Efficient Entry Management of Vehicles Through KNN-based Character Recognition of Number Plates

Urjani Chakravarti
School of Electronics Engineering
Vellore Institute of Technology, Vellore
Vellore, India
uchakravarti@hotmail.com

Aryan Bansal
School of Electronics Engineering
Vellore Institute of Technology, Vellore
Vellore, India
aryanbansal1710@gmail.com

Vikram Baruah
School of Electronics Engineering
Vellore Institute of Technology, Vellore
Vellore, India
vikram.baruah@gmail.com

J.Valarmathi
School of Electronics Engineering
Vellore Institute of Technology, Vellore
Vellore, India
jvalarmathi@vit.ac.in

Naumi Krishna K. Panicker
School of Electronics Engineering
Vellore Institute of Technology, Vellore
Vellore, India
naumikrishna@gmail.com

2023 Innovations in Power and Advanced Computing Technologies (i-PACT) | 979-8-3503-2518-8/23/$31.00 ©2023 IEEE | DOI: 10.1109/I-PACT58649.2023.10434895

***Abstract*— This study focuses on character recognition techniques for vehicle number plate recognition. It employs the K Nearest Neighbor (KNN) algorithm to detect the characters in the vehicle's number plate. In addition, it successfully identifies the state of origin of vehicles from the vehicle number thereby easing documentation of vehicles. Moreover, the algorithm marks the time of entry of vehicles in a systematic manner. The algorithm provides a methodological solution to keep track of the numerous vehicles entering a premise in an orderly fashion.**

Keywords*— *Character Recognition, Vehicle Number Plate Recognition, Machine Learning, KNN

I. INTRODUCTION

The field of software development has simplified mundane tasks that require manpower and effort. In the recent decade, the demand for functional and optimal technology has been on the rise to simplify administrative and directorial work. One such technology is optical character recognition (OCR) that enables software to recognize fonts and numbers. OCR is a technology that enables the extraction and identification of any character from a document or image and its utilization in a form that is meaningful. It is a program that can be used to recognize characters that exist in a certain image. The field of OCR has several applications. One of the most popular applications can be seen in the form of Google Lens, a popular software developed by the tech-giant Google. Another such application is searching for a particular word or character in a large document. Clearly, OCR has evolved through the ages and has a lot of scope in modern technology. The main challenge lies in examining the information obtained through OCR and using this information in a responsible and innovative manner. Apart from this, the challenge may arise due to the different types of fonts and the plethora of languages known to humankind.

The problem of character recognition is one of the most common computer vision problems. Despite the large number of various character recognition applications available, the importance of developing new software has not diminished. This is primarily because most free software programs have low accuracy and are designed to recognize printed text and the image processing operations required to detect the characters are limited. Based on the principles of OCR, it was decided to develop a system that uses its own character recognition algorithms, built on existing ones to recognize number plates of vehicles as an application.

The number plate recognition system is based on image processing technology. The primary objective is to develop an autonomous system that can recognize a number plate successfully so that the number of fatalities caused by reckless drivers and wrong doers can be reduced and the chargeable can be detected. In addition to this, the system has added functionalities of identifying and storing the state of origin or the vehicle. The algorithm can also keep a record of the time of entry of each vehicle allowing for a systematic and organized database. This allows one to automate the task of keeping track of the vehicles that enter a premise. It creates a database containing important information like the state of origin, time of entry and the vehicle number. The system first processes the captured image using image processing techniques to reduce distortion and improve the resolution. Post this, the exact location of the number plate is recognized, using the segmentation technique, after which the length of number plate is determined and correlated. Finally, the recognized number plate is printed as an output string. This string can further be processed to get the state of origin by matching the characters

to a dataset of all the states in India. The system is implemented and simulated on Python 3 and performance is tested on real images. This type of system can be used in densely populated areas, toll booths, parking areas, and on highways. This paper proposes a method to identify vehicles coming from COVID hotspots. A buzzer or gate control system can be integrated to prevent people coming from COVID hotspots from entering the premises. It can also serve as a measure to restrict entry of unauthorized vehicles.

