SSRG International Journal of Electronics and Communication Engineering
ISSN: 2348-8549/ https://doi.org/10.14445/23488549/IJECE-V11I8P118

Volume 11 Issue 8, 172-184, August 2024

Original Article

Smart Insights on the Move: Deep Convolution Neural Network and Segmentation for License Plate Recognition

Shajan Jacob[1], M.K Jeyakumar[2]

[1]*Department of Computer Science and Engineering, Noorul Islam Centre for Higher Education, Tamil Nadu, India*
[2]*Department of Computer Applications, Noorul Islam Centre for Higher Education, Tamil Nadu, India*

[1]*Corresponding Author : shajanjacob24@outlook.com*

Received: 11 June 2024 Revised: 25 July 2024 Accepted: 12 August 2024 Published: 31 August 2024

Abstract - *In the modern world, the highway is becoming an increasingly significant part of the entire transportation sector. Due to the widespread interest in Intelligent Transport System technology, numerous systems are being developed and implemented globally. The Intelligent Transport System relies heavily on license plate recognition. Modern technological advancements enable the automatic identification and interpretation of license plate details from images and video streams facilitated by sophisticated license plate recognition systems. These devices are able to reliably read alphanumeric characters from license plates, even in difficult situations like changing angles, lighting, or occlusion. This paper suggests a deep convolutional neural network-based, effective system for license plate recognition. The methodology comprises multiple significant stages, commencing with the acquisition and preprocessing of images via methods like grayscale conversion and thresholding. After that, morphological processes like dilation and erosion are employed to boost the quality of the image, and segmentation is used to separate the area of interest. Next, contour extraction within this segmented area is used to estimate the character. The features that were extracted from the segmented regions are then used to drive a CNN model that accurately recognizes the characters on a license plate. The simulation results confirm that the suggested methodology is effective in correctly identifying and decoding license plate numbers from images, with an impressive recognition accuracy of 99.54%. This method provides an effective and dependable solution for automated license plate recognition tasks. It shows great promise for real-world applications in traffic management, law enforcement, and intelligent transport systems.*

Keywords - *Character segmentation, Convolutional neural network, Deep learning, Image processing, Intelligent transportation system, License plate recognition.*

1. Introduction

Intelligent Transportation Systems (ITS) aim to enhance mobility, security, safety, and the overall productivity of transportation through the application of cutting-edge technologies. As such, they have the capability to enhance the lives of individuals [1] greatly. The capacity to recognize and identify a vehicle is a fundamental need for any ITS. After a vehicle has been located and identified, further processing can be carried out, such as maintaining track of the number of vehicles, monitoring traffic, enforcing access control to various locations, sounding alarms, rerouting traffic based on traffic volumes, etc. A License Plate (LP) is a means of uniquely identifying a motor vehicle in almost every country in the world. In order to identify and recognize the vehicle LP, the ITS system can make use of an LPR system, which is created by fusing a hardware and software module [2]. Following the successful location and accurate identification of the LP, additional processing can be carried out to obtain more vehicle information for the ITS.

The dimensions, design material, colour, and character style of LPs vary by country. The front plate of a vehicle in India measures 500 x 120 mm, while the rear side plate measures 340 x 200 mm [3]. The LP of several Indian states is displayed in Figure 1.

It may be more difficult to extract information about LP in certain countries due to the background texture. Some countries use multiple fonts, sizes, colors, and characters at the same time. Figure 2 shows sample LP images from different countries.

A License Plate Recognition (LPR) system uses soft computing techniques and some image processing theory to identify vehicles based only on their LPs. The systems can be very helpful in a number of applications, encompassing LPR systems for an ITS, highway toll collection, and traffic monitoring systems [4]. LPR technology is utilized in automated parking systems, traffic monitoring, and public space security. A digital camera is used to take images of both

the front and the back of the vehicle. The image segmentation method is used to segment the LP area. The information system recognizes and stores LP. Conventional cameras are needed for some systems, while infrared and speciality illumination cameras are needed for others. In low-light conditions, a flashlight is used. In recent decades, text detection systems have been developed for security systems and document creation. Text localization, segmentation, and recognition processes are also used for this detection. Figure 3 displays an example of an ALPR system.

Fig. 1 License plates in India

Fig. 2 License plates in different countries

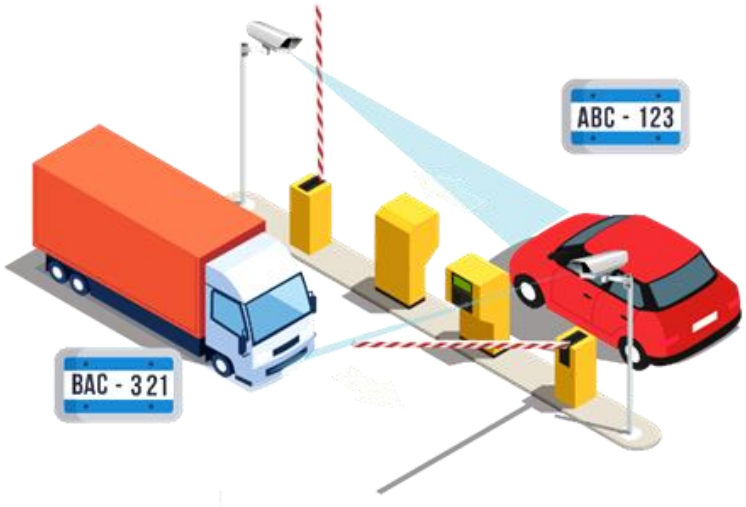

Fig. 3 An illustration of ALPR

2. Literature Review

Yi Wang et al. [5] introduced a precise and real-time LPR framework. The suggested method includes two Convolutional Neural Network (CNN) models; one is used for LP detection, and the other for recognition. It was suggested to use a vertex-estimation branch of VertexNet to rectify LPs as SCR-Net input images. A weight-sharing classifier for character recognition and a horizontal encoding approach for left-to-right feature extraction was proposed in SCR-Net. Based on the simulation results, it was found that the suggested system had superior detection and recognition performance.

An LPR system was introduced by Sergio M. Silva and Claudio Rosito Jung [6] for unconstrained scenarios in which oblique views could cause the LP to be significantly distorted. The suggested system can detect and unwarp numerous distorted LPs due to the affine transformation that regresses and maps a canonical square to the corners of the LPs. This network demonstrated remarkable adaptability in recognizing License Plates (LPs) on cars and motorcycles, even when faced with diverse lighting conditions and viewpoints. Surprisingly, it achieved this level of performance after being trained on only 693 annotated images. The simulation results showed that the suggested strategy achieved improved detection performance.

Salah Alghyaline [7] suggested an LPR system using two-stage CNNs for Jordanian LPs. CNN is built using the YOLO3 network architecture. The suggested method eliminates incorrect predictions by utilizing temporal data from several frames. The LPs of the vehicles are tracked and erroneously removed using a collection of array data structures. A new dataset was produced by this work. The experimental results on actual YouTube videos showed that the suggested method is highly effective at identifying Jordanian LPs and has improved recognition accuracy.

