Vehicle Logging System

Nithin Kumar M R[1], Sanjay C Ravi[2], Vinu Suhas Dilma[3]

Under the guidance of **Mr. Duddela Sai Prashanth[4]**

[1,2,3,4]*Department of Computer Science and Engineering, Sahyadri College of Engineering & Management, Adyar, Mangaluru(575007), Karnataka, India*

---***---

1. INTRODUCTION

The ideology proposed paper was inspired by the challenges that security guards encounter; they may not be able to record the logging information of vehicles due to a variety of factors, including poor vision, bad light conditions, multiple vehicles traveling at once, and incorrect interpretation at an organization's entrance gate. The major goal of this model is to digitize an organization's vehicle logging details, which are currently done manually by maintaining a record book. This model, created for security control purposes, has a dashboard filter that displays logging information about a vehicle according to a variety of criteria, including the vehicle's license plate number, the date and time the vehicle entered the organization, and the frequency with which the vehicle visited the organization.

The camera that is installed at the organization's entrance gate is used by the designed model to detect the license plates of moving vehicles that are entering or leaving the building. The object (vehicle) near the entrance gate is detected by the model using YOLOv3. During license plate extraction, it is necessary to isolate and classify individual characters using methods like optical character recognition (OCR) and finding rectangular contours. Character identification is carried out in the last stage utilizing deep learning classifiers to identify the segmented characters. Since CNNs operate best with images, we chose CNN for this project. The database will keep the extracted characters for future review and searching purposes.

This paper was developed to work in real time on Jetson Nano with 40 FPS, which made use of Raspberry Pi Camera Module v2 (IMX219) and It is capable of detecting all types of vehicle number plates in India.

2. RELATED WORK

Recognition of vehicle registration plates can be broadly divided into two main categories – traditional computer vision techniques and classifier methods such as statistical (SVM) and deep learning (CNN) methods.

2.1 LICENSE PLATE DETECTION

A cascade framework consisting of two convolutional neural networks was used to detect the license plate region in [1]. The first convolutional neural network (CNN) in the cascade framework was responsible for extracting regions containing text from the image while the second CNN classified text regions as general text. In [3] each image was divided into sub regions which were fed independently into a CNN. The CNN outputs a score which indicates how likely a specific sub-region is to contain a license plate. A cascaded approach R-CNN was used in [4] for license plate localization. A weak SNOW classifier generates license plate regions which were fed into a strong CNN classifier (Alex Net) to be scrutinized. Images that failed to pass a confidence test were fed into another CNN (Alex Net) which identified the reason for failure. Failure identification can help in identification of probable problems which helps in troubleshooting the ALPR system at the earliest convenience. A simple CNN was used in [8] for license plate detection. Region Proposal Network and Box Regression layer was used to detect the license plate location. Many real-time LPD approaches [2], [6], [7], [10] used YOLO networks or its modified versions due to its fast inference speed. Most deep neural networks struggle at detecting small objects. To tackle this issue, the study in [6] trained FAST-YOLO to detect the front view of a car. The detected front view was cropped and fed into the same network to detect the license plate. The license plate appears bigger in the cropped front view image which makes it easier for the network to detect the license plate. The study in [2] modified YOLO and YOLO9000 for application in license plate detection and achieved better accuracy than the original YOLO networks. A multidirectional license plate detection method [7] used two networks to detect rotated LPs. The first network, referred to as the attention network, detected the license plate region while the second network MD-YOLO (modified version of YOLO) detected the rotated bounding box of the license plate. Two YOLO networks, one for vehicle detection and the other for license plate detection, were used in [10]. A CNN called WPOD-NET that regresses coefficients of an affine transformation for detecting and unwarping distorted license plates. The input to WPOD-NET is the image of a vehicle which was detected by YOLO.

2.2 LICENSE PLATE RECOGNITION

Many studies have tried to unify the subtasks (character segmentation and character recognition) of license plate recognition. The study in [1] proposed the use of a recurrent neural network (RNN) with long short-term memory (LSTM) and Connectionist Temporal Classification (CTC) for license plate recognition and considered license plate recognition as a sequence labelling problem making the character segmentation step unnecessary. On the other hand, the study in [5] used Bidirectional RNNs (BRNNs) with CTC loss for license plate recognition. The study in [4] employed a sweeping OCR technique that swept an OCR classifier across the license plate image and localized the characters by utilizing a probabilistic inference method based on hidden Markov models (HMMs). Viterbi decoding was used for determining the most likely code sequence using a language model. A slightly modified version of YOLO called CR-NET along with a heuristic method was used in [6] for Brazilian license plate recognition. Inception Blocks was used for real time license plate recognition in [8]. The work in [9] combined feature maps after ROI pooling and fed them into subsequent classifiers for license plate recognition.

