padmasiri, Heshan, Jithmi Shashirangana, Dulani Meedeniya, Omer Rana, and Charith Perera. 2022. "Automated License Plate Recognition for Resource-Constrained Environments" Sensors 22, no. 4: 1434. https://doi.org/10.3390/s22041434.

[3] Zhenbo Xu, Wei Yang, Ajin Meng, Nanxue Lu, Huan Huang, Changchun Ying, and Liusheng Huang, "Towards End-to-End License Plate Detection and Recognition: A Large Dataset and Baseline", School of Computer Science and Technology, University of Science and Technology of China, Hefei, China(2018).

[4] V Gnanaprakash, N Kanthimathi, N Saranya, "Automatic number plate recognition using deep learning", Bannari Amman Institute of Technology, Department of Electronics and Communication Engineering, Sathyamangalam, Erode, Tamilnadu, India, ICCSSS 2020.

[5] J.-S. Chou and C.-H. Liu, "Automated Sensing System for Real-Time Recognition of Trucks in River Dredging Areas Using Computer Vision and Convolutional Deep Learning," Sensors, vol. 21, no. 2, p. 555, Jan. 2021.

[6] https://www.kaggle.com/datasets/sahajap99/characters-dataset-for-license-plate-recognition -anpr-with-python/

[7] S´ergio Montazzolli Silva, Cl´audio Rosito Jung, "License Plate Detection and Recognition in Unconstrained Scenarios", Institute of Informatics - Federal University of Rio Grande do Sul Porto Alegre, Brazil, ECCV 2018.

[8] V. Koval, V. Turchenko, V. Kochan, A. Sachenko and G. Markowsky, "Smart license plate recognition system based on image processing using neural network," Second IEEE International Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, 2020. Proceedings, 2003.

[9] S. Mascarenhas and M. Agarwal, "A comparison between VGG16, VGG19 and ResNet50 architecture frameworks for Image Classification," 2021 International Conference on Disruptive Technologies for Multi-Disciplinary Research and Applications (CENTCON), Bengaluru, India, 2021, pp. 96-99, doi: 10.1109/CENTCON52345.2021.9687944.

[10] Cheng-Hun Lin, Yong-Sin Lin, Wei-Chen Lui, "An Efficient License Plate Recognition System Using Convolutional Neural Networks", IEEE ICASI 2018.

[11] R.Mittal and A. Garg, "Text extraction using OCR: A Systematic Review," 2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA), 2020

[12] M. Jogin, Mohana, M. S. Madhulika, G. D. Divya, R. K. Meghana and S. Apoorva, "Feature Extraction using Convolution Neural Networks (CNN) and Deep Learning," 2018 3rd IEEE International Conference on Recent Trends in Electronics, Information and Communication Technology (RTEICT), 2018

[13] A. Reghunath, S. V. Nair and J. Shah, "Deep learning based Customized Model for Features Extraction," 2019 International Conference on Communication and Electronics Systems (ICCES), 2019

[14] Y. He, "Research on Text Detection and Recognition Based on OCR Recognition Technology," 2020 IEEE 3rd International Conference on Information Systems and Computer Aided Education (ICISCAE), 2020

[15] Ibtissam Slimani, Abdelmoghit Zaarane, Wahban Al Okaishi, Issam Atouf, Abdellatif Hamdoun,An automated license plate detection and recognition system based on wavelet decomposition and CNN,Array,Volume 8,2020.

[16] M. Jogin, Mohana, M. S. Madhulika, G. D. Divya, R. K. Meghana and S. Apoorva, "Feature Extraction using Convolution Neural Networks (CNN) and Deep Learning," 2018 3rd IEEE International Conference on Recent Trends in Electronics, Information and Communication Technology (RTEICT), 2018.

[17] Lubna, N. Mufti, and S. A. A. Shah, "Automatic Number Plate Recognition:A Detailed Survey of Relevant Algorithms," Sensors, vol. 21, no. 9, p. 3028, Apr. 2021.

[18] Shraddha S. Ghadage , Sagar R. Khedkar, 2019, A Review Paper on Automatic Number Plate Recognition System Using Machine Learning Algorithms, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH and TECHNOLOGY (IJERT) Volume 08, Issue 12 (December 2019),

[19] X. Luo, D. Ma, S. Jin, Y. Gong and D. Wang, "Queue Length Estimation for Signalized Intersections Using License Plate Recognition Data," in IEEE Intelligent Transportation Systems Magazine, vol. 11, no. 3, pp. 209-220, Fall 2019.