Heshan Padmasiri et al. [8] created a proof-of-concept and an efficient hardware-based automated LPR system for a limited environment. The suggested method works well in situations where there are drastic variations in illumination, such as during the day and night. It is only designed for low-resource, edge devices. Various datasets collected in real-time, during the day, and at night are employed to verify the proposed methods' precision, energy conservation, communication reliability, and computational speed. The suggested study's findings also demonstrated competitive performance when compared to the most advanced server-grade hardware solutions.

A system for recognizing vehicle LPs using image-processing techniques was proposed by Dilshad Islam et al. [9]. The proposed system consists of four consecutive modules: preprocessing, number plate extraction, number plate character segmentation, and character recognition. Preprocessing serves the objective of improving the quality of the image. The desired LP region is extracted by applying different morphological operations after preprocessing. The bounding box method is then used for segmentation, breaking each letter and number on the LP region into individual segments. The process of identifying every segmented character in the LP image is completed by applying template matching. The effectiveness and accuracy of the suggested system's ability to identify LP characters were demonstrated by the experimental results.

Reda Al-batat et al. [10] introduced an approach that does not rely on pre-established processing rules or take advantage of any prior experience of the LP to complete the ALPR task in a fully automated and streamlined pipeline that incorporates all three phases, consisting of a vehicle classifier. The YOLOv4 detector is used in a darknet framework for the experiments. Experiments using five distinct publicly available datasets revealed encouraging accuracy and performance results from the suggested approach.

A deep learning model for LPR was introduced by Imran Shaf et al. [11]. The plate detection process begins with the initial step of locating the plate, followed by analyzing the identified plate area to determine its contents. This algorithm relies on the 53-DCNN architecture as its foundation. To pinpoint the exact position of the LP, the input image undergoes pre-processing to enhance its quality before being divided into grid cells of suitable dimensions. CNN training involves segmenting the LP characters. The accuracy values are finally calculated after post-processing the results. The experimental findings revealed that the suggested system performed better in terms of detection and recognition.

Jithmi Shashirangana et al. [12] developed an optimal decision support solution for LPR that is only compatible with edge devices during the night. The goal of the suggested strategy was to develop a system that could be installed in isolated, ungoverned locations without a direct connection to the internet or the electrical grid. Models were developed and optimized for hardware using a Neural Architecture Search (NAS) model. The outcomes of the simulation demonstrated that the suggested approach produced superior detection and recognition outcomes.

Alif Ashrafee et al. [13] offered a 2-stage recognition pipeline combined with the vision API that offers reliably accurate detection and recognition performance and real-time inference speed. The primary MobileNet SSDv2 detection framework was filtered using a haar-cascade classifier. The pipeline offered by MobileNet SSDv2 has an extremely high detection rate, but the slower inference speed comes at a price. A temporal frame separation strategy was used to differentiate between different LPs of different vehicles in the same clip. A newly created dataset was used to examine the

suggested model. The simulation findings demonstrated that the suggested system achieved improved detection performance.

Anmol Pattanaik and Rakesh Chandra Balabantaray [14] provided a GAN-based methodology in order to reconstruct clear, high-resolution images from blurry, low-resolution ones. The first step in the detection process involves the localization of LPs using the Haar Cascade in conjunction with the IBA and CCA framework. To enhance the real-time performance and recognition rate of the LPR system, the transfer learning algorithm was used in combination with the DL tool. According to the simulation results, the proposed system achieved higher recognition accuracy.

Existing work on LPR faces several significant limitations. The performance of LPR in real-world occurrences is hampered by the fact that many studies ignore blurry or low-resolution LP images. In addition, difficulties with distance, viewing angles, lighting, and complicated backgrounds degrade recognition performance. The computational burden of Neural Architecture Search increases the need for significant computing resources to achieve high accuracy. Data disparity makes it difficult to transfer models from datasets such as ImageNet to LPR. Moreover, systems frequently experience difficulties handling complex scenes, producing broken objects, and struggling with images that contain multiple vehicles. In addition, using color spaces such as RGB or HLS imposes restrictions on illumination and noise sensitivity, which can result in incorrect processing and insufficient character extraction. The performance of LPR systems is further deteriorated by feature extraction inefficiencies, especially when non-robust features are used. This emphasizes the necessity for strong techniques to get around these restrictions. So, in order to overcome the above-mentioned limitations, an effective deep learning based novel model has been proposed in this paper.

3. Materials and Methods

The proposed methodology for LPR using deep learning involves several key steps. Initially, images are acquired and pre-processed through techniques such as gray scaling and thresholding. After that, morphological processes like dilation and erosion are used to boost the image quality. Following this, segmentation is performed to isolate the region of interest containing the LP. The contours are extracted for character estimation within this region. Finally, a CNN model is utilized to recognise LP characters, leveraging the features extracted from the segmented regions to accurately identify the LP numbers. The detailed schematics of the suggested LPR approach are visualized in Figure 4.