There are various approaches to develop an algorithm for OCR. Once such approach is the KNN algorithm which has been used in this study. This study also requires an additional Python library called Open-Source Computer Vision Library (Open CV) for various objectives like image processing and implementing the KNN algorithm.

The key motivation behind this study was to implement vehicle number plate recognition which enables users to monitor the state of origin of the vehicles. Due to the COVID-19 pandemic, there has been a sudden rise in pandemic hotspots. This makes social distancing a primary solution in a populated country. As a result, one can easily identify the vehicles which have come from a COVID hotspot and restrict the entry of these vehicles into the premises.

II. Literature Review

One of the earliest studies conducted by J Mantas [1] in 1986, explores various algorithms and methodologies of character recognition. The study presents an overview of character recognition methodologies that have evolved in this century. For example, one of the earliest technologies of character recognition is that of scanning technologies. These mainly revolve around on-line and handwritten character recognition based on a sample. Using contour and edge detection, characters can be recognized and deciphered. This study helped to clear the basics and provide a holistic understanding of the fundamentals.

The paper published by Pustokhina, et al [2], in 2020 elaborates the methods to optimize the KNN approach using a deep learning algorithm. This paper aims to successfully extract the vehicle's number using the said algorithm. It focuses on the different techniques of extraction used, based on the application of neural networks, KNN, and deep learning. This study was the main motivation to implement the KNN algorithm due to its simple, yet effective method of recognizing characters.

De Campos, et al [3], published a paper in 2009, on camera-based character recognition. The paper highlighted the obstacle of identifying characters accurately in images of natural scenes. For example, the angle and lighting in pictures of vehicles on a road can be inferior in quality. This study focused on these anomalies and tried to improve the accuracy using various OCR techniques. For their research, they utilized an annotated database of images containing English and Kannada characters. The database comprised of images of street scenes taken in Bangalore, India using a standard camera. This study gave a clear descriptive analysis of dealing with images that were taken from unfavorable angles, allowing for a more accurate algorithm.

In a study done by Mithe, et al [4], in 2013, a new system for OCR was proposed by the group. They scrutinized the existing method of using a scanner and a computer to identify and recognize characters and came up with the solution of using an android phone with a superior camera instead, which results in saving space, with a tradeoff of slower computational speed. Their system used Tesseract, which is an open-source OCR engine. Their proposed system comprised of the following steps to correctly recognize the text: scanning, segmentation, preprocessing, feature extraction and finally, recognition. Scanning was the process of taking an image of the text that needed to be recognized and the subsequent grey-scaling required for further processing. Segmentation was the process of locating regions of printed or handwritten text used to isolate words or characters. Preprocessing was used to remove any noise or distortion which might alter the results. Feature extraction was used to extract the features of the symbol, and in the end, the Tesseract algorithm would recognize the characters.

Another study conducted by ND Cilia, et al [5], in 2017 stated the various techniques for character recognition. Here, it was mentioned that feature selection was a crucial step in the process. It reduced the computational cost of the classification track. This paper also focused on handwriting recognition and the various set of challenges in this feat. This clarified the methodology behind feature selection, which is a crucial step in OCR.

A similar study was conducted in Japan by T Clanuwat, et al [6], in 2019. This study aimed to recognize characters native to the Japanese language. The study explored Character Recognition using Deep Learning. This study has a similar approach to the study conducted by Pustokhina [2] with a variation in the algorithm which not only could recognize the Japanese language but could also predict the location and identify all the characters given in a page. This study explored the usage of OCR to recognize different fonts and languages, enabling one to explore the various uses of the OCR technology.

