Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 12 Issue 07, July-2023

Performance Comparison of Real-Time Automatic Vehicle Number Plate Detection and Recognition System using Haar Cascade, Local Binary Pattern (LBP) and YOLO v5

Nikhil S M, Chippagiri Vishnu, Anuradha S*
Department of Electronic Science, Bangalore University, Bengaluru-560056

***Abstract* - Automatic Vehicle Number Plate Detection and Recognition (AVNPDR) system can be used to control and implement traffic rules effectively. A number of machine learning classifiers are employed for numerous object detection and recognition purposes. In this work, three classifiers viz., Haar cascade, Local Binary Pattern and YOLO v5 are used to detect and localize the number plate present in an image. The performance of these classifiers and their suitability for AVNPDR is studied for real-time implementation. In this work, the machine learning models are trained with number plates whose characters are present in a single line. After localization of the plate, the characters present in the number plate are segmented using image processing techniques. Finally, the characters are fed to an optical character recognition system PyTesseract, which recognises the characters and displays the recognised character as a string. The implementation of the three models in real-time implementation is carried out with the help of Raspberry Pi Model 3B and their performances are compared. Among the three classifiers, Haar classifier ranks the best on accuracy (99%), precision (100%) and F1 score (99.3%) followed by YOLO and LBP.**

Keywords: Automatic Number Plate Detection and Recognition, HAAR cascade classifier, Local Binary Pattern, YOLO v5

I. INTRODUCTION

Automatic Vehicle Number Plate Detection and Recognition system (AVNPDR) detects and isolates the number plate (or licence plate) present in an image and recognizes the characters present in the region of interest (ROI). AVNPDR has enormous possibilities in the current information driven world. As the vehicular volume is increasing year by year unprecedently, the government's role in monitoring, maintaining, and enforcing laws have become strenuous and laborious. AVNPDR system can be implemented in traffic monitoring and vehicle tracking systems to assist in enforcing the law and order. It can be used in toll gates to identify the vehicle and link to its databases for automatic toll collection. These systems can also be used for entry and denial systems in parking lot and automate the whole process involved in managing a parking lot [1]. These real time applications demand the system to be accurate and automatic in nature.

However, the task of detecting the number plate and thereon its license number in real time is challenging for reasons such as i) the obtained image may be skewed when the vehicle is in motion, ii) images may be noisy either due to poor lighting or weather conditions, camera fault etc., iii) non-uniformity in the placement and format of the number plates for different vehicles. This demands a detection and recognition model that is robust to these variations, can accurately localize the number plate and as well as read the characters present in the licence plate.

In this work, we intend to build and implement AVNPDR system using machine learning techniques. For the detection of number plate, the recognition models studied are Haar cascade classifier, Local Binary Pattern (LBP) and YOLO v5 which are implemented separately and their performances are compared.

The AVNPDR system is built on three sub systems namely

1. *The number plate detection system*: The system is used to detect and localize the number plate in an image. The three classifier models mentioned before are considered separately for implementing AVNPDR.
2. *The character segmentation system*: The image detected with number plate, in real-time, is subjected to preprocessing to segment out the characters present in the number plate. Several image processing techniques are implemented such as binarization, building contours, and filtering.
3. *The character recognition system*: PyTesseract (PyTesseract 0.3.10) OCR is used to recognize the segmented characters.

All the three classifier models are used for implementing real time AVNPDR system using the Raspberry Pi Model 3B. The performance based on the accuracy, the time taken for training, and the computational cost are compared.

The literature survey of related work is presented in Section II. A brief review of the classifiers used and the implementation of AVNPDR is presented in Section III. The details of training the classifiers are presented in Section IV. Section V presents the results and discussions and finally the conclusions are presented in Section 6.

II. RELATED WORK

Owing to the importance and requirement of accurate number plate detection systems, many studies have been published

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 12 Issue 07, July-2023

related to this domain. The literature review reveals that the number plate detection system has been initially studied by implementing only the morphological image processing algorithms for edge detection and character recognition [2, 3] and have moved on to making use of the machine learning techniques such as Convolutional Neural Network (CNN) and Neural Networks [3]. A detailed survey of the algorithms and processes used for number plate detection and character recognition is available in [4]. To highlight the kind of work that has been carried out previously in this area, few papers have been reviewed which is relevant to this work.