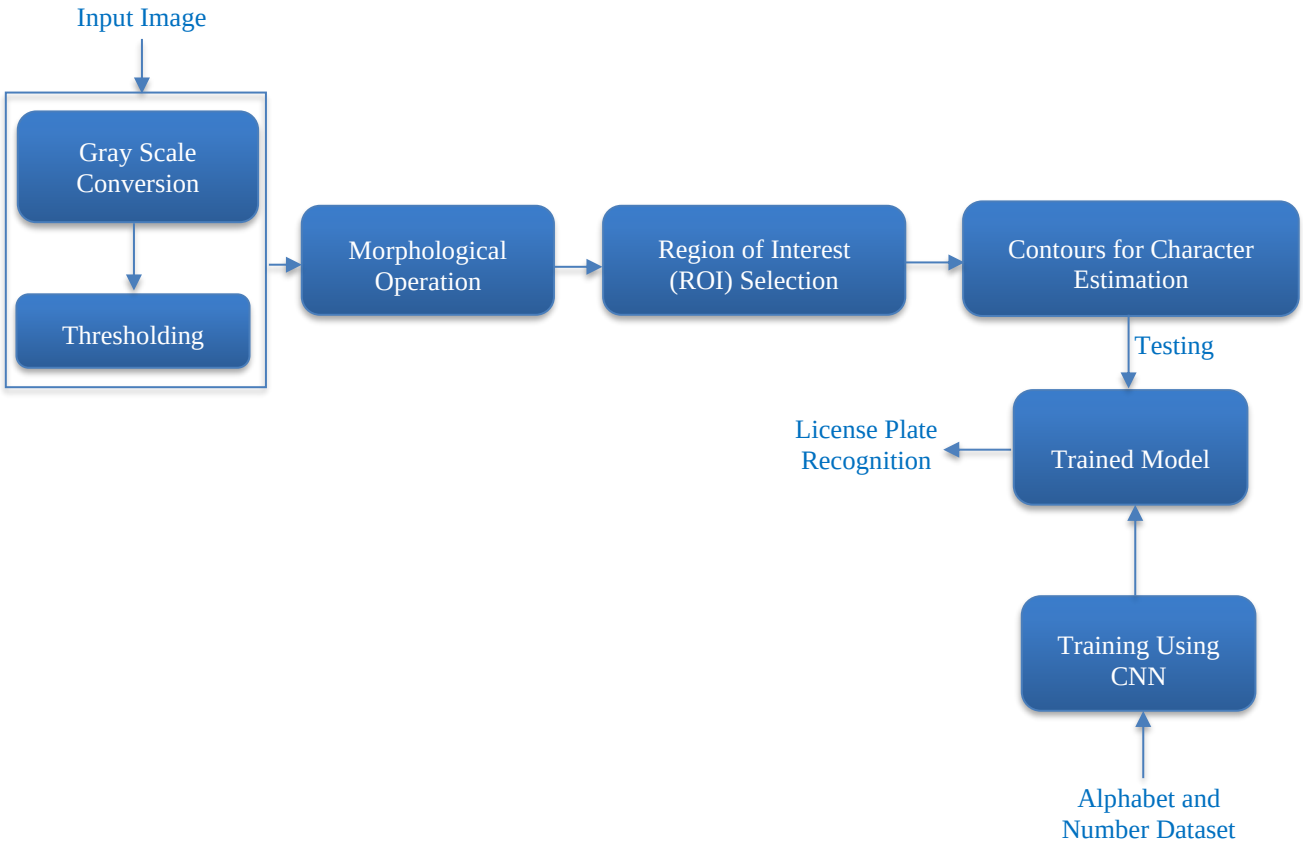

Fig. 4 Block diagram of proposed methodology

3.1. Image Acquisition and Dataset

The image acquisition phase begins its operation when the user presses the "Upload Image" button. A folder named “test_data" has been created in the current directory to store the images for processing. The "images" folder contains a diverse set of images. This diversity includes various types of vehicles, multi-line LP images, multi-font LP images, and images captured under different illumination conditions. The test data can be used to test the proposed LPR model. The 28x28-sized images of the alphabets (A–Z) and digits (0–9) compose the data for the model training. The data is also balanced.

3.2. Image Preprocessing

Image preprocessing is essential for various computer vision tasks, as it improves image quality and extracts pertinent information for further analysis. It includes a variety of methods that aid in improving and standardizing the input data for later processing. The image pre-processing techniques utilized in this study are gray scale conversion and thresholding.

3.2.1. Gray Scale Conversion

A crucial step in image processing is grayscale conversion. It is the process of reducing a colour image, which normally has Red, Green, and Blue (RGB) channels, to a single channel that represents brightness or intensity. This grayscale image simplifies the representation of the original image while preserving its essential details. The formula for grayscale conversion using weighted averages can be expressed as

$$Gray\ Scale\ Value\ (G) = (0.2989 * R) + (0.5870 * G) + (0.1140 * B) \quad (1)$$

Here, the coefficients 0.2989, 0.5870, and 0.1140 are chosen based on the luminance of the colors.

3.2.2. Thresholding

In image processing, the thresholding technique separates objects or features in an image according to their intensity levels. The basic concept involves transforming a grayscale image into a binary image, in which pixels are categorized according to a threshold value as either background (everything else) or foreground (object of interest). The ideal threshold for image segmentation is automatically determined by Otsu's method. Otsu's method is a widely used method for automatically figuring out the ideal threshold value for image thresholding [15]. Optimizing the between-class variance of pixel intensities yields the optimal threshold for distinguishing between the foreground and background classes. Otsu's approach maximizes the between-class variance of pixel intensities to automatically determine the ideal threshold value. The idea is to select the threshold that maximizes the variance between classes (inter-class variance) and minimizes the variance within each class. After iterating through all potential values, the algorithm determines a measure known as "between-class variance" for each threshold. The ideal threshold is determined by taking the value that maximizes the variance between classes. To accomplish this task, a threshold value ranging from 0 to 255 is employed. In this case, the threshold value has been set to 200. The thresholding process can be formulated as

$$Binary\ Image\ (x,y) = \begin{cases} 255, & if\ G(x,y) \geq 200 \\ 0, & otherwise \end{cases} \quad (2)$$

3.2.3. Morphological Operations

Morphological operations are essential to image processing because they alter the geometric pattern of objects within an image. Morphological operations help to fill gaps, join nearby structures, enhance object boundaries, filter out noise, and fill in gaps by selectively shrinking or expanding regions of interest. Erosion and dilation are the two different morphological processes that are used in the suggested study [16].

The binary image is prepared for the erosion process, which aims to exclude undesired pixels from the object's boundary. Specifically, those pixels are erroneously marked with a value of 1 that should be 0. Erosion involves a straightforward procedure where each pixel in the image is individually assessed. For each pixel, its neighboring pixels (the number determined by the kernel size) are examined. The pixel retains a value of 1 only if all neighboring pixels are also 1; otherwise, it is assigned a value of 0. This method systematically refines the object's boundaries by iteratively removing erroneous pixels. Mathematically, erosion can be formulated as

$$Output\ (x,y) = \min_{i,j \in SE} \{Input(x+i, y+j\} \quad (3)$$

Where output (x,y) is the value of the pixel at coordinates (x,y) in the output image, input(x+i,y+j represents the pixel values in the neighborhood specified by the Structuring Element (SE), $\min_{\top}$(i,j∈SE) denotes the minimum operation overall pixel values in the neighborhood. The visual representation of erosion operation is illustrated in Figure 5.

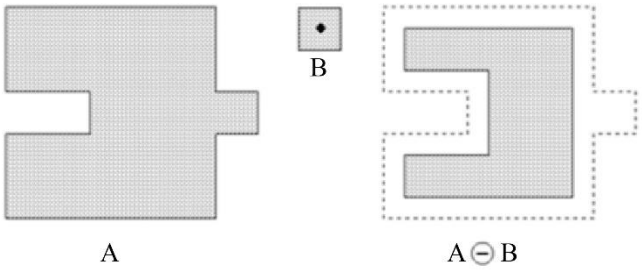

Fig. 5 Basic operation of erosion

As a morphological operation, dilation enlarges or expands the boundaries of the foreground elements in a binary image. The process entails traversing the input image with a SE and updating each pixel with the maximum pixel value within the neighborhood defined by the SE. It evaluates each pixel in the image individually and examines its neighbors (the count of neighbors is determined by the kernel size). If at least one neighboring pixel is set to 1, the pixel under consideration is assigned a value of 1. Dilation can be mathematically formulated as

$$Output\ (x, y) = \max_{i,j \in SE}\{Input(x + i, y + j\} \quad (4)$$

Where max┬(i,j∈SE) denotes the maximum operation overall pixel values in the neighborhood. The dilation operation can be visually represented in Figure 6.