A different approach which segmented characters from a license plate using semantic segmentation followed by a counting refinement stage was adopted in [10]. A modified version of DeepLabv2 ResNet-101 model was used for semantic segmentation. The counting refinement stage extracted character regions and fed them into Alex Net for character counting. A sliding-window single class detector via tiny YOLO classifiers was used in [11] for license plate recognition.

3. PROPOSED METHODOLOGY

The system's primary goal is to recognise the license plate of a moving vehicle. Capturing the license plate and identifying the characters and numerical on the plate is referred to as recognition. In general, we see license plates in white and yellow with black lettering. The block diagram depicts the entire recognition process. The most difficult challenge is detecting moving vehicle licence plates. If there are numerous vehicles at the same time, a human eye may not catch the precise data, resulting in inaccurate data being entered into the log book. To avoid such a catastrophe, we must create algorithms that record and maintain precise data.

3.1 SYSTEM ARCHITECTURE

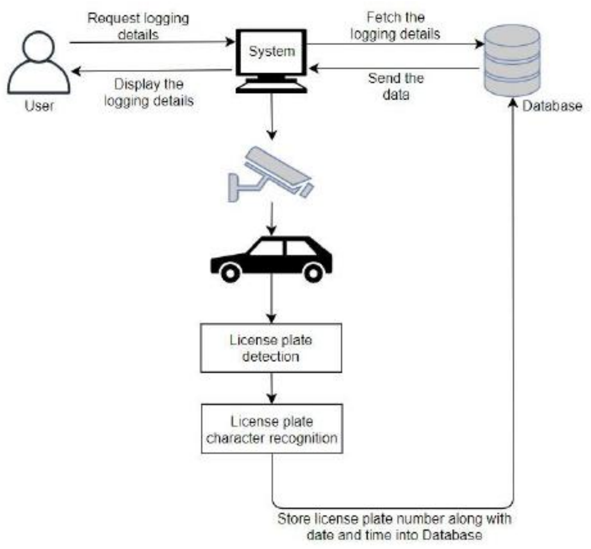

Request logging details
Fetch the logging details
System
User
Display the logging details
Send the data
Database
License plate detection
License plate character recognition
Store license plate number along with date and time into Database

Figure 1: System Architecture of Proposed System

3.2 FLOW CHART

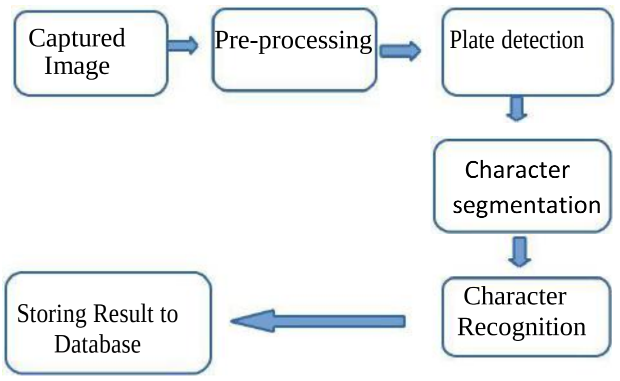


Figure 2: Block Diagram of Proposed System

Step 1: Capturing Vehicle Image

To capture the images of vehicles, we must use high resolution LPR cameras that are designed to recognise them at the gate and capture an image of the vehicle. We have used jetson IMX219 camera for capturing the image.

Step 2: Pre-Processing

To improve the image quality, the image captured by the camera must be processed. Binarization, background subtraction, sharpening or smoothing, and noise reduction for brightness were included in this process. From the input RGB image, it has to be converted to grayscale and the 8 bit grey value is calculated. After grayscale conversion, the morphological operations begin, which starts with the dilation of an image. After the dilation of the image, we erode the image and apply the median filtering technique, which is the basis of noise reduction.

```
Grayscale(Y) = (R + G + B) / 3

Y= R / 3 + G / 3 + B / 3

Y = 0.299R + 0.587G + 0.114B
```

We used a median filtering technique to reduce noise. We have used *3x3* masks to get eight neighbours of a pixel and their corresponding grey value. After applying the median filtering technique, we erode the image by a structural element in the form of any shape. We go on to find the morphological gradient for edge enhancement, and then we convert the class to double for brightening the edges and perform the convolution of the double image.

