Spectrum of Engineering Sciences
ISSN (e) 3007-3138 (p) 3007-312X Volume 3, Issue 3, 2025

AN EFFICIENT METHOD TO LOCATE LICENSE PLATES UNDER DIVERSE ENVIRONMENTS

Izhar Khan*[1], Harris Sarfraz[2], Muhammad Bilal[3], Muhammad Masood Ur Raheem[4], Rabia Farooq[5]

[1,2]*COMSATS University Islamabad, Abbottabad campus*
[3]*COMSATS University Islamabad*
[4]*COMSATS University Islamabad, Sahiwal Campus*
[*5]*Department of Computer Science, University of Okara*

[*1]izharqazi839@gmail.com

DOI: https://doi.org/10.5281/zenodo.15099895

Keywords
Automatic License Plate Recognition, License Plates Detection, Region Convolutional Neural Network (RCNN).

Article History
Received on 19 February 2025
Accepted on 19 March 2025
Published on 28 March 2025

Corresponding Author: *

Abstract
Advances in artificial intelligence (AI) have made car and road object identification more efficient. Researchers face a difficult problem in detecting license plates (LPD), which necessitates the use of a dependable and precise automatic detection method. Recent approaches to deep learning have showed promise, although they are frequently limited to specific locales or private datasets. This study focuses on accurately finding license plate locations using machine vision and deep learning approaches, especially under adverse weather situations such as blurriness, nighttime, and high brightness. A modified RCNN model is developed for LPD, which uses the China City Parking Dataset (CCPD) to forecast nine different stages. The suggested method includes data preprocessing, weight generation, and model training. The model is taught using self-generated weights rather than pre-learned weights, and it detects number plates with 98% accuracy under a variety of weather situations. After 50 epochs of training, the detection module predicts suitable bounding boxes. The Region Convolutional Neural Network (RCNN) is highly effective in detecting number plates under diverse environments.

INTRODUCTION

Automatic License Plate Recognition (ALPR) is essential in many facets of daily life, particularly in urban regions with expanding populations. ALPR provides a technology solution for managing traffic congestion and optimizing resource allocation by automating license plate recognition. License plates are unique identifiers for vehicles and are essential for good traffic control [1]. However, identifying and recognizing license plates can be difficult due to differences in size, text style, color, location, and environmental factors such as illumination or weather. Additionally, recognition methods may differ by area. Furthermore, variables such as shifting backdrops, illumination, and vehicle speeds must be considered [2].

Recent advances in computer vision technology have had a considerable impact on automobile license plate identification. Hubel and Wiesel introduced the first perceptron-based technique to object identification in 1963, drawing inspiration from biology and emphasizing edge information [3]. The 1970s saw an increase in three-dimensional modeling, stereoscopic vision, and the first use of artificial intelligence (AI) in computer vision. Since the 1980s,

computer vision has advanced significantly with the development of mathematical algorithms such as Canny edge detection and pattern recognition. In the 1990s, machine learning became popular for tasks such as recognition, detection, and segmentation. As a result, computer vision has evolved from depending primarily on signal processing to using a variety of approaches, including machine learning. This article combines the efficiency of traditional signal processing techniques (such as the Sobel operator and morphological operations) with the precision of machine learning approaches, particularly convolutional neural networks (CNNs) [4].

License plate recognition normally consists of three stages: picture pre-processing, license plate localization, and character recognition. The goal of image preprocessing is to minimize noise and make plate boundary extraction more efficient [5]. This includes converting the image to grayscale, using a Gaussian blur to remove noise, and using the Sobel operator to improve character readability. Finally, the image is binarized into a binary image, with each pixel having a value of 0 or 1, greatly simplifying subsequent processing [6].

Researchers continue to face challenges in developing a highly accurate, high-speed ALPR system capable of handling low-resolution or skewed images [7]. ALPR systems have numerous applications, including forensics, traffic management, and public parking. While governments and private enterprises frequently use specialized cameras, police car cameras encounter issues due to vehicle speed, angles, and severe weather conditions, which often result in insufficient resolution, especially in certain weather scenarios. A dependable ALPR system is essential since license plates provide vital vehicle information [8]. Low image quality, blurriness, uneven illumination, variable font styles, amount of characters, size, color, orientation, and complex backdrops observed in different nations or even provinces within the same country, are some of the challenges encountered here. These elements add to the overall difficulty of the task [9].

Figure 1: Examples of License Plate Variations within a Single Country (Canada) [9]

Despite major advances, identifying license plates remains a difficult task, as seen in Figure 1. Current license plates use a variety of fonts, some of which are difficult or impossible for machines to read [10]. However, recent advances in deep learning approaches have had a substantial impact on progress in vehicle identification, license plate recognition, and optical character recognition [11]. Deep Convolutional Neural Networks (CNNs) have been particularly beneficial in developing software for vehicle and license plate recognition [9, 12].