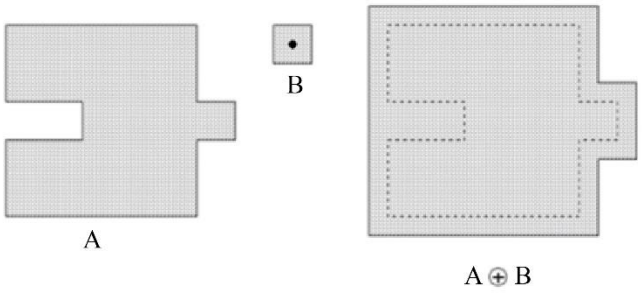

Fig. 6 Basic operation of dilation

3.2.4. Segmentation of Alphanumeric Characters from License Plate

A clean binary image can be obtained after pre-processing and morphological operations. These images are further processed for segmentation of alphanumeric characters from LP. Image segmentation is a crucial stage in image processing, which entails breaking an image into segments or significant areas [17]. The first step is to locate every contour in the binary LP image. Constant curves or boundaries of objects with the same color or intensity are represented by contours. A bounding rectangle is computed for every contour after it has been identified. This rectangle is the smallest that can completely enclose the contour. It aids in defining the precise location of every character in the image of the LP. The Region of Interest (ROI) containing each contour is then visualized by drawing bounding rectangles around them. This step aids in comprehending the bounding rectangle computation and contour detection efficacy. The term ROI describes the precise area or region that includes an image's interesting objects or features. A dimension comparison is done to remove the necessary rectangles containing the characters. The following conditions are typically applied:

- The width of the bounding rectangle is determined by dividing the length of the image by the number of characters and ensuring it falls within the range of 0.
- The length of the bounding rectangle falls within a range of (width of image) / 2, 4 * (width of image) / 5.

The remaining bounding rectangles are deemed to contain individual characters after filtering based on dimension comparison. The original LP image extracts binary images of each character using these rectangles. These binary images can be further processed for character recognition or analysis. They represent individual characters. The contours and segmented individual characters are displayed in Figure 7.

3.2.5. Convolutional Neural Network for LPR

CNN is a feed-forward, deep neural network that is widely employed for visual imagery analysis. It functions similarly to how humans perceive things. CNNs stand as one of the most frequently utilized neural network architectures for handling high-dimensional datasets. CNNs function remarkably similarly to conventional neural networks. After receiving some inputs, each neuron executes a dot product and may or may not follow it with non-linearity. A single, differentiable score function that runs the length of the network maintains the link between raw image pixels at one end and class scores at the other [18]. Furthermore, the last layer of CNNs offers a loss function. Figure 8 depicts the multiple hierarchical layers in which millions of neurons are arranged within a typical CNN.

The three basic layers that compose the CNN architectures are convolutional, pooling, and Fully Connected (FC). Pooling layers are optional and sometimes partially connected to minimise the input images' size. The primary function of an output, or Fully Connected (FC) layer, is classification. The networks' learnability is dictated by the hidden layers present in the convolution and FC layers. CNNs are characterized by their depth, which is directly proportional to the number of layers they contain. A deeper layer yields higher-level feature extraction [19]. Scaling higher-resolution images is facilitated by communication between neurons in the hidden layer and subsequent layer neurons. The input layer neurons in CNN processing are activated by visual stimuli. The main function of the convolution layer revolves around extracting features from the image. Subsequently, these extracted features are utilized by the output layer to propel them into the hidden layers for computation. Activation functions frequently help transfer important information between hidden layers so that the subsequent layers can use it.

Convolution is a mathematical process that applies a filter effect to the input images. A mathematical process known as convolution takes two functions as inputs and outputs a third function that is simply an improved version of one of the original two functions. The convolution operation can be formulated as

$$y\ [m, n] = \sum_{i=0}^{k-1} \sum_{j=0}^{k-1} x[m + 1, n + j].w\ [i, j] \quad (5)$$

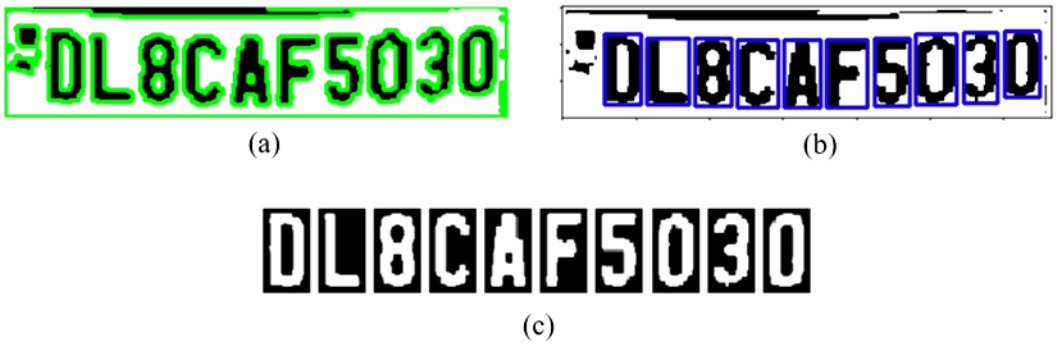

Fig. 7 (a) All contours in the LP image (b) Contours of individual characters (c) Segmented individual characters