In 2021, Goyal et al [7] published a paper which delved into the application of character recognition systems, which is vehicle license plate recognition. Goyal et al argued that vehicle license plate recognition has become essential now, but the proper systems have not yet been designed, which can accurately recognize the character and number on license plates. In this paper, the researchers propose a procedure for successfully recognizing the license plates of cars. They analyzed different methods and found that template matching with cross correlation was the best way to proceed, as it provided an accuracy of 98.07% for Indian vehicle license plates. Convolution neural network and proficient machine learning methods were also discussed in brief. They also listed the various challenges that arose, such as low file resolution, smear images, overexposure, reflection or shadows on the number plates and dirt on the plate or something covering the characters.

A study conducted by J Liu, et al [8], in 2020 also focused on the KNN algorithm for vehicle number plate recognition. The paper discusses in detail the various methods of vehicle number plate recognition using the KNN approach. This study

implemented KNN because of its simplicity, and ability to accommodate new data without having to retrain a model.

The above review of literature explores the various methodologies for character recognition and vehicle number plate recognition. The body of literature ranges from the fundamentals of OCR to the more niche areas. Keeping the COVID-19 pandemic in mind, this study provides a conclusive approach to implement the 'new norm', to restrict the entry of vehicles from certain COVID hotspots. Apart from this, it allows for clear organization by noting the time of entry. This not only allows for documentation of visitors on premises but also allows following safe COVID practices. For example, if a certain vehicle from a COVID hotspot enters the premise at a particular time, the list of vehicles that entered the premise at the same time can be identified and made aware of the situation, which would enable them to practice self-quarantine or take any other appropriate measure. This was the key motivation of this study.

This paper is comprised of six sections. Section III provides the various methods and data sets used in this paper. Section IV explores the results and interpretation of the output. Section V deals with the performance analysis and section VI states the conclusion.

III. METHODS AND DATA SET

A. Research Questions

- To develop an algorithm that can recognize printed text in images accurately and efficiently,
- To use the character recognition algorithm developed to recognize characters on the license plates of Indian vehicles,
- To recognize and classify the vehicles according to the states in which they were registered, based on the results obtained from recognizing their number plate.
- To store the registration number, state of registration and the time of entry of the vehicle into college premises in an excel sheet.

B. Data Set

Five training images have been used to recognize alphabets from A through Z and digits from 0 through 9 for each alphabet and digit, i.e., 180 training images in total (Fig. 1). The training process produced two parallel data structures – a set of images, and a set of numbers indicating which "group" or "classification" each corresponding image is in. The classification is the alphabet or the digit of which the corresponding image belongs to. The trained model was used after completing the training process to recognize printed characters in images (Fig. 1).

An excel sheet was created with Indian state vehicle codes to compare the values retrieved from the character recognition algorithm to return the state of registration of the vehicle (Fig. 2).

Fig. 1. Character Table

	A	B	C
1	Code	Subdivision name	Subdivision category
2	AP	Andhra Pradesh	state
3	AR	Arunachal Pradesh	state
4	AS	Assam	state
5	BR	Bihar	state
6	CT	Chhattisgarh	state
7	GA	Goa	state
8	GJ	Gujarat	state
9	HR	Haryana	state
10	HP	Himachal Pradesh	state
11	JH	Jharkhand	state
12	KA	Karnataka	state
13	KL	Kerala	state
14	MP	Madhya Pradesh	state
15	MH	Maharashtra	state
16	MN	Manipur	state
17	ML	Meghalaya	state

Fig. 2. Indian State Vehicle Codes

C. Methodology

The K-nearest neighbor is one of the more popular algorithms in the field of machine learning and data sciences. In this body of research, the K-nearest neighbor or the KNN algorithm has been opted.

The model works by trying to match as many pixels as possible. Assuming each character's image has dimensions of 10 x 10 pixels, the total area of each image is 100 pixels. If the classification of a character "X", and its recognition has a best match of 99 pixels with a given training sample, that sample is considered the nearest neighbour. Each image in the training sample is checked to identify the character "X" correctly. The best match is taken as the character, and then "X" is classified.

