License Plate Recognition using Deep Learning: Detection to E-Challan Application

Neetu Narayan Rao Chikyal1, Lakshmi Sindhura Dubasi2, Sahaja Puligadda3, Vishnu Priya Samugari4

1Assistant Professor, Department of ECE, Vasavi College of Engineering, Hyderabad, Telangana, India

2, 3, 4 Department of ECE, Vasavi College of Engineering, Hyderabad, Telangana, India

***ABSTRACT* - Recognition of the license number plate of vehicles is significant for the control and surveillance systems. It plays a vital role in smart cities for investigation of stolen vehicles, traffic monitoring and vehicle management. Automobiles are usually recognized by number plates, as they contain a unique mix of numbers and alphabets. However, it is a tedious job for humans to recognize all the vehicle number plates manually. Hence, an efficient license number plate recognition system is built that consists of three main stages, namely, license plate localization, character segmentation, and character recognition. The system has mainly two model architectures, namely, mobilenetv2 architecture for character recognition and wpod-net architecture for license plate detection.**

***KEYWORDS* - plate detection, character recognition, OCR, wpod net, mobilenetv2, training, weights, deep learning, convolutional neural network, license plate, database, mails, SMTP, computer vision**

1. **INTRODUCTION**

India is a country with the third-largest road network in the world with around 295 million vehicles (registered). Registration of the vehicles will be done based on the license number plate given to that number. The license number is a unique combination of characters and numbers, using which the vehicle can be identified. Recognition of the license plates plays a significant role in vehicle control, such as toll collection on highways and parking lot management, traffic control, to reduce traffic violations, surveillance, or any other criminal acts. To overcome these problems, a license plate recognition system can be used.

An Automatic Number Plate recognition system (ANPR) was first invented at the police scientific development branch as a prototype in the UK, 1976. Working for 3 more years, the first contract was given to produce industrial systems, firstly for EMI electronics and then for computer recognition systems, UK. Later, as technologies emerged, optical character recognition using neural networks, machine learning, and deep learning became one of the most trending technologies used for license plate recognition. The efficiency has been increased from 1976 to date and has been increasing day by day.

License plate recognition changes from country to country, as the license plate guidelines are not constant at all places. So, the recognition also varies as per the country's guidelines provided for the license plate. Building an efficient license plate recognition system that localizes the license plate and recognizes the characters accurately, is crucial for the nation's traffic and vehicles management.

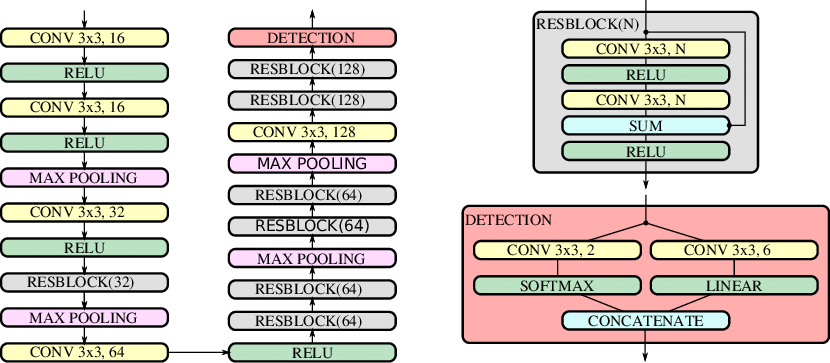
The algorithms used in the present systems used either pytesseract or only one model for the whole system. Using pytesseract to recognize characters gives results with a very low accuracy. This also increases the time complexity of the system and cannot determine the characters in various possibilities like blur images, poor lighting images due to which the efficiency of the system is comparatively low. In this project, the trained model is used to pre-process the image and segment the character followed by recognition of characters. Using only one model for the whole procedure increases the accuracy and training the model with datasets increases the recognition quality of the system. This makes the project even more efficient than the previously done projects.

This paper gives an organized structure of the proposed methodology. In the very first segment, abstract and keywords are described with information of the problem statement, the proposed solution, and commonly used words. This is followed by an introduction to the paper which is in section 1. Further, the literature review in section 2 deals with the architectures used for the project. In continuation, the materials and methods in section 3 talk about the methodology and components required. Then, the data required and the results acquired are mentioned in section 4. Section 5 deals with the discussions which include the positive outcomes and a few fail cases encountered. The acknowledgments are written in section 6. The final segment i.e. section 7 consists of improvements that can be added and the conclusion. All the resources and information used are provided in the form of links along with a description in section 8 at the end of the paper.