An automated number plate localization model is studied in [5] which is based on the modified Grab cut algorithm. The model is applied to licence plates of different countries and have reported 99.8% accuracy (500 images) for plate localization. The study is however limited to plate localization and the model does not include character recognition.

In [6], the number plate recognition system is based on image processing techniques such as morphological transformation, Gaussian smoothing, Gaussian thresholding and contouring. The character recognition is achieved using the K-nearest neighbour algorithm. The accuracy stated by the authors are 98.02% for plate localization (101 images) and 96.22% for character recognition (768 characters). However, the algorithm is not studied for an automated system.

In [7], deep learning technique for number plate recognition is employed using Faster- RCNN and Inception V2 model. Tesseract OCR model is used for character recognition in the detected number plate. The accuracy of the number plate detection model is reported to be 98.6%. In [8], the authors have developed a character recognition model based on template matching. The model is implemented on localized number plates obtained from static images and an average accuracy of 80.8% is reported.

In [9], the authors have implemented detection of vehicle number plate using a pre trained model called YOLO v3 and for character recognition using deep learning model based on CNN and have achieved an accuracy of 98%.

A notable deficit in the various works that have been published till now is that some have studied only the localization of the number plate without considering the character recognition, while some of them have considered both number plate recognition as well as character recognition in their work. However, real time implementation has not been considered. In this study, we propose to study and implement some unexplored algorithms in AVNPDR for real-time implementation.

In the context of face recognition, the two classifiers that have shown promising results are the Haar classifier and the Local Binary Pattern (LBP).

In [10], a robust method for detecting human faces in images and video in real-time is proposed which is claimed to be high on detection accuracy with minimum computation time. The method uses integral images and extracts the Haar features, for face recognition followed by a cascade of boosted classifiers, which is a type of machine learning model. The cascade of classifiers is trained to reject easy negatives early in the cascade and pass on difficult negatives for further consideration. This helps to achieve high detection rates while keeping false positives low. Since the method seems promising on detecting faces, a similar approach based on Haar features is implemented in this work for the detection of vehicle number plates.

In [15], the authors present a method for face recognition using Local Binary Patterns (LBP). They have proposed a feature extraction technique that captures local texture information from facial images and are used to train the classifier for face recognition. The LBP method is reported to be computationally efficient and robust to illumination variations with an accuracy of 95% in face recognition. The method is also reported to achieve high recognition rates on several datasets, demonstrating its effectiveness for face recognition applications.

In conclusion, the literature review has revealed various approaches and techniques used for vehicle number plate detection and recognition systems. While some studies have focused on the localization of number plates, others have integrated character recognition to achieve accurate identification. However, there is still a need for real-time implementation of these systems to make a complete performance comparison. The Haar and LBP classifiers that have shown promising results in face recognition, and the YOLO classifier will be implemented here for number plate detection. Additionally, PyTesseract is used for character recognition. This study aims to contribute to the advancement of AVNPDR systems by providing a performance comparison of different algorithms and techniques for real-time implementation.

III. REVIEW OF HAAR CASCADE CLASSIFIER, LOCAL BINARY PATTERN (LBP) AND YOLO V5

The classifiers used in this work for localization of the number plate are Haar cascade classifier, Local Binary Patterns (LBP) and You Only Look Once (YOLO) v5. This section provides a brief review of these classifiers.

A. Haar Cascade Classifier

The Haar cascade classifiers are effective for object detection. This method was proposed by Paul Viola and Michael Jones [14]. Haar Cascade is a machine learning-based approach where a lot of positive and negative images are used to train the classifier. Cascading is a particular case of ensemble learning based on the concatenation of several classifiers; it is a multi-stage system. Cascading classifier are trained with several positive and negative images. Positive images are images that contain objects which is to be detected by the classifier and negative images constitute images which do not contain the object to be detected. The classifier is pre-trained before it can be applied to a region of an image and detect the object in question.