Step 3: License Plate Detection

The primary goal of this stage is to constrain or restrict the search area for the character recognition step. The localized license plate plays a crucial role in reducing the number of false character detections outside the license plate area. We used YOLO network architecture in this stage for license plate detection. The YOLO object detector works by splitting an image into an $S \times S$ sized grid. For each grid cell, the K number of bounding boxes along with the confidence scores $Pr(Object)\ IOU^{truth}$ are predicted. The confidence score indicates the extent to which the model is confident about the existence of an object. Confidence score is zero in the absence of an object. In the presence of an object, the confidence score is equal to the IOU between the predicted and the ground truth bounding box. A conditional class probability score $Pr(Classi\ |Object)$ is also predicted for each grid cell containing an object. The class-specific confidence score for each box is calculated using Equation 3 and is encoded as an $S \times S \times (K \times (5 + C))$ tensor.

$$Pr\ (Class_i|Object) * Pr(Object) * IOU^{truth} = Pr\ (Class_i) * IOU^{truth} \qquad (3)$$

The main reason for selection of YOLO is its real time performance and high accuracy. Specifically, we utilize the latest YOLOv3 instead of YOLOv2 even though YOLOv2 achieves higher frames per second (fps). This is because YOLOv3 achieves a higher mean average precision (mAP) score and makes predictions at three different scales. An accurate system with a reasonable level of real-time speed is preferred over a high fps system with a low accuracy. The ability of YOLOv3 to make predictions at varying scales preserves fine features which enables the network to detect small objects. YOLO v3 uses the Darknet-53 network for feature extractor which has 53 convolutional layers. It is much deeper than the YOL v2 and also had shortcut connections

Step 4: Character Segmentation

We used a function named *regionprops()* to segment the characters. It measures a set of properties for each labelled region in the label matrix. To measure the attributes of the image region, we employ a bounding box. After labelling the connecting components, the region will be extracted from the input image. The region props function returns the centroids in a structure array.

```
S=regionprops(BW,centroid)
```

And store the x- and y-coordinates of the centroids into a two-column matrix.

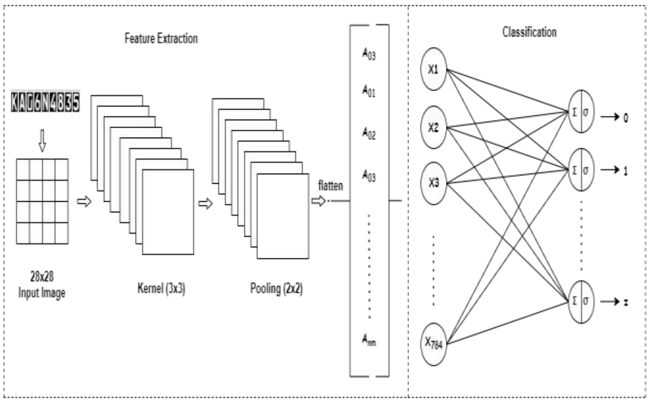
Step 3: Character Recognition

This step is responsible for recognizing characters in the extracted license plate from the previous step. In this phase we just need to predict each character using the model. For this, we'll first fix the dimension of each character image using the function fix_dimension, in which it converts an image to a 3-channel image. The image can then be sent to model.predict_classes() in order to get the predicted character.

Step 6: Storing Results to Database

The resulting data is saved in a database, which contains details such as the vehicle's licence plate, the time and date the data was collected, and the frequency with which it was collected.