**2. LITERATURE REVIEW**

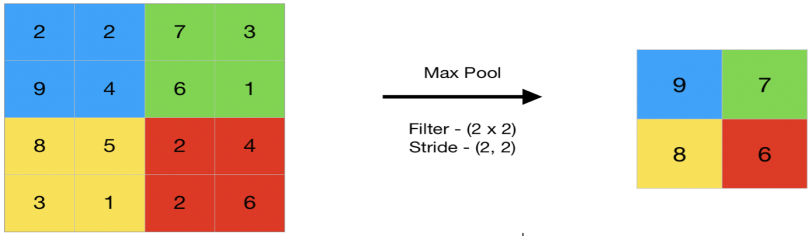
In the proposed model, two model architectures are used, namely, MobileNetV2 architecture and WPOD-NET architecture.

**2.1. WPOD-NET Architecture:** Warped Planar Object Detection Network (wpod-net) is an architecture starting from convolution layer to detection, has 21 convolutional layers of size 3 × 3, in which there are 14 inside residual blocks. There are ReLU activations, which are used in the entire network, except in the detection block.



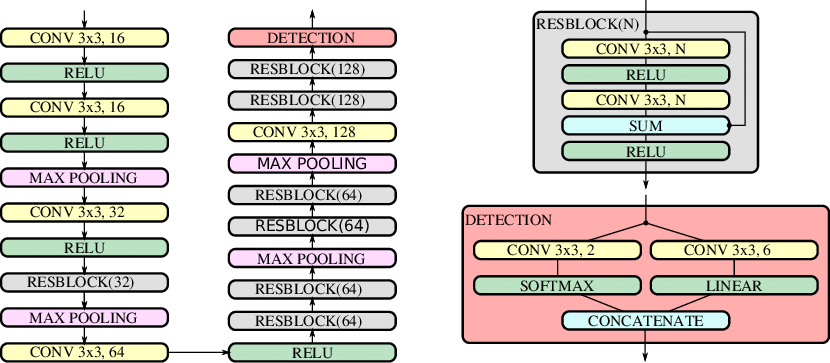
*Fig. 1: WPOD-NET architecture[1]*

Maximum pooling or max pooling is one of the pooling operations which calculates and selects the maximum value from the region of each feature map. Here, it has 4max-pooling layers of size 2×2 and stride 2, reducing the input dimensionality by a factor of 16.

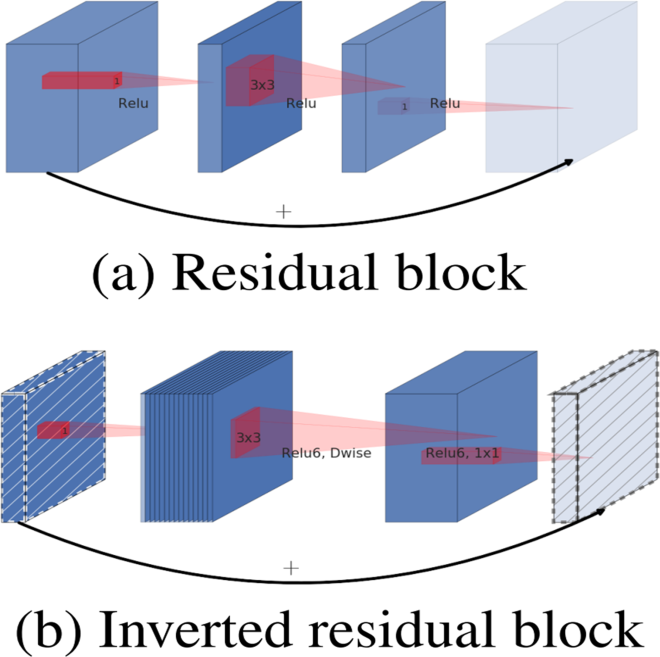


*Fig. 2: The output obtained after the max-pooling layer would be a feature map containing the maximum element of the previously featured map.*

Residual blocks are a notable case of highway networks without any gates in their skip connections. A residual block is a stack of layers from initial layers to last layers that flows the memory or information. The basic structure of the residual Block has a wide >> narrow >>wide approach with several required channels.