The Haar Cascade classifier is implemented in three stages

i) Calculating Haar Features:

The Haar features are determined by calculating the difference between the sum of intensity in the dark region and the lighter region. The size of the Haar feature should be more than 1 X 1 pixels. Fig 1 shows different types of Haar features.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 12 Issue 07, July-2023

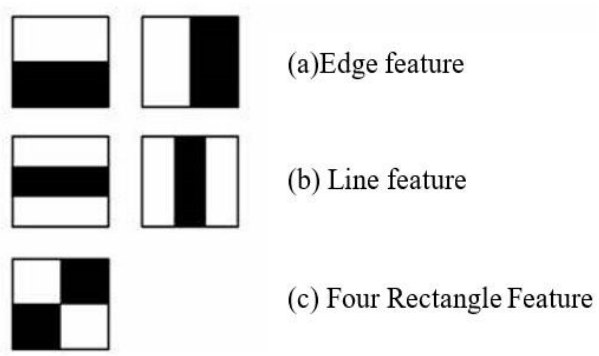


Fig 1. Haar Features

ii) Creating Integral Images:

Integral images essentially speed up the calculation by decreasing the number of computations by summing the pixel intensity in darker and lighter regions. This calculation is performed by first creating a new image of the same size as the original image, with each pixel initialized to zero. Then, for each pixel in the new image, the algorithm adds up all the pixel values in the rectangle defined by the original image from the origin. Fig 3 shows the integral image generated using original image shown in Fig 2 These are then used to compute the Haar features.

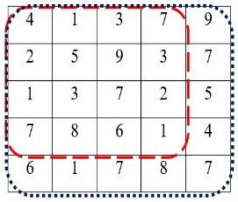


Fig 2. Original Image

4	5	8	15	24
6	12	24	34	50
7	16	35	47	68
14	31	56	69	94
20	38	70	91	123

Fig 3. Integral Image

iii) Ada Boost Training:

To determine the best features that represent an object from the thousands of Haar features, Adaboost training is used. It combines a group of weak classifiers to give out strong classifier that an algorithm can use to detect objects [14].

B. Local Binary Pattern (LBP)

LBP classifier, is an object detection algorithm that identifies objects in an image. The striking property of the LBP operator is its robustness to monotonic grey-scale changes caused by illumination variations. Another aspect of LBP is its computational simplicity, which makes it possible to analyse images in real-time [15].

LBP is basically an efficient texture operator which labels the pixels of an image by thresholding the neighbourhood of each pixel and considers the result as a binary number. To generate the LBP texture descriptor, the image is first converted to grayscale. The LBP value is then calculated for the center pixel and stored in the output 2D array. The center pixel (highlighted in red in Fig 4) is the threshold value to be compared with its neighborhood of 8 pixels.

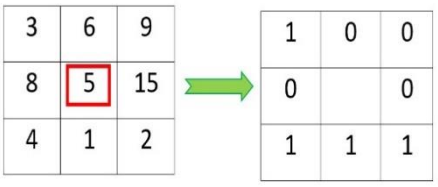


Fig 4. Local Binary Pattern Values

If the intensity of the center pixel is greater-than-or-equal to the neighboring pixel, then the corresponding neighboring pixel location value in LBP pattern is set to 1 otherwise, set to 0. Fixing one of the 8 neighbors as the starting bit, and traversing through the rest of the seven bits in clockwise or counter-clockwise direction, a total of 28 = 256 possible combinations of LBP codes can be generated. The results of this binary test are stored in an 8-bit array which is then converted to decimal. An example of generating LBP code is shown in Fig 5.

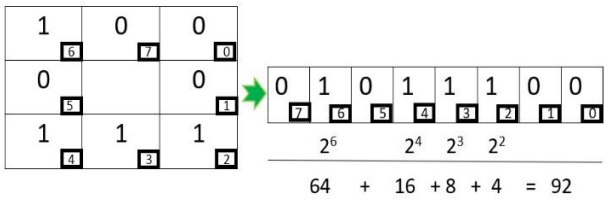


Fig 5. Local Binary Pattern Codes

This value is stored in the output LBP 2D array. The process of thresholding, accumulating binary strings, and storing the output decimal value in the LBP array is repeated for each pixel in the input image. Finally, a 256-bin histogram of LBP codes forms the final feature vector.