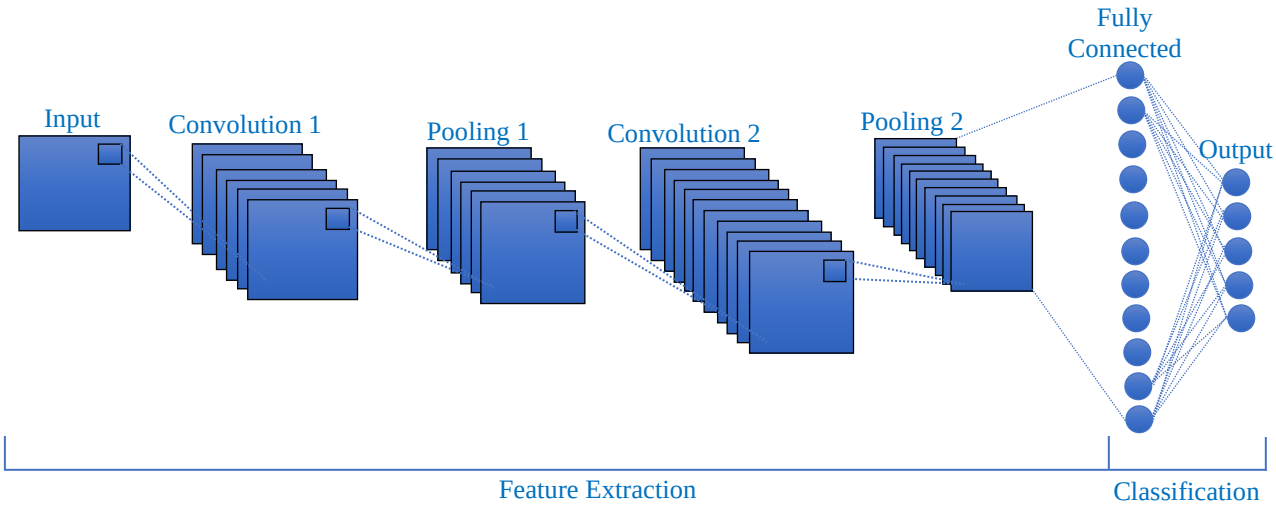

Fig. 8 Basic architecture of CNN

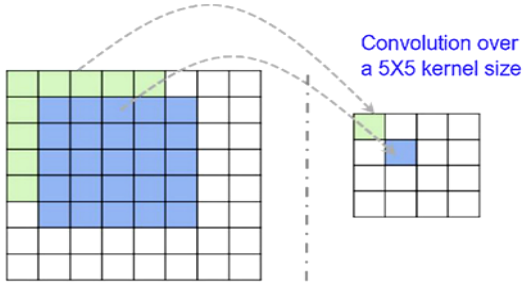

Fig. 9 Visualization of convolution operation

Where y [m, n] is one data word in the output, x [m, n] is an input data word, and w [m, n] are values of the filters. In the convolution process of an image, a pixel and its surrounding pixels are multiplied by a small filter matrix known as the kernel, and the resultant output central pixel is then added up. The input image is subjected to a convolution operation by the CNN's convolutional layers, which then pass the outcome to the subsequent layer. This layer extracts various features from the input image to aid in subsequent analysis. The initial layer is a convolutional layer with 32 output filters that employ the Rectified Linear Unit (ReLU) as the activation function and a convolution window of size (5,5). A filter/kernel (5x5 matrix) is applied to the input image to obtain the convolved feature. This convolved feature is then passed on to the next layer. The convolution operation is visualized in Figure 9.

A CNN is composed of a neuron's activation function, which adds non-linearity to neural networks. The neural network is more capable of learning intricate patterns in the input data due to the activation function. A perfect activation Function is differentiable and nonlinear. In neural networks, ReLU is the most frequently utilized activation function. The ReLU function is expressed as

$$f(x) = \max(0, x) \tag{6}$$

$$ReLU(x) = \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases} \tag{7}$$

The ReLU function deactivates neurons rather than activating all neurons at once when the output of the linear transformation is lower than zero. Since this function activates a specific number of neurons, it is computationally more efficient than the sigmoid and tanh functions. The ReLU function can be visualized in Figure 10.

The convolution layer of a CNN generates feature maps, which are downsized by a pooling or down sampling layer. The depth of the convolution layer remains unchanged in the pooling layer. In addition to helping to prevent overfitting, the pooling layer reduces computational costs by removing superfluous spatial information [20]. The output of neurons from the preceding layer of convolutional networks can be combined into a single neuron for the following layer using local or global pooling layers. The two most commonly employed techniques for layer pooling are average pooling and maximum pooling. The process of pooling that selects the largest element from the feature map region where the filter intersects is known as max pooling. The feature map is produced by the max-pooling layer would, therefore, contain the most notable features from the prior feature map. An input representation (image, hidden-layer output matrix, etc.) must be downscaled to lower its dimensionality and infer characteristics from the binned sub-regions. The max pooling operation is visualized in Figure 11.

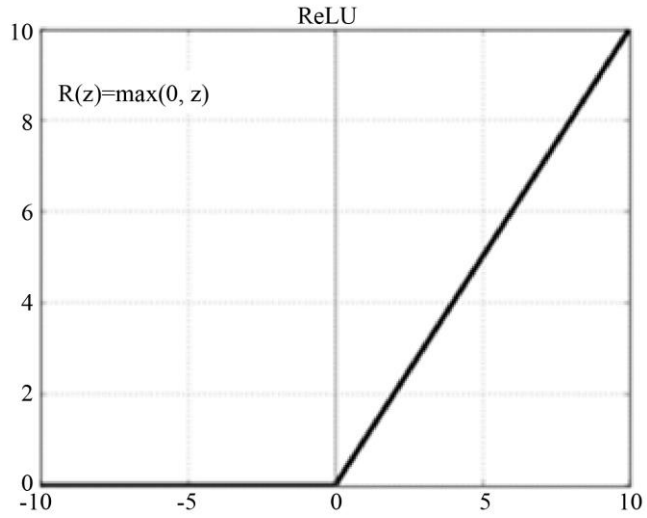

Fig. 10 ReLU activation function

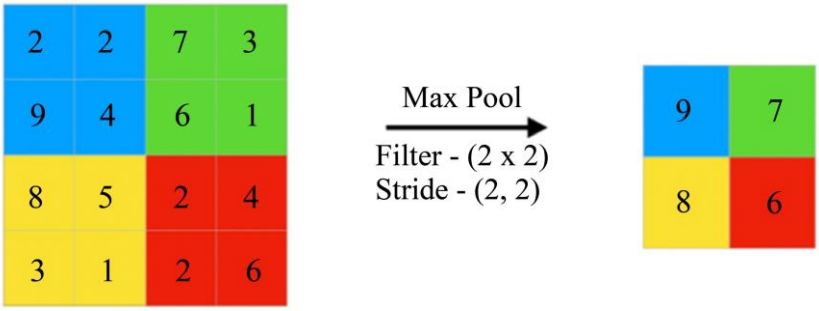

Fig. 11 Maxpooling operation

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 28, 28, 32)	55328
max_pooling2d (MaxPooling2D)	(None, 14, 14, 32)	0
dropout (Dropout)	(None, 14, 14, 32)	0
flatten (Flatten)	(None, 6272)	0
dense (Dense)	(None, 128)	802944
dense_1 (Dense)	(None, 36)	4644