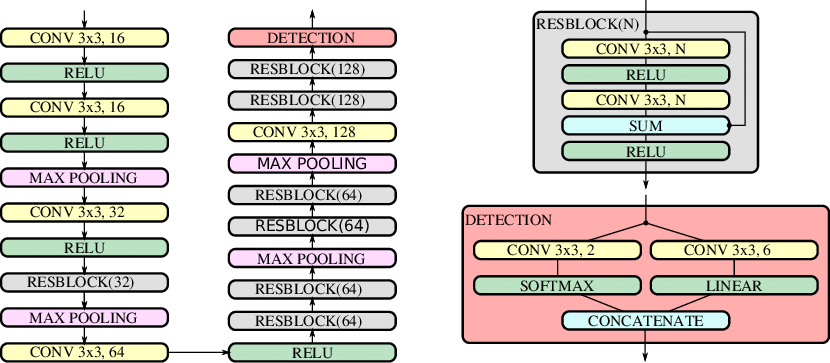


*Fig. 3(i): Residual block in WPOD-NET architecture[1]*

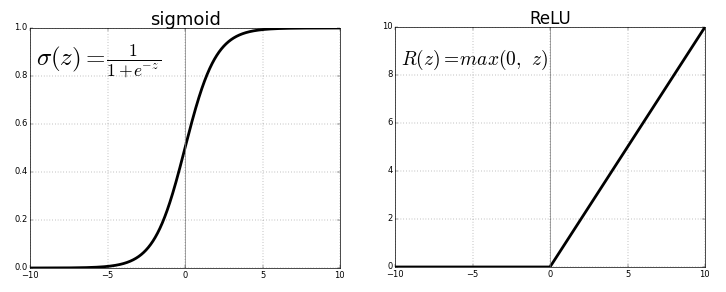


*Fig. 3(ii): A residual block structure*

The detection block has two parallel convolutional layers; one of the convolutional layers has the SoftMax function, which is an activation function and approximation of Max, used for inferring the probability. The other convolutional layer has the linear function, which is also an activation function; also called "no activation" used regressing the affine parameters. The detailed structure can be seen in fig 4(i).



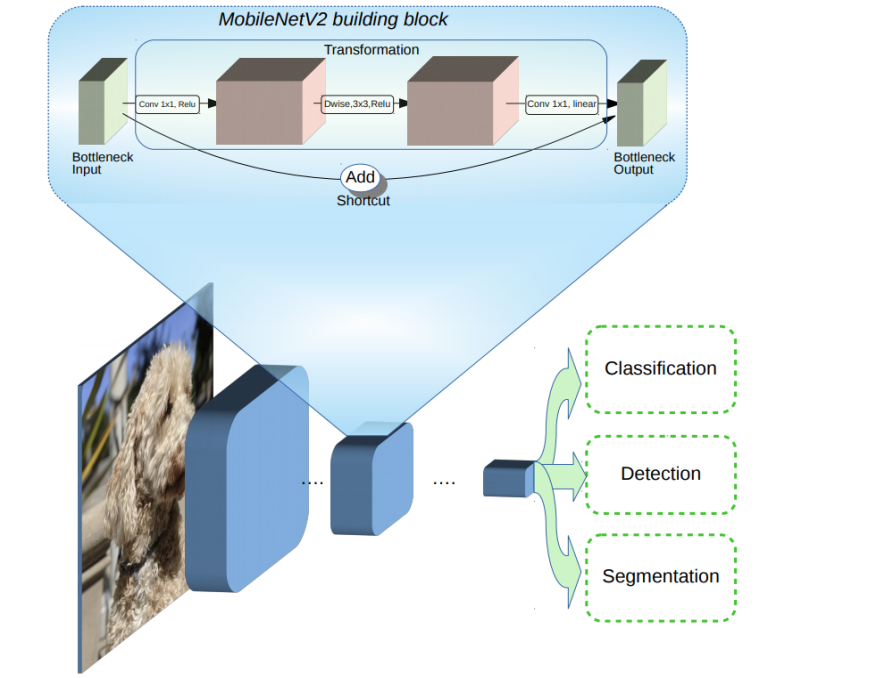
*Fig. 4(i): A detection block[1]*

**

*Fig. 4(ii): ReLU - Rectified Linear Activation Function*

As seen in the fig 4(ii), ReLU is a rectified linear activation function, mathematically represented as *R (z) = max(0, z)*. If *z* is less than zero then *R (z)* is equal to zero, if *z* is above or equal to zero then *R (z)* is equal to *z*. By using this function, architecture can learn fast and give better overall performance. In the proposed architecture, WPOD-net architecture is used to detect the license plate, which can be viewed in the coming sections.