3.3 Neural Network



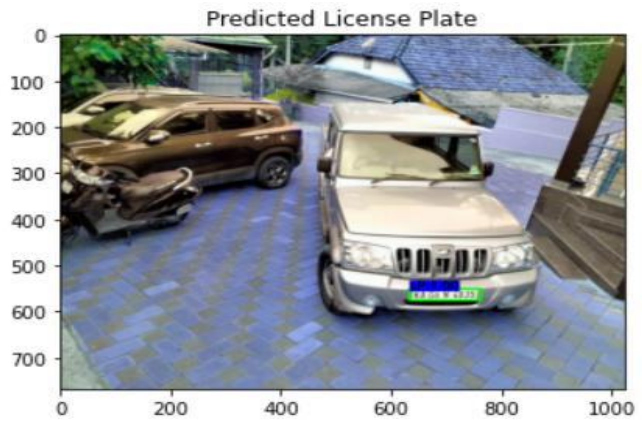
4. RESULTS AND DISCUSSION

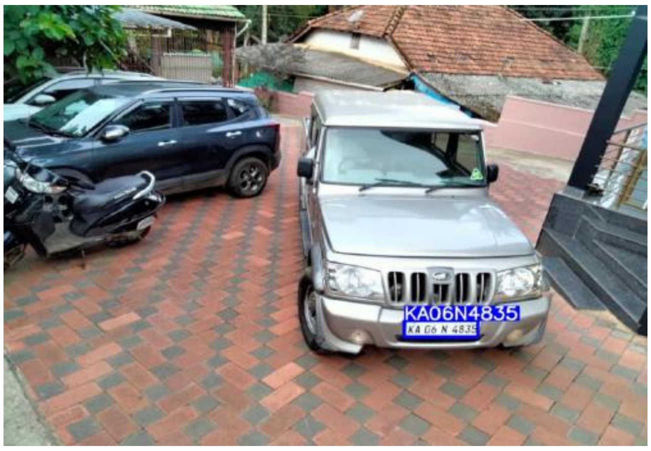
4.1 Captured Image

The image is extracted from the live camera and sent for further processing.

4.2 Plate Detection

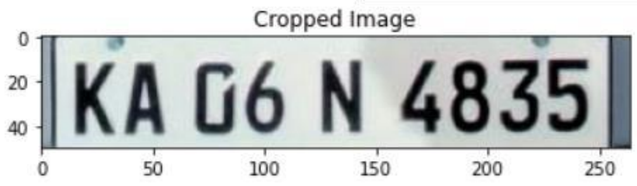
In this phase, it creates a 4D blob from the image, sets input to the network, runs the forward pass to get output of the output layers and sends the image to the function postprocess() for removing low confidence boxes and drawing the predicted box.

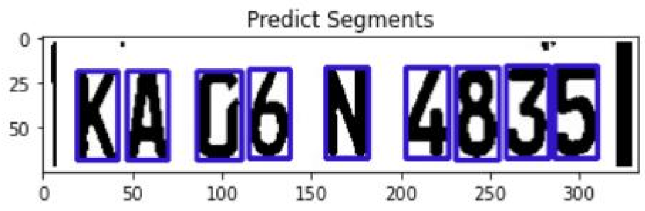




4.3 Character Segmentation

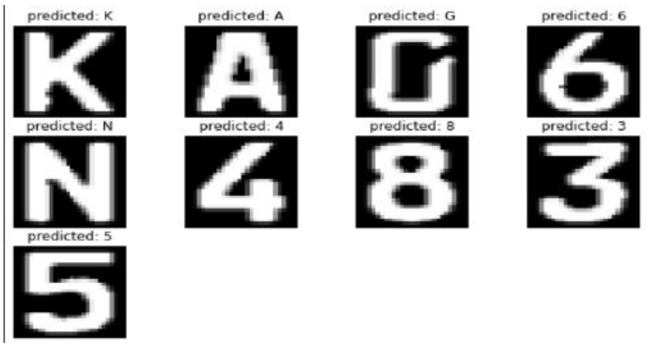
Character segmentation is an operation that seeks to decompose an image of a sequence of characters into sub images of individual symbols. It is one of the decision processes in a system for optical character recognition (OCR).