Total params: 862,916
Trainable params: 862,916
Non-trainable params: 0

Fig. 12 Model summary of the proposed approach

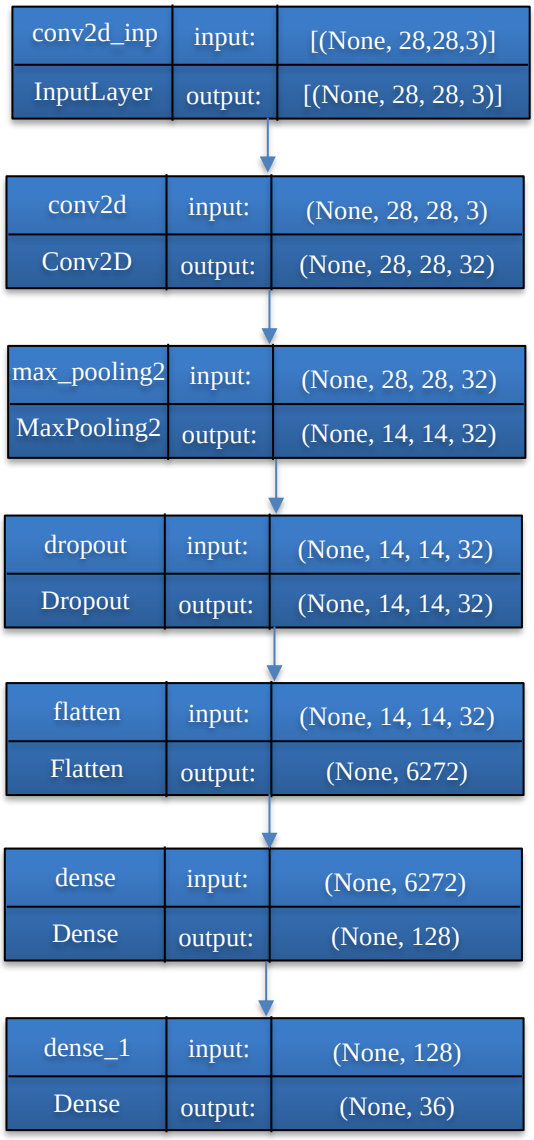

Fig. 13 Proposed model architecture

In order to avoid overfitting, a critical part of CNNs is the dropout layer. During each training iteration, a specific percentage of neurons are dropped randomly, forcing the network to learn redundant representations. Dropout can improve the network's ability to generalize by reducing its dependence on individual neurons. The proposed CNN model utilizes a dropout rate of 0.4, which implies that 60% of the nodes are preserved during training. Following the dropout layer, the data from the preceding layer is flattened into a single dimension. Subsequently, two dense layers are incorporated into the CNN architecture. The initial dense layer has an output space dimensionality of 128, with the ReLU activation function applied. The final layer, responsible for categorizing characters into 36 classes (26 alphabets from A to Z and 10 digits from 0 to 9), consists of 36 output nodes and employs the softmax activation function. The model summary is visualized in Figure 12, while Figure 13 illustrates the model's structure.

Table 1. Hyperparameters

Hyperparameters	Values
Loss Function	Categorical Crossentropy
Optimizer	Adam
Learning Rate	0.00001
Batch Size	1
Number of Epochs	150

4. Results and Discussion

4.1. Simulation Setup

The proposed model has been trained and tested using the Google Collaboratory platform. Table 1 contains a tabulation of the different hyperparameters used in this study.

4.2. Performance Evaluation

The suggested model was examined using the alphanumeric dataset following model training. The recognition performance was measured in terms of accuracy. Recognition accuracy is a critical metric in LPR systems, determining the system's effectiveness in accurately identifying LP numbers from images. High recognition accuracy ensures reliable performance in various conditions, such as varying lighting, angles, and plate types. According to the simulation results, the suggested approach achieved 99.54 %. Figure 14 demonstrates the accuracy plot of the suggested model.

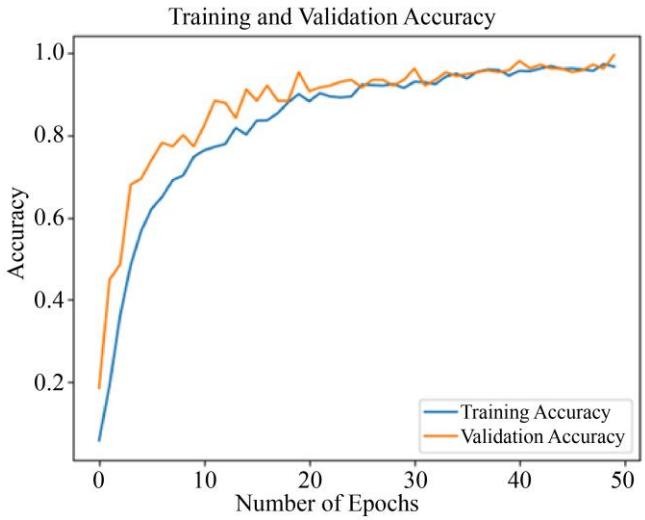

Fig. 14 Accuracy plot of proposed model

The accuracy plot serves as a visual representation of the performance of a model over epochs. It shows the progression of the model's accuracy on a training dataset during the training process. As epochs progress, the accuracy plot allows us to observe trends in the model's learning behavior, such as convergence or fluctuations in performance. Analyzing the accuracy plot is crucial for understanding how well the model is learning from the data and whether adjustments need to be made to the training

process, such as modifying hyperparameters or introducing regularization techniques. Ultimately, the accuracy plot provides valuable insights into the training dynamics of the framework and aids in making informed decisions to improve its overall performance. Figure 15 provides the loss plot of the suggested approach.

The loss plot illustrates the changes in the loss function's value over successive epochs or iterations during the training phase. The loss function evaluates the difference between the predicted output and the true ground truth, acting as a measurable indicator of how well the model performs. Typically, a decreasing trend in the loss plot indicates that the model is effectively learning to minimize its prediction errors. The recognition outcomes achieved from the suggested LPR model are visualized in Figures 16, 17 and 18, respectively.

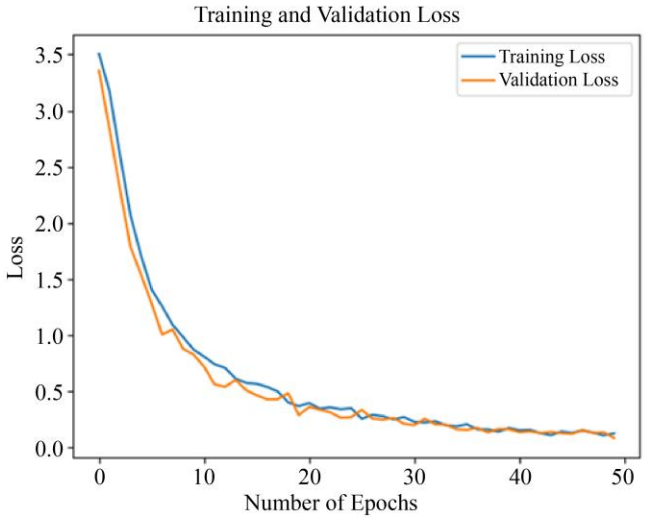

Fig. 15 Loss plot of proposed model

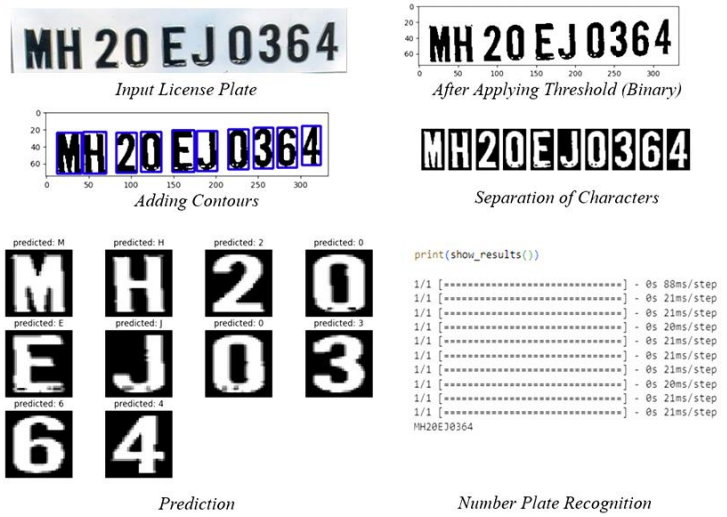

Fig. 16 Prediction of number plate "MH20EJ0364"