**2.2. MobileNetV2 architecture:** MobilenetV2 is a neural network model for the resource-constrained environment which can handle problems like classification, detection, and segmentation; it achieves a perfect balance between performance results, implementation efficiency and reduces required memory. MobilenetV2 has three advanced structures used in it.

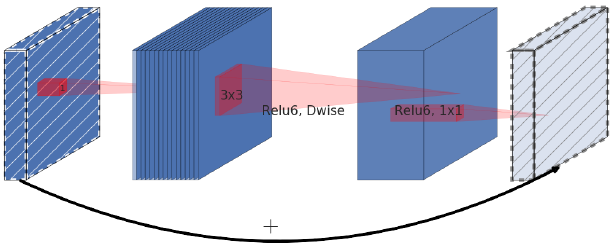


*Fig. 5: Overview of MobileNetV2 Architecture and blue blocks shows composite convolutional building blocks.*

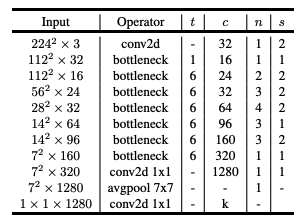
1. Depth wise separable convolution: It reduces the number of parameters when compared to the network with regular convolutions with the same depth as in the nets. This functionality results in deep neural networks that are lightweight. A depth wise separable convolution has two operations:
2. Depth wise convolution: Unlike the simple convolutions in which the convolution is done for all the channels together, in this type, convolution is applied to a single channel at a time.
3. Point-wise convolution: In this type, several 1\*1 filters are applied to all channels of the output of the previous phase.

2. Linear bottleneck: Here, linear activation is used in the bottleneck to reduce the loss of information.

3. Inverted residual: In this, an expansion layer is added at the beginning of the block and ReLU is used to add some non - linearity to the model and then, the input and output are added together and the summary of them is used as the output of the whole block as seen in fig.6. In contrast, an Inverted Residual Block follows a narrow >> wide >> narrow approach done to get better propagation of gradients.



*Fig. 6: An inverted residual block*



*Table 1: MobileNetV2 Overall Architecture*

Table 1 illustrates the overall architecture of MobileNetV2 which contains the initial 32 filtered fully convolutional layer, which is further followed by residual bottleneck layers that are 19 in number. As seen in the table, c depicts the count of output channels, n indicates the repeating number, s refers to stride, and for spatial convolution, 3×3 kernels are used. The

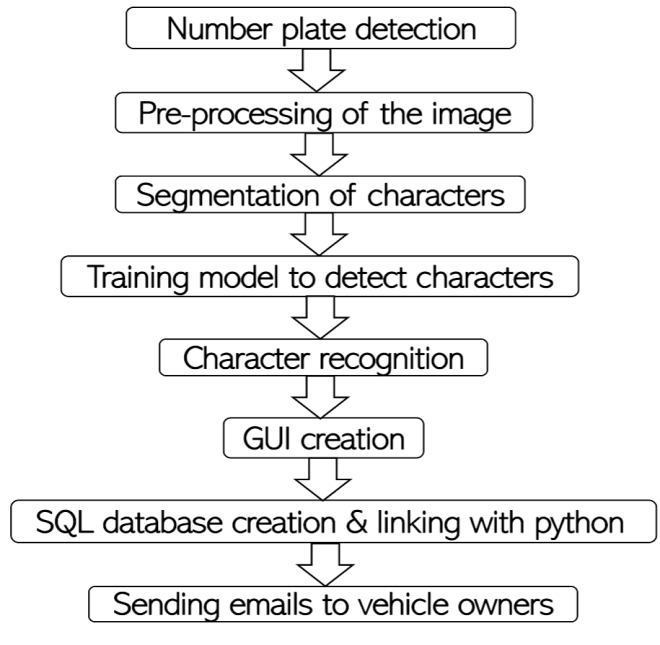
width multiplier of a primary network has of 1, 224×224. It uses 3.4 million parameters and has a computational cost of 300 million multiply adds. The model size has number of parameters between 1.7 million and 6.9 million. To recognize the cropped characters, the model of a neural network is created using MobileNetV2 architecture which can be seen in a further paper.