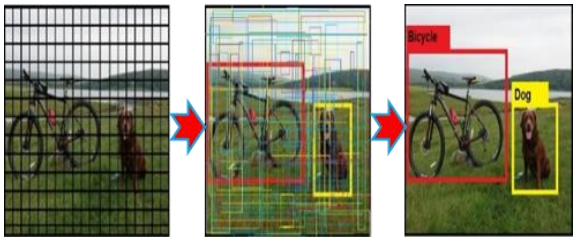
C. You Only Look Once (YOLO) v5

The YOLO v5 algorithm requires only a single forward propagation through a CNN network to detect objects [16]. The prediction in the entire image is done in a single algorithm run. Glenn Jocher introduced YOLO v5 using the Pytorch framework [16]. Object detection in YOLO v5 is a regression problem and provides the class probabilities of the detected images and predict bounding boxes simultaneously.

YOLO algorithm works using the following three stages:

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 12 Issue 07, July-2023

1)Residual blocks: The image is divided into various grids. Each grid has a dimension of S x S. Fig 6 (a) shows an input image divided into equal dimension grid cells. Every grid cell detects objects that appear within them. An object that appears within a certain grid cell, is detected and the object is indicated by green colour grid in the grid cell.



(a) (b) (c)

Fig 6. Grid Cells

2)Bounding box regression: A bounding box is an outline that highlights an object in an image. Every bounding box in the image consists of the following attributes: Width (bw), Height(bh), Class (for example, number plate, dog, bicycle.) represented by the letter c and bounding box centre (bx, by). Fig 6 (b) shows an example of a bounding box. The bounding box has been highlighted. YOLO uses a single bounding box regression to predict the height, width, centre, and class of objects.

3)Intersection over Union: IoU is a phenomenon in object detection that describes how boxes overlap. YOLO v5 uses IoU to provide an output box that surrounds the objects perfectly. Each grid cell is responsible for predicting the bounding boxes and their confidence scores. Fig 6 (c) shows detected bounding boxes. The IoU is equal to 1 if the predicted bounding box is the same as the real box. This mechanism eliminates bounding boxes that are not equal to the real box. Fig 7. provides a simple example of how IoU works. There are two bounding boxes, one in green and the other one in red. The red box is the predicted box while the green box is the real box. YOLO v5 ensures that the two bounding boxes are equal (IoU=1).

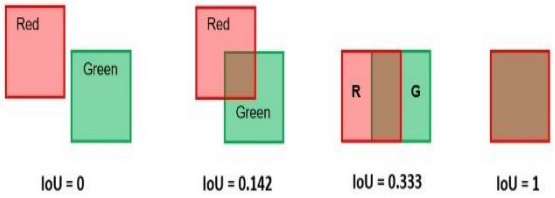


Fig 7. IoU Values

IV. Implementation of AVNPDR

The proposed Automatic Vehicle Number Plate Detection and Recognition system has several sub systems as shown in Fig 8.

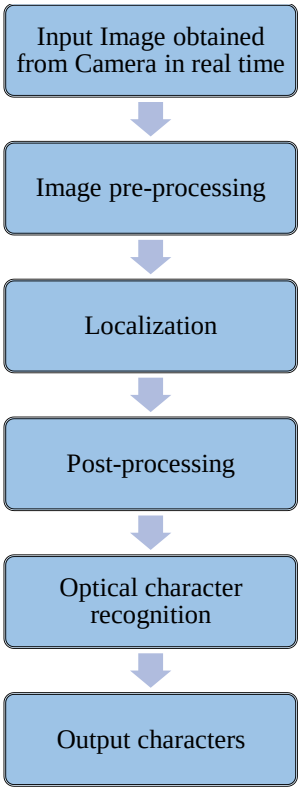


Fig 8. Implementation of the proposed AVNPDR model

Input Image: The input image is the image of the automobile acquired in real-time whose number plate has to be detected and the characters in the number plate have to be recognised. In real time, each frame of the video is taken as the input image, for example, a car image is shown in Fig 9, which is obtained by interfacing Logitech c270 720p camera to Raspberry Pi Model 3B.