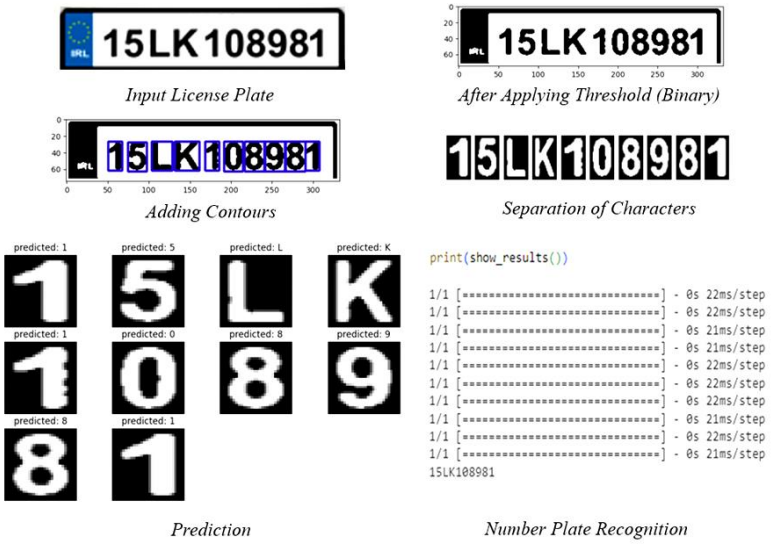

Fig. 17 Prediction of number plate "15LK108981"

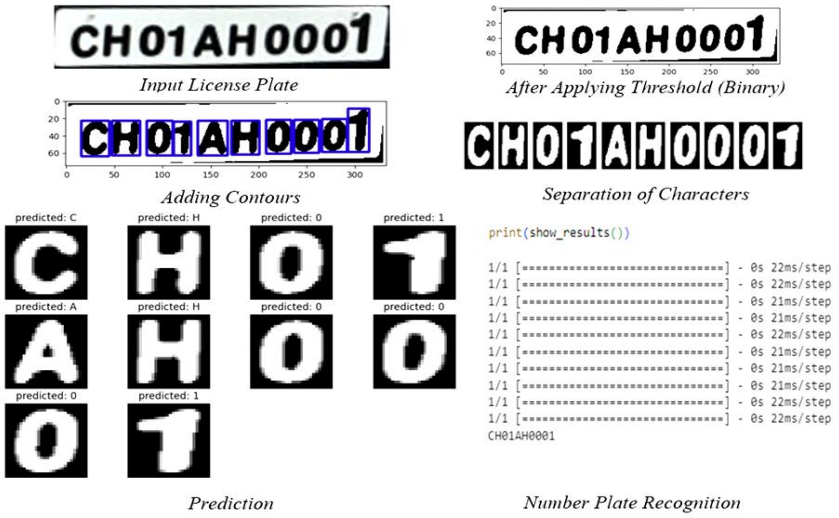

Fig. 18 Prediction of number plate "CH01AH0001"

Table 2 provides a performance comparison of the suggested LPR system with existing approaches. YOLO-Darknet Deep Learning and Deep Neural Networks achieved 78% and 95% accuracy, respectively, while YOLO and Faster R-CNN demonstrated higher accuracies of 95.33% and 97%. Artificial Neural Networks and Optimal K-means with CNN both reached accuracies of 98.1%.

Table 2. Performance comparison

Authors	Methodology	Recognition Accuracy
Hendrya and Rung Ching Chen [21]	YOLO- Darknet Deep Learning	78 %
Sergey Zherzdev and Alexey Gruzdev [22]	Deep Neural Network	95 %
Rayson Laroca et al. [23]	YOLO	95.33%
Naaman Omar et al. [24]	Faster R-CNN	97 %
İbrahim Türkyılmaz and Kirami Kaçan [25]	Artificial Neural Network	97 %
Irina Valeryevna Pustokhina et al. [26]	Optimal K-means with CNN	98.1 %
Proposed Method		**99.54 %**

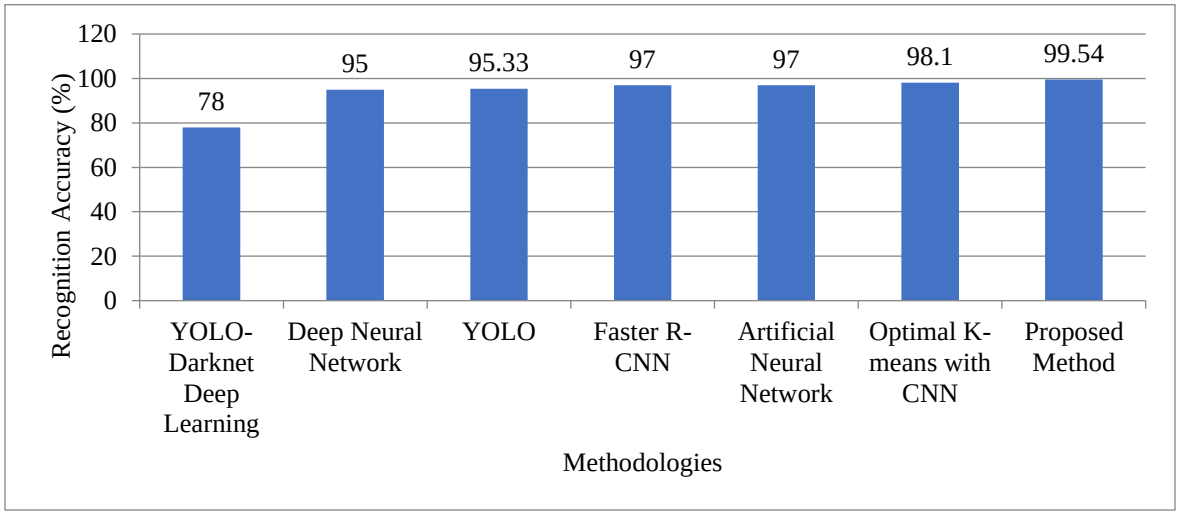

Fig. 19 Performance comparison

Notably, the proposed method stands out with an impressive recognition accuracy of 99.54%, surpassing all other techniques. This comparison underscores the efficacy of the proposed approach, indicating its superiority in accurately identifying and decoding LP information from images compared to existing approaches. Figure 19 provides a graphic representation of the performance comparison between the suggested and existing approaches.