**3. METHODS AND MATERIALS**

The materials used are python libraries, like OpenCV, matplotlib, Keras, TensorFlow, Tkinter, MySQL.connector, smptlib​.

**3.1. Our proposed methodology:**

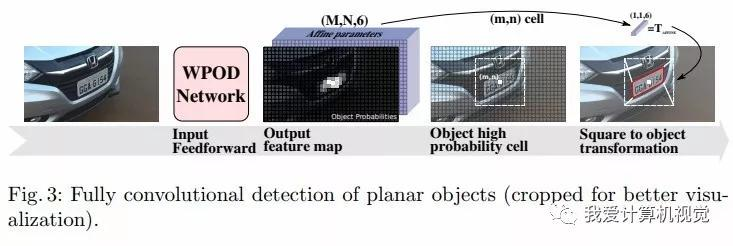
The working of the proposed model takes place in the sequence of steps shown below:



*Fig. 7: Flowchart of the working process*

**3.1.1. License plate detection**

A pre-trained model named WPOD-NET is loaded. To begin with, the resized vehicle image is given as the input to the WPOD net model, and the complete image is divided into (m,n) cells where the network will search for the cells which have a high target probability of license plate. A point (m,n) will be selected, and by referring to that point, an imaginary box will be drawn, and further, it selects an area, which is converted into a license plate as seen in fig 8.



*Fig. 8: The detection process in the wpod-net model[1]*

The WPOD-NET model is used to first detect and later, crop the license plate. The results obtained in this step are visible in fig.9 and fig.10.



*Fig. 9: Output of license plate detection*



*Fig. 10: The license plate obtained after cropping*

**3.1.2. Pre-processing of the image**

The preprocessing of the cropped number plate is done in four steps. The first step is “grayscale” which is used to convert an image from one color space to gray, the second step is “Gaussian blur” which helps in reducing the noise by smoothening. In the next step, this blurred image is converted to a binary image by inverse thresholding, and then in the fourth step, to have good recognition, dilation is performed. This dilated image is utilized in the next stage i.e., for performing segmentation on the characters. The results obtained after pre-processing of the image can be seen in fig.11.



*Fig. 11: Output of pre-processing of cropped license plate*

**3.1.3. Segmentation of characters**

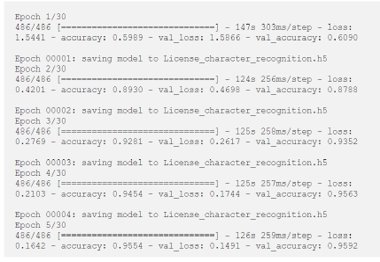
To segment the characters, first, the contours need to be found, that is the closed figures, then they should be sorted in ascending order based on the x-coordinate so that the order of the license plate characters is not lost and further, a bounding box can be drawn for each character contour. The cropped characters are plotted in a figure as seen in fig.12.



*Fig. 12: Output of segmentation of characters*

**3.1.4. Training model to recognize characters**

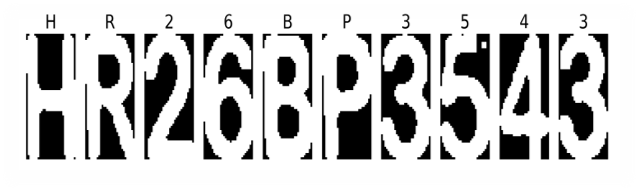
To recognize the characters with more efficiency, a neural network model is used. Here, a model based on MobileNetV2 architecture is created and then trained. To do so, first, the dataset-specific to this project is loaded. The dataset has 37,623 images spread across 36 classes. Then, the model is created with MobileNetV2 architecture. For training, the initial hyper parameters like learning rate and decay value, 30 epochs, and a batch size of 64 are initialized. A sample screenshot of the training process can be seen in fig.13. After training, the model is saved along with its weights and labels to the system which is later used to recognize the characters.