Pre-Processing: This process involves basic image processing techniques and manipulations which are performed on input RGB images. The obtained image is converted to its grey scale image shown in Fig 10. This is done to increase the computational throughput of the system.

Localization: In this step the number plate present (if any) in the image is detected and the region of interest (ROI) is determined by the pre-trained classifier, which localizes only the number plate from the entire image, thereby removing all background noise. The localized number plate is enclosed in bounding box as shown in Fig 11.

Post-Processing: After localization of the plate, the obtained number plate is resized to maintain uniformity and to assist in segmentation process. Post processing involves, image processing techniques such as binarization which is done to segment the image into foreground and background, thereby making it much simpler to extract and recognize characters in the number plate. Fig 12. (a) shows the binarized image. On the binarized image, the contours are built. Contours of a certain size are selected and others rejected. For the selected contours bounding boxes are drawn as shown in Fig 12. (b). Fig 12. (c) shows a segmented character after eliminating noise using median filter.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 12 Issue 07, July-2023

Fig 9. Input Image

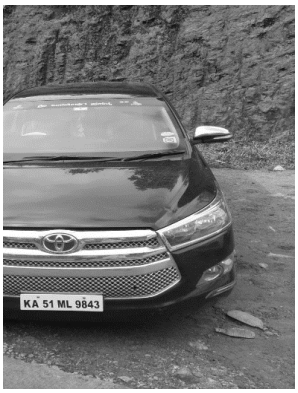


Fig 10. Grey-Scale Image

Fig 11. Localized Number Plate (using Haar Classifier)

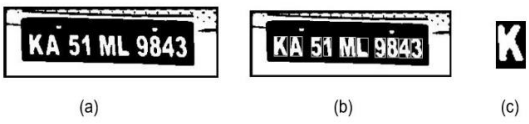


Fig 12. Post-Processing (a) Binarized image (b) Segmented Characters (c) Segmented character with median blur

Optical Character Recognition: The optical character recognition system PyTesseract (PyTesseract 0.3.10) OCR is used in this work to recognise and display the characters as a string.

Output Text: The characters present in the number plate are available as the output of the OCR in the text format.

B. Dataset and training the classifiers

For training the classifiers, we require dataset which have the essential positive images and negative images. Haar cascade classifier and Local Binary Pattern require such datasets. For YOLO v5 an annotated dataset is required, which has the image as well as the corresponding bounding box co-ordinates. Table 1 gives the details of the datasets used in this work for training the classifiers.

TABLE 1: DATASET FOR TRAINING

Classifier	Dataset
Haar & LBP	Positive images – 2800 images from [17] Negative images – 5000 images obtained from [18]
YOLO v5	Annotated images—1500 obtained from [19]

The 2800 positive images used for training the Haar and LBP contain images of number plates, which includes images that are blurred, skewed images and of different sizes. The negative images make up 5000 images of cars which strictly do not contain number plates and are handpicked from [18]. As intended, we compared all three classifiers on different parameters such as accuracy, time taken for training the classifiers, and also the memory requirement of each classifier.

The system configuration is AMD Ryzen 5 4600H Processor, installed RAM 8.00 GB DDR4, 64-bit operating system and Nvidia GEFORCE GTX 1650 Ti. The time taken for training Haar Classifier is, 3 days 2 hours 50 minutes 6 seconds, for LBP it is 4 hours 47 minutes 37 seconds and for YOLO v5 it is 3 hours 12 minutes 23 seconds. The time taken for training of the classifiers is tabulated in Table 2. The Memory Size of the trained weights are 147 Kilo bytes for Haar Classifier, 61 Kilo bytes for LBP and 27.3 Megabytes for YOLO v5.