5. Conclusion

Transportation infrastructure has experienced rapid expansion over the past decade, leading to a significant increase in traffic volume. Therefore, there is a crucial need to closely monitor road traffic, which remains the primary mode of transportation. This study proposed an efficient LPR system utilizing deep CNNs. The methodology involves several key stages, including image acquisition and preprocessing techniques such as grayscale conversion and thresholding, succeeded by morphological processes like erosion and dilation to enhance image quality. Subsequent segmentation isolates the region of interest, and character estimation is performed by extracting contours within this area. Leveraging a CNN model for accurate character recognition, features extracted from segmented regions contribute to achieving impressive recognition accuracy of 99.54%. The simulation outcomes highlight the efficacy of the proposed approach in precisely detecting and recognising license plate numbers from images. The demonstrated performance suggests significant potential for real-world applications in traffic management, law enforcement, and intelligent transport systems, providing a reliable and efficient solution for automated license plate recognition tasks.

Funding Statement

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

Acknowledgments

I would like to express my sincere gratitude to all those who contributed to completing this research paper. I extend my heartfelt thanks to my supervisor, my family, my colleagues and fellow researchers for their encouragement and understanding during the demanding phases of this work.

References

[1] Muhammad Alam, Joaquim Ferreira, and José Fonseca, *Introduction to Intelligent Transportation Systems*, Intelligent Transportation Systems, Studies in Systems, Decision and Control, vol. 52, pp. 1-17, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[2] C.N.E. Anagnostopoulos et al., "A License Plate-Recognition Algorithm for Intelligent Transportation System Applications," *IEEE Transactions on Intelligent Transportation Systems*, vol. 7, no. 3, pp. 377-392, 2006. [CrossRef] [Google Scholar] [Publisher Link]

[3] The Tamil Nadu Motor Vehicles Rules, Transportinfo, pp. 1-221, 1989. [Online]. Available: https://transportinfo.in/forms/tnmvr1989.pdf

[4] M. Shridhar et al., "Recognition of License Plate Images: Issues and Perspectives," *Proceedings of the Fifth International Conference on Document Analysis and Recognition*, Bangalore, India, pp. 17-20, 1999. [CrossRef] [Google Scholar] [Publisher Link]

[5] Yi Wang et al., "Rethinking and Designing a High-Performing Automatic License Plate Recognition Approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 8868-8880, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[6] Sergio M. Silva, and Cláudio Rosito Jung, "A Flexible Approach for Automatic License Plate Recognition in Unconstrained Scenarios," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 6, pp. 5693-5703, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[7] Salah Alghyaline, "Real-Time Jordanian License Plate Recognition using deep Learning," *Journal of King Saud University-Computer and Information Sciences*, vol. 34, no. 6, pp. 2601-2609, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[8] Heshan Padmasiri et al., "Automated License Plate Recognition for Resource-Constrained Environments," *Sensors*, vol. 22, no. 4, pp. 1-29, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[9] Dilshad Islam, Tanjim Mahmud, and Tanjia Chowdhury, "An Efficient Automated Vehicle License Plate Recognition System Under Image Processing," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 29, no. 2, pp. 1055-1062, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[10] Reda Al-Batat et al., "An End-to-End Automated License Plate Recognition System Using YOLO Based Vehicle and License Plate Detection with Vehicle Classification," *Sensors*, vol. 22, no. 23, pp. 1-17, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[11] Imran Shafi et al., "License Plate Identification and Recognition in a Non-Standard Environment Using Neural Pattern Matching," *Complex & Intelligent Systems*, vol. 8, pp. 3627-3639, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[12] Jithmi Shashirangana et al., "License Plate Recognition using Neural Architecture Search for Edge Devices," *International Journal of Intelligent Systems*, vol. 37, no, 12, pp. 10211-10248, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[13] Alif Ashrafee et al., "Real-Time Bangla License Plate Recognition System for Low Resource Video-Based Applications," *2022 IEEE/CVF Winter Conference on Applications of Computer Vision Workshops (WACVW)*, Waikoloa, HI, USA, pp. 479-488, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[14] Anmol Pattanaik, and Rakesh Chandra Balabantaray, "Enhancement of License Plate Recognition Performance using Xception with Mish Activation Function," *Multimedia Tools and Applications*, vol. 82, pp. 16793-16815, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[15] Sunil L. Bangare et al., "Reviewing Otsu's Method for Image Thresholding," *International Journal of Applied Engineering Research*, vol. 10, no. 9, pp. 21777-21783, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[16] K.A.M Said, and A.B Jambek, "Analysis of Image Processing using Morphological Erosion and Dilation," *Journal of Physics: Conference Series*, vol. 2071, pp. 1-7, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[17] H.J. He, C. Zheng, and D.W. Sun, *Chapter 2 - Image Segmentation Techniques*, Computer Vision Technology for Food Quality Evaluation (Second Edition), Academic Press, pp. 45-63, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[18] Jianxin Wu, "*Introduction to Convolutional Neural Networks*," National Key Lab for Novel Software Technology, Nanjing University, pp. 1-31, 2017. [Google Scholar] [Publisher Link]

[19] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton, "Deep Learning," *Nature*, vol. 521, pp. 436-444, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[20] Hossein Gholamalinezhad, and Hossein Khosravi, "Pooling Methods in Deep Neural Networks, A Review," *arXiv*, pp. 1-169, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[21] Hendry, and Rung-Ching Chen, "Automatic License Plate Recognition via Sliding-Window Darknet-YOLO Deep Learning," *Image and Vision Computing*, vol. 87, pp. 47-56, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[22] Sergey Zherzdev, and Alexey Gruzdev, "LPRnet: License Plate Recognition via Deep Neural Networks," *Arxiv*, pp. 1-6, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[23] Rayson Laroca et al., "A Robust Real-Time Automatic License Plate Recognition Based on the YOLO Detector," *2018 International Joint Conference on Neural Networks*, Rio de Janeiro, Brazil, pp. 1-10, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[24] Naaman Omar et al., "Fused Faster RCNNs for Efficient Detection of the License Plates," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 19, no. 2, pp. 974-982, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[25] İbrahim Türkyılmaz, and Kirami Kaçan, "License Plate Recognition System using Artificial Neural Networks," *ETRI Journal*, vol. 39, no. 2, pp. 163-172, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[26] Irina Valeryevna Pustokhina et al., "Automatic Vehicle License Plate Recognition using Optimal K-means with Convolutional Neural Network for Intelligent Transportation Systems," *IEEE Access*, vol. 8, pp. 92907-92917, 2020. [CrossRef] [Google Scholar] [Publisher Link]