*Fig. 13: Training process*

**3.1.5. Character Recognition**

The trained model from the previous step is loaded and the characters are predicted one by one. To view the results, each character image along with the recognized character is displayed in an image as shown in fig-14. Finally, the license plate number is obtained in machine editable format, which can be further used to develop an application. Here, an E-challan application is developed which uses the previously recognized characters.



*Fig. 14: Characters recognized license plate*

**3.1.6. GUI creation**

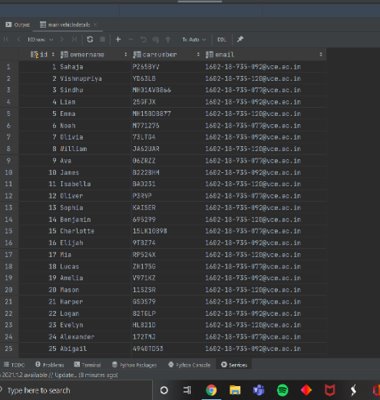
An interface is created by using the Tkinter module, wherein the user can upload a vehicle image consisting of a number plate, and by clicking on specific buttons, the user can access the text of that image which is machine editable. A GUI, which stands for Graphical User Interface, is made so that a person can easily access the application without worrying about the backend code. In this GUI, a title named "License Number Plate Recognition and E-Challan" is created, an icon for the window is set, labels for the detected text are created, and buttons are placed as "Browse Image", "Detect License Plate", and "Send Mail". The screenshot of this GUI application created is seen in fig.15.



*Fig. 15: GUI application*

**3.1.7. SQL database creation and linking with python**

A database of 100 records is created, each containing owner details like name, vehicle number, and email id, by using sqlite3 and later, connecting it with python to use this database for the application i.e. e-challan. The screenshot of the sample database which is created can be seen in fig.16 and to view the database, [click here](https://vasavicollegeofenginee-my.sharepoint.com:443/:x:/g/personal/1602-18-735-120_vce_ac_in/EdClwVdHVKNJtBuAFDryL48BeTCqhVT_L9WPknlxQnuBNg?email=1602-18-735-092%40vce.ac.in&e=4%3a7aPFrl&at=9).



*Fig. 16: Database created*

**3.1.8. Sending emails to vehicle owners**

Previously, the GUI interface is seen, on that window, there is a "send mail" button which when pressed, the mail is sent to the respective owners using a protocol named SMTP (Simple Mail Transfer Protocol). In the sample mail which is created, we have attached the respective owner vehicle image and mentioned the subject as "Reg: Traffic Rules Violation" and the body as

"Sir/Madam,

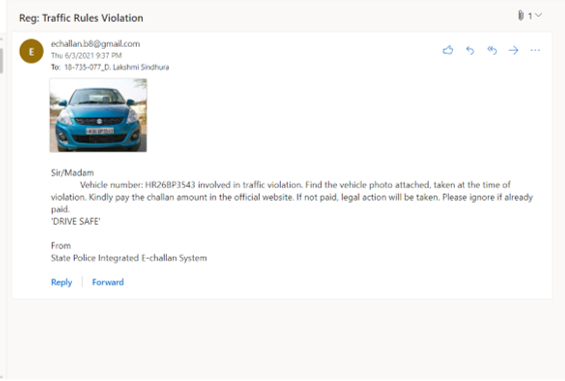
Vehicle number: {} was involved in traffic violation recently. Find the vehicle photo attached, taken at the time of the violation. Kindly pay the challan amount on the official website. If not paid within a month, legal action will be taken. Please ignore this mail, if you already paid.

'DRIVE SAFE'

From

State Police Integrated E-challan System”

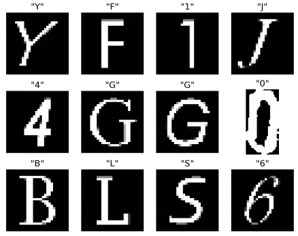
The screenshot of the sample mail which we have sent can be seen in fig.17. This is the last step in the working.

*Fig. 17: Sample mail*

Thus, an efficient license number plate recognition system can be developed by following the series of steps mentioned above. Now, a brief overview of the data used and the results obtained are given in the below section.