TABLE 2: TIME TAKEN FOR TRAINING OF THE CLASSIFIERS

Classifier	Time taken for training
Haar	3days 2Hrs 50min 6s
LBP	4Hrs 47min 37s
YOLO v5	3Hrs 12min 23s

V. RESULTS AND DISCUSSION

All the classifiers were implemented in real time on Raspberry Pi. For testing the model, 100 real time images are taken

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 12 Issue 07, July-2023

of which 85 of them are images that contain number plate and other 15 are random images that do not include number plate.

A confusion matrix is used to evaluate the performance of a machine learning algorithm in classification tasks. It summarizes the number of positive and negative predictions of the algorithm on a set of test data. The inference from confusion matrix parameters relevant to the work is presented in table 3.

TABLE 3: CONFUSION MATRIX PARAMETERS FOR AVNPDR

True Negative [TN]	The image has NO number plate and model did NOT predict the number plate
False Positive [FP]	The image has NO number plate and model predicted (YES) the number plate
False Negative [FN]	The image has (YES) number plate and model did NOT predict the number plate
True Positive [TP]	The image has (YES) number plate and model did (YES) predicted the number plate

The confusion matrix of all three classifiers is shown in the Fig 13.

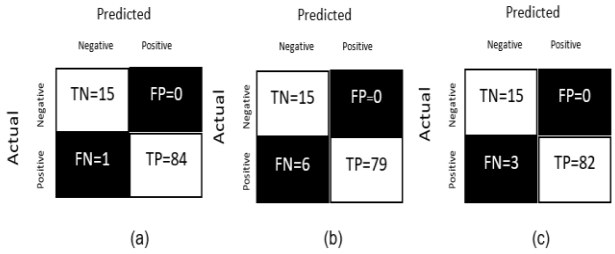


Fig 13. confusion matrix (a) Haar classifier (b) LBP classifier (c) YOLO v5 classifier

The performance parameter scores of the confusion matrix which include accuracy, precision, recall and F1-score for the three classifiers for number plate detection are presented in Table 4. The accuracy achieved by using Haar classifier is 99%, 94% for LBP and 97% for YOLO v5.

TABLE 4: COMPARISON OF CLASSIFIER PARAMETERS

Parameters	**Classifiers**		
	Haar	**LBP**	**YOLO v5**
Accuracy $\left[\frac{TP+TN}{TP+TN+FP+FN}\right]$	99%	94%	97%
Precision $\left[\frac{TP}{TP+FP}\right]$	100%	100%	100%
Recall $\left[\frac{TP}{TP+FN}\right]$	98.8%	92.9%	96.4%
F1-Score $\left[\frac{2*precision*Recall}{Precision+Recall}\right]$	99.3%	96.3%	98.1%

VI. CONCLUSION

The study involves performance comparison of the three classifiers namely, Haar, LBP and YOLO v5, for real time implementation of automatic number plate detection and recognition. An accuracy of 99% was achieved using Haar classifier. However, LBP provided an accuracy of 94% and of YOLO is 97%. Also, LBP and YOLO v5 techniques either fall short of accuracy or memory size, hence making Haar Classifier an ideal choice for the current scenario. Post processing steps include, binarization, segmentation and median filtering that are carried out on the localized number plate. Post processing of the localized number plates assist the OCR in recognising the characters present in the number plate. It was observed that PyTesseract provided an accuracy of 91.6% in recognizing the segmented characters. The accuracy of the OCR also contributes to the overall performance of the AVNPDR system.

This work has focussed on detecting the vehicle number plates with characters present in a single line. The work can further be extended to detect number plates with two or more lines.

REFERENCES

[1] Artificial intelligence cameras at 50 Bengaluru junctions to spot traffic offences, The Times Of India, https://timesofindia.indiatimes.com/city/bengaluru/artificial-intelligence-cameras-at-50-bengaluru-junctions-to-spot-traffic-offences/articleshow/93015182.cms,July 21, 2022.