At first, a dataset of roughly 15-20 images was taken. These images had different resolutions, angles, colours, and redundant objects in the background. The images were selected in such a way as to give vast coverage for the vehicles present on the roads. Moreover, the images were selected in a way to develop the algorithm to be efficient and accurate (Fig. 3).

Fig. 3. Original Sample Image

Next, the image was converted into a greyscale image using the OpenCV function "cvtColor" (Fig. 4). The function converts an input image from one color space to another. Greyscale images store less information per pixel compared to coloured images, which helped get rid of redundant information. The intensity of greyscale is represented as an 8-bit integer, with 256 different shades of grey ranging from black to white. Furthermore, for many activities, greyscale photographs are sufficient, so there was no need to employ more intricate and difficult-to-process colour images. The threshold image was created from the greyscale image (Fig. 5). In a threshold image, there is a threshold intensity which is the benchmark for the intensity to be considered entirely white or entirely black. All intensities above the threshold are considered to be white, while all intensities below the threshold are considered to be black. Given below is a sample for the greyscale and threshold image of the original sample image.

Fig. 4. Grayscale Image

Fig. 5. Threshold Image

The next step was contour detection to determine edges (Fig. 6). The function used for this process was OpenCV's "drawContours". The function draws contour outlines in the image if thickness >= 0 or fills the area bounded by the contours if thickness < 0. The "findContours" function retrieves contours from the binary image using the algorithm [Suzuki85]. Contour detection is essential to determine the exact location of the number plate. Since number plates are rectangular in shape, detecting contours is the perfect method of detecting the location of the number plate in the image. If there is a failure in detecting the location of the number plate, the vehicle's number cannot be obtained.

Fig. 6. Contours for the Image

In order to find the vectors, the redundant information in the contoured image needed to be processed (Fig. 7). A feature vector, or vector, is just a vector that contains information describing an object's important characteristics. It is a one-dimensional matrix which is used to describe a feature of an image. For a given contoured image, there can be more than a single vector.

Fig. 7. Vector of the Image

The next step was to find the vectors of vectors (Fig. 8). In order to successfully identify the number plate, the vectors needed to be grouped. On grouping the vectors, all the potential vector combinations were found. For a given image, there could be several vectors of vectors.

Fig. 8. Vector of Vectors

The vectors were iterated through in order to find the most probable location of the number plate. The vectors of the original image were obtained first. As observed, the algorithm was able to identify two potential vectors for the number plate (Fig. 9). The first vector was a cropped version of the original number plate, and served as a contender. The second vector was of the original number plate with the redundant information omitted and was the correct vector. The algorithm then obtained the greyscale (Fig. 10) and threshold images (Fig. 11) for these vectors.

Fig. 9. Potential Vectors of Original Images

Fig. 10. Vector after Grayscale Processing

Fig. 11. Vector of Vectors

Next, the vectors of possible characters in plates needed to be retrieved. This was done by finding the vector of vectors of matching characters. The final process was to remove the inner overlapping shapes and find the longest vector of matching characters in the plate. This was done by recognizing the characters in the plate. Fig. 12, 13, and 14 represent the step by step process.

Fig. 12. Finding Vectors of Potential Characters

Fig. 13. Longest Vector of Matching Characters in a Plate

Fig. 14. Recognized Characters

As the algorithm successfully recognized the characters, each character was compared to a list of characters. Here, KNN algorithm was used to recognize the characters. In this algorithm, different fonts for numbers through 0 – 9 and letters from A – Z (all characters that may be present on a valid number plate) were used. Assuming the algorithm had been trained to recognize numbers from 0 to 9, using 5 images for each character and each character having a width and length of 10 pixels, and hence an area of 100 sq. pixels (Fig. 15, 16), the training process generated 2 datasets – the first was a set of numbers indicating which "group" the image belonged to, and the second was a set of images. To recognize an unknown character X, X was compared to the previously mentioned images to search for the best match. If for example, image4 was the best match, X was classified as '0'.