**4. DATA AND RESULTS**

To train the model built with mobilenetv2 architecture, a dataset of 37,623 images is given as input for training and validation, as seen in fig.19. The dataset contains images of alphabets and numbers, (A-Z) and (0-9) respectively, some of which are shown in fig.18.

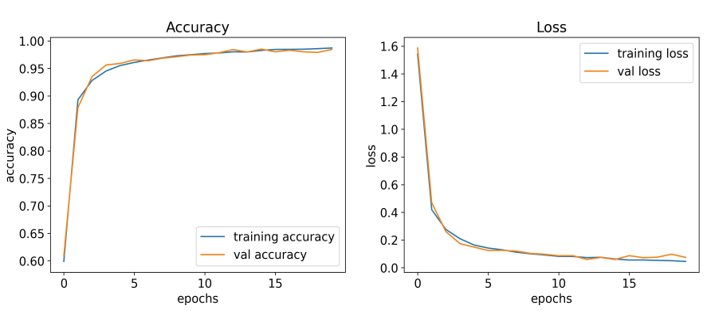


*Fig. 18: Output of a random list of images from the dataset*



*Fig. 19: The number of images and classes*

Using LabelEncoder, these alphabets and numbers are considered as 36 classes after performing one-hot encoding. A validation ratio needs to be given for the training model for both pieces of training as well as testing. In the proposed model, a ratio of 9:1 is considered for validation and testing respectively. In other words, 90% of images of the dataset are used for training purposes and the remaining 10% of images are used for validation. After loading all the contents, 30 epochs are given to the model, where one epoch indicates that each image in the dataset has a chance to update the internal model parameters for once. The iteration stops when there is no change in a given parameter, like, validation accuracy, validation loss, training accuracy, or training loss.

*Fig. 20: Training and testing accuracy and loss graphs w.r.t. epochs*

When the iterations are completed, a trained model will be created which is stored in a JSON file format. Also, the results are plotted on graphs, that is, training and validation accuracies and losses as parameters as seen in fig.20. Apart from the graphs, the actual values obtained after training the model are 98.03% accuracy and 5.86% loss in training, whereas 97.34% accuracy and 8.69% loss in validation.

Later, for the E-Challan application, a database of 100 records is created to test the vehicle images with the trained model, containing details of owner name, number plate, and email, using which a notifying mail will be sent to that respective vehicle owner.

**5. DISCUSSIONS**

The system, hence built, proved to be efficient as it could recognize number plates accurately. It detected small number plates correctly. Moreover, it produced exact results in the cases where the image is clear and has sufficient light. Also, it gave proper results independent of the color of the number plate. This system is also capable of detecting multiple number plates from a single image. Nevertheless, this model gave incorrect outputs in a few cases as mentioned below.

*Fail cases:* When there were poor lighting conditions, lights behind the numbers, or blurry image, the model could not predict the characters as expected. Variation in lighting affects the image which needs to be pre-processed, resulting in the loss of characters or wrong detection.

**6. ACKNOWLEDGEMENTS**

Our sincere thanks to Vasavi College of Engineering and the lecturers of the ECE department for providing access to the information and resources, and offering their valuable guidance, without which this project could not have been completed.

**7. CONCLUSION**

In this project, an efficient license plate recognition system is built with a pre-trained model to detect the license plate in the image and another trained model to recognize the characters after pre-processing the license plate after detection. The system reduces the manual work and hence, provides greater accuracy than a human can. Deep learning techniques are implemented to increase the recognizing ability of the model which resulted in 97.34% accuracy. Further advancements made in the project can make it more feasible and thus, give precise outputs.

***Future Scope***: The previously mentioned fail cases can be eliminated by improving the accuracy and efficiency of the model to detect poor-quality images and thus, overcome the hurdle of recognizing the images of poor lighting, and images that are blurry. In the current project, features like attaching the image taken while violating traffic rules and the license plate number are added to the email. In the future, other advanced components like time, date, place and reason of violation can be added in the mail, which increases the readability.