[2] Razvan-Daniel Vitoiu, Marius Brezovan, and Adrian Graur, "Automatic Number Plate Recognition System," Annals of the University of Craiova, Mathematics and Computer Science Series, Vol. 38(1), pp. 62-71, 2011. ISSN: 1223-6934.

[3] P. Kulkarni, A. Khatri, P. Banga and K. Shah, "Automatic Number Plate Recognition (ANPR) system for Indian conditions," 2009 19th International Conference Radioelektronika, Bratislava, Slovakia, 2009, pp. 111-114, doi: 10.1109/RADIOELEK.2009.5158763.

[4] Lubna, Mufti, N., & Shah, S. A. A. (2021). Automatic Number Plate Recognition: A Detailed Survey of Relevant Algorithms. Sensors, 21(9), 3028. https://doi.org/10.3390/s21093028

[5] Ayodeji Olalekan Salau, Thomas Kokumo Yesufuand Babatunde Sunday Ogundare," Vehicle plate number localization using a modified Grab-Cut algorithm," 23 January 2019 Journal of King Saud University.

[6] Ravi Kiran Varma Pa, Srikanth Gantaa, Hari Krishna Bb, Praveen SVSRK," Novel Method for Indian Vehicle Registration Number Plate Detection and Recognition using Image Processing Techniques," International Conference on Computational Intelligence and Data Science (ICCIDS 2019)

[7] T. Vetriselvi, E. Laxmi, Sachi Nandan ,4, Eatedal, Shaha, Amal and Romany," Deep Learning Based License Plate Number Recognition for Smart Cities," Computers, Materials & Continua Tech Science, 2022, vol.70, no.1

[8] Aniruddh Puranic, Deepak K. T, Umadevi V," Vehicle Number Plate Recognition System: A Literature Review and Implementation using Template Matching," International Journal of Computer Applications, Volume 134 – No.1, January 2016

[9] B Kokila, Sudarshan Pattabiraman, M Praveena and H Thabass, "Smart System for Indian License Plate Detection and Recognition Using Deep

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 12 Issue 07, July-2023

Learning Techniques," International Journal of Science, Engineering and Technology,2021, 9:2.

[10] Paul Viola and Michael J Jones, "Robust Real-Time Face Detection," International Journal of Computer Vision 57(2), 137-154,2004

[11] Naveen M,Arasan J,S Shivakumar and Adithya U, "Vehicle Number Plate Detection Using Image Processing and OpenCV-Python," International Journal of Innovative Research in Science, Engineering and Technology, Volume 9, Issue 7, July 2020.

[12] B.Santosh manoj kumar, M.V.K.Prasad and K.Sripath Roy, "University Campus Number Plate Logging System," International Journal of Innovative Technology and Exploring Engineering, Volume-8 Issue-7, May, 2019.

[13] Souhail Guennouni, Ali Ahaitouf and Anass Mansouri, "A Comparative Study of Multiple Object Detection Using Haar-Like Features Selection and Local Binary Patterns in Several Platforms," Hindawi Publishing Corporation Modelling and Simulation in Engineering Volume 2015, Article ID 948960

[14] P. Viola and M. Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features," in Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2001, Kauai, HI, USA, Dec. 2001, pp. I-511-I-518. doi: 10.1109/CVPR.2001.990517.

[15] Ahonen, T., Hadid, A., Pietikäinen, M. (2004). Face Recognition with Local Binary Patterns. In: Pajdla, T., Matas, J. (eds) Computer Vision - ECCV 2004. ECCV 2004. Lecture Notes in Computer Science, vol 3021. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-24670-1_36

[16] Glenn Jocher, YOLOv5 Documentation, https://docs.ultralytics.com/

[17] Thamizhselvan, https://www.kaggle.com/dtasets-/thamizhsterio/indian-license-plates

[18] Paul, 60,000+ Images of Cars, Kaggle https://www.kaggle.com/datasets/prondeau/the-car-connection-picture-dataset

[19] Saworz, Real-time Plate Reading (video) | YOLOv5 & easy OCR https://www.kaggle.com/code/saworz /real time -plate-reading-video-yolov5-easyocr