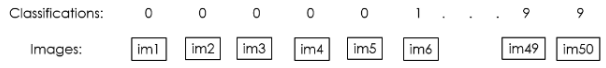

Fig. 15. Classification Table

Fig. 16. Character Table

The final output of our image processing algorithm was in two formats for our convenience. In the first format, the number was printed on the original image while simultaneously highlighting the number plate. Fig. 17 shows how the algorithm successfully identified the number plate and removed redundant information such as writings on the number plate.

Fig. 17. Output Format 1

The second format of output was in the form of printed text on Visual Studio Code (VSCode) terminal. The algorithm had been written using VSCode, and a Python 3.8.8 interpreter had been used. Fig. 18 shows the output of the program.

```
(base)
aryan@aryan-laptop MINGW64 ~/Downloads/OCR
$ python Main.py

2 possible plates found

License plate read from image: KL07E5200

----------------------------------------
['KL', 'Kerala', 'state']
[['KL07E5200', 'Kerala', 'Tue Nov 23 13:18:41 2021']]
```

Fig. 18. Output Format 2

For identifying where the vehicle was registered, an excel sheet containing a list of the two-letter codes in the first column, the corresponding states and union territories of India in the second column, and the subdivision category in the third column was used. The subdivision column was used to identify whether the place of registration is a state or a union territory. The excel sheet acted as a database in this case.

After the number plate had been correctly identified, the program accessed the excel sheet, and compared the first two characters of the recognized plate with the two-letter codes in the excel sheet's first column. Once the letters on the plate matched with a key in the database, the state's registration code, name and the subdivision category were displayed.

Next, the time of entry was stored in a systematic manner to manage the vehicles in an orderly fashion to keep track of the vehicles and allows for better security on the premises. Fig. 19 shows the file documenting all important details.

	A	B	C	D
1	Registration No.	State	Time of Entry	
2				
3	MH0AV86	Maharashtra	Mon Nov 22 19:04:43 2021	
4				
5	MH01AV8866	Maharashtra	Mon Nov 22 19:05:08 2021	
6				
7	DL4CAF4943	Delhi	Mon Nov 22 19:05:12 2021	
8				
9	KA19P8488	Karnataka	Tue Nov 23 12:29:00 2021	
10				
11	KL07E5200	Kerala	Tue Nov 23 13:18:41 2021	
12				
13	WB01AZ1042	West Bengal	Tue Nov 23 13:19:43 2021	
14				

Fig. 19. Output showing time of entry of vehicles

IV. Results and Interpretation

It is evident from the results that the methodology works well with the images that had been used. The model was able to correctly identify the number plates of the vehicles, and subsequently classify the state of registration of the vehicle.

The program can be used to identify the license plates on highways, in parking lots, and other locations as well. Its application can be augmented by creating a database containing the numbers of vehicles, which can be used to provide access to vehicles to an enclosed area only if their license plate matches a number in the database.

The main drawback of this model was that it did not produce good results if the fonts in the images were not standardized. For example, in some fonts, the digit "1" looked like the alphabet "I" (such as in the font Times New Roman). This proved to be problematic if the training images do not have a wide variety of fonts. Even if many fonts are used, the model will work best if the characters look as similar as possible in all fonts while training. Fig 20, Fig 21 and Fig 22 shows the results.