**8.   REFERENCES**

[1]Montazzolli, Sérgio & Jung, Claudio. (2018). License Plate Detection and Recognition in Unconstrained Scenarios. Available:<https://openaccess.thecvf.com/content_ECCV_2018/html/Sergio_Silva_License_Plate_Detection_ECCV_2018_paper.html>

[2] “Automatic License Number Plate Recognition using Deep Learning”, TechVidvan, [online]. Available: <https://techvidvan.com/tutorials/python-project-license-number-plate-recognition/>

[3] “Detect and Recognize Vehicle’s License Plate with Machine Learning and Python - Part 1: Detection License Plate with Wpod-Net”, [Quang Nguyen](https://medium.com/@quangnhatnguyenle?source=post_page-----795fda47e922--------------------------------), April 11, 2020, [online].Available:<https://medium.com/@quangnhatnguyenle/detect-and-recognize-vehicles-license-plate-with-machine-learning-and-python-part-1-detection-795fda47e922>

[4]“Plate\_detect\_and\_recognize'', Dataset and wpod-net pre-trained model, [quangnhat185](https://github.com/quangnhat185), [online]. Available: <https://github.com/quangnhat185/Plate_detect_and_recognize>

[5] “Sending Emails With Python”, [Joska de Langen](https://realpython.com/python-send-email/#author), [online]. Available:<https://realpython.com/python-send-email/>

[6] Fig. 4(ii): ReLU - Rectified Linear Activation Function, ”Activation functions in neural networks” , SAGAR SHARMA, September 6, 2017 [online]. Available: <https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6>

[7] Fig. 5: Overview of MobileNetV2 Architecture and blue blocks shows composite convolutional building blocks, ”MobileNetV2: The Next Generation of On-Device Computer Vision Networks”, Mark Sandler and Andrew Howard, Google Research, Tuesday, April 3, 2018, [online]. Available: <https://ai.googleblog.com/2018/04/mobilenetv2-next-generation-of-on.html>

[8] Table 1: MobileNetV2 Overall Architecture, Le Huy Hien, Ngo & V.H., Nguyen. (2020). “Automatic Plant Image Identification of Vietnamese species using Deep Learning Models. International Journal of Emerging Trends & Technology in Computer Science. 68. 25-31. 10.14445/22315381/IJETT-V68I4P205S [online]. Available: <https://www.researchgate.net/publication/341151927_Automatic_Plant_Image_Identification_of_Vietnamese_species_using_Deep_Learning_Models>

[9] Fig. 2: The output obtained after the max-pooling layer would be a feature map containing the maximum element of the previously featured map, “CNN | Introduction to Pooling Layer”, Geeksforgeeks, 26 Aug, 2019 [online]. Available: <https://www.geeksforgeeks.org/cnn-introduction-to-pooling-layer/#:~:text=Max%20pooling%20is%20a%20pooling,of%20the%20previous%20feature%20map.>

[10] Fig. 3(ii): A residual block structure, Fig. 6: An inverted residual block, “An artificial intelligent diagnostic system on mobile Android terminals for cholelithiasis by lightweight convolutional neural network “, Shanchen Pang, Shuo Wang, Alfonso Rodríguez-Patón, Pibao Li, Xun Wang, September 12, 2019, [online]. Available: <https://journals.plos.org/plosone/article/figure?id=10.1371/journal.pone.0221720.g007>

[11] Pallavi Khare, Rupesh Dudhe, Amolsing Chungade, Ajit Naykinde, 2015, Advanced License Number Plate Recognition System, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) ICONECT – 2015 (Volume 3 – Issue 06). Available: <https://www.ijert.org/advanced-license-number-plate-recognition-system>

[12] “Automatic number plate recognition”, Wikipedia, [online].Available:<https://en.wikipedia.org/wiki/Automatic_number-plate_recognition#:~:text=ANPR%20was%20invented%20in%201976,Jenoptik)%20in%20Wokingham%2C%20UK.>

[13] “An ALPR System-based Deep Networks for the Detection and Recognition”,Mouad Bensouilah, Mohamed Nadjib Zennir and Mokhtar Taffar,Department of Computer Science, Jijel University, BP 98, Ouled Aissa, 18000, Jijel, Algeria.Available:

<https://www.scitepress.org/Papers/2021/102292/102292.pdf>

[14]  Fig. 9: Output of license plate detection , “Number Plate Detection with Python”, AMAN KHARWAL, December 24, 2020,[online].Available:<https://thecleverprogrammer.com/2020/12/24/number-plate-detection-with-python/?hcb=1>