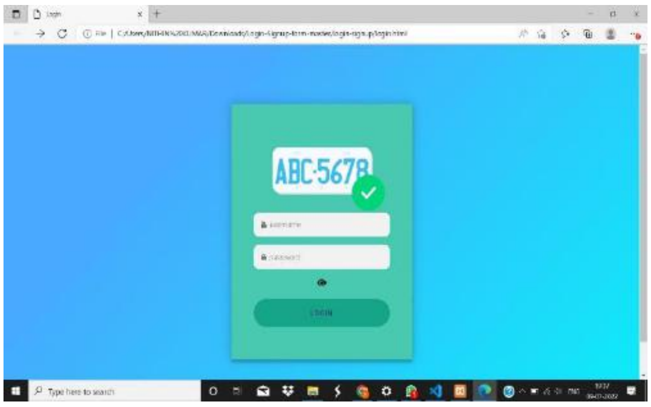


4.4 Character Recognition

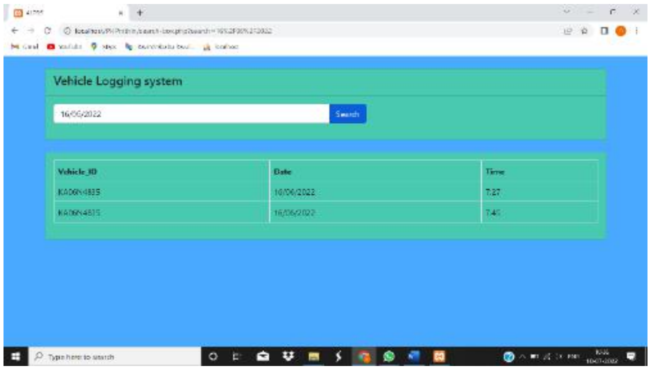
In this phase we just need to predict each character using the model. For this, we'll first fix the dimension of each character image using the function fix_dimension, in which it converts an image to a 3-channel image. The image can then be sent to model.predict_classes() in order to get the predicted character.



4.5 Admin Login Page



4.6 Dashboard


Vehicle Logging system

16/06/2022 Search

Vehicle_ID	Date	Time
KA06N4835	16/06/2022	7:27
KA06N4835	16/06/2022	7:45

5. CONCLUSION AND FUTURE WORK

The developed model is capable of recognising various kinds of licence plates across India. The use of Jetson Nano and IMX219 cameras will enable implementing this model at an organization's front gate easier, while also improving security and lowering security effort. With the help of a developed model, anomalous activities within the organisation can be easily traced. The results show that our modal achieves an average accuracy rate of 98.34%.The proposed modal and the current license plate recognitions are designed to work on license plates from specific countries and use country-specific information which limits practical applicability. Such license plate recognition systems require changes in the algorithm to work on other countries license plates. Design and develop a modal which works on multinational license plates recognition by using datasets from various countries that share the same license plate layout. And improving the dashboard by adding more filtering and searching options to logging details, to make the modal more user-friendly.

REFERENCES

[1]. Li, H., & Shen, C. (2016). Reading car license plates using deep convolutional neural networks and LSTMs. *arXiv preprint arXiv:1601.05610*.

[2]. Hsu, G. S., Ambikapathi, A., Chung, S. L., & Su, C. P. (2017, August). Robust license plate detection in the wild. In *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)* (pp. 1-6). IEEE.

[3]. Kurpiel, F. D., Minetto, R., & Nassu, B. T. (2017, September). Convolutional neural networks for license plate detection in images. In *2017 IEEE International Conference on Image Processing (ICIP)* (pp. 3395-3399). IEEE.

[4]. Bulan, O., Kozitsky, V., Ramesh, P., & Shreve, M. (2017). Segmentation-and annotation-free license plate recognition with deep localization and failure identification. *IEEE Transactions on Intelligent Transportation Systems*, *18*(9), 2351-2363.

[5]. 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.

[6]. Montazzolli, S., & Jung, C. (2017, October). Real-time brazilian license plate detection and recognition using deep convolutional neural networks. In *2017 30th SIBGRAPI conference on graphics, patterns and images (SIBGRAPI)* (pp. 55-62). IEEE.

[7]. 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.

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

[9]. Xu, Z., Yang, W., Meng, A., Lu, N., Huang, H., Ying, C., & Huang, L. (2018). Towards end-to-end license plate detection and recognition: A large dataset and baseline. In *Proceedings of the European conference on computer vision (ECCV)* (pp. 255-271).

[10]. Zhuang, J., Hou, S., Wang, Z., & Zha, Z. J. (2018). Towards human-level license plate recognition. In *Proceedings of the European Conference on Computer Vision (ECCV)* (pp. 306-321).

[11]. Chen, R. C. (2019). Automatic License Plate Recognition via sliding-window darknet-YOLO deep learning. *Image and Vision Computing*, *87*, 47-56.