Fig. 20. Number Plate Recognition

```
(base)
aryan@aryan-laptop MINGW64 ~/Downloads/OCR
$ python Main.py

2 possible plates found

License plate read from image: KL07E5200

----------------------------------------
['KL', 'Kerala', 'state']
[['KL07E5200', 'Kerala', 'Tue Nov 23 13:18:41 2021']]
```

Fig. 21. State recognition

	A	B	C	D
1	Registration No.	State	Time of Entry	
2				
3	MH0AV86	Maharashtra	Mon Nov 22 19:04:43 2021	
4				
5	MH01AV8866	Maharashtra	Mon Nov 22 19:05:08 2021	
6				
7	DL4CAF4943	Delhi	Mon Nov 22 19:05:12 2021	
8				
9	KA19P8488	Karnataka	Tue Nov 23 12:29:00 2021	
10				
11	KL07E5200	Kerala	Tue Nov 23 13:18:41 2021	
12				
13	WB01AZ1042	West Bengal	Tue Nov 23 13:19:43 2021	
14				

Fig. 22. Time of entry of vehicles

V. Performance Analysis

The model successfully recognized the vehicle number plates by employing the KNN algorithm. A variety of images taken at different angles and having different resolutions were used. The algorithm was successful in the case of images of poor resolution and angles. The algorithm was also tested to identify a variety of fonts, however several other fonts like cursive or calligraphy cloud have been included. Moreover, the algorithm can be enhanced to include other languages and roman numericals. The algorithm had been tested over 20 images, each having unique characteristics, and only 2 images gave incorrect results, leading to an accuracy of 90%. However, these instances can be rectified by training the algorithm to include more fonts. Apart from this, the algorithm had successfully recognized the state of origin of the vehicles along with the time of entry. The algorithm can be improved by making the time complexity more optimal to allow for faster documentation which can be crucial during busy times of the day.

VI. Conclusion

The paper successfully recognized the vehicle number plates by employing the KNN algorithm. The number plates were recognized as a string of characters. The algorithm also identified the state of origin of the vehicle. This can aid in monitoring where the vehicles are coming from. It can be used to determine if the vehicle is coming from a state with several COVID-19 cases, and the vehicle can be stopped from entering the premises to prevent the spread of the virus. Apart from this, the time of entry of the vehicle was documented and stored in a systematic manner to keep track of the in-time of vehicles. The paper can be improved to include different languages and a plethora of fonts, which will help widen the reach and improve accuracy. These improvements can make the paper extremely functional, and it can be adopted by universities and offices.

References

[1] Mantas, J. (1986). An overview of character recognition methodologies. *Pattern recognition*, *19*(6), 425-430.J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[2] Pustokhina, I. V., Pustokhin, D. A., Rodrigues, J. J., Gupta, D., Khanna, A., Shankar, K., ... & Joshi, G. P. (2020). Automatic vehicle license plate recognition using optimal K-means with convolutional neural network for intelligent transportation systems. *Ieee Access*, 8, 92907-92917.

[3] De Campos, T. E., Babu, B. R., & Varma, M. (2009). Character recognition in natural images. *VISAPP (2)*, *7*.

[4] Mithe, R., Indalkar, S., & Divekar, N. (2013). Optical character recognition. *International journal of recent technology and engineering (IJRTE)*, *2*(1), 72-75.

[5] Cilia, N. D., De Stefano, C., Fontanella, F., & di Freca, A. S. (2019). A ranking-based feature selection approach for handwritten character recognition. *Pattern Recognition Letters*, *121*, 77-86.

[6] Clanuwat, T., Lamb, A., & Kitamoto, A. (2019, September). Kuronet: Pre-modern Japanese kuzushiji character recognition with deep learning. In *2019 International Conference on Document Analysis and Recognition (ICDAR)* (pp. 607-614). IEEE.

[7] Goyal, S., Dube, S., Mali, N., & Udawant, P. (2021). Vehicle License Plate Detection and Recognition System: A Review. *International Journal of Recent Advances in Multidisciplinary Topics*, *2*(10), 125-128.

[8] Liu, J., Zheng, F., van Zuylen, H. J., & Li, J. (2020). A dynamic OD prediction approach for urban networks based on automatic number plate recognition data. *Transportation Research Procedia*, *47*, 601-608.