Many commercial ALPR systems use deep learning techniques, including Open-ALPR, Sighthound, Genetec ALPR, and Amazon Recognition. However, most cutting-edge systems today have limits. Some can only recognize license plates from the front of a car, while others focus on certain regions such as the United States or China. These geographical variations are frequently adjusted to satisfy unique needs, such as parking validation or toll monitoring. Modern detection algorithms continue to struggle with accurate license plate detection. As a result, a

powerful, dependable, and precise license plate detection algorithm is required to detect fraudulent activity, stolen automobiles, or accidental vehicles under various daytime and environmental situations.

A. Comparison of Different Methods

In this section, we present a comprehensive comparison of various techniques used for license plate detection. The table below summarizes the accuracy, merits, and demerits of each method, along with the corresponding dataset used for evaluation. The techniques have been evaluated on various challenges such as extreme brightness scenarios, variations in scale and illumination, cluttered backgrounds, and more.

The goal of license plate detection is to achieve high accuracy while considering the computational cost and execution time. As we examine each approach, we also highlight their strengths and weaknesses, shedding light on their ability to handle specific challenges encountered in real-world scenario.

Table 1: Summary of few methods that achieved license plate detection accuracy on few datasets.

Reference	Techniques	Accuracy	Merits	Demerits	Dataset (images)
[9]	Two CNN Network	99.5%	High detection rate. Low execution time.	Fails to handle extreme brightness scenario.	200
[13]	YOLOv3	95.05%	Effectively dealing variance in: Scale, illumination. LP types	Fails at extreme observation angle.	5719
[14]	YOLO	98%	High accuracy in light and background variations	Computation cost high. Not scale invariant.	640
[15]	De-skewing filtering and K-nearest algorithm	98.02%	High detection rate	Execution time is high.	101
[16]	Mobile Nets +inception-v3 SSD	91.76%	Dealing well with: Cluttered background and illumination variations.	High execution time.	590
[17]	Tensor flow KERAS deep learning	96.36%	High detection rate.	Computational cost is high	34580
[18]	CNN and HAAR classifier	96.92%	Decreased Execution time. Reduced search space	Accuracy compromised for small objects in the image	390
[19]	CNN+ wavelet decomposition	98.4%	High Detection Accuracy	Inefficient use of convolutional network.	2049
[20]	Faster R-CNN with LPLM	99.04%	High Detection Accuracy in: Low resolution. Poor image quality.	Extreme reflective glare	4000
Proposed technique	Modified- RCNN		High detection accuracy in Low resolution, Poor image, quality, variations in angle, shape, colour.		290,000

B. Aims and Objectives

This research aims to contribute to automatic license plate detection in images through the following:

- Developing a state-of-the-art license plate detection algorithm using advanced machine learning and deep learning concepts. This method is expected to achieve high license plate detection accuracy.
- Analysing the developed method for challenging situations like bad weather conditions (rainy days, dusty environments) and whitish image dominance (snow, bright light). Furthermore, an

advanced algorithm will be developed to handle license plate detection under these conditions.

2. METHODOLOGY

Figure 2 illustrates the comprehensive workflow of the proposed methodology of this research work. It is divided into three main parts. The first part involves data pre-processing, which includes extracting relevant data points and dividing the images into separate training and testing sets. The second step of the proposed approach focuses on localizing the images. This involves generating weights and training the proposed model using the training data and corresponding ground truth information (shape file). By leveraging the generated weights and training data, a well-trained model is developed specifically for number plate detection. Finally, after the training phase, the model is tested using the testing dataset, and the accuracy results are estimated. This step provides an evaluation of the model's performance.

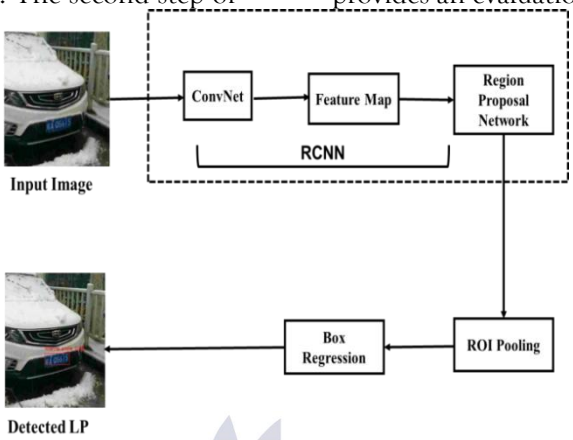

Figure 2: Proposed Architecture

A. Data Pre-processing

The purpose of the pre-processing step is to identify the appropriate pattern of input data so that it may be incorporated into the updated CNN model. We have over 272,000 photos in our collection. First, we take these photos and extract the data points from them. The next step is to create a.csv file containing these data points. Second, we separate the dataset into segments that will be used for training, validation, and testing.

B. Proposed model

Convolutional layers, pooling layers, and fully-connected layers are the three types of layers that are commonly found in a basic convolutional neural network (CNN). The architecture of the model that was developed in this study is different from that of a CNN. Instead, the structure of the model that is being offered is significantly more intricate and complex. RCNN (Region-based Convolutional Neural Network) is a popular object detection algorithm that has been used in a variety of real-world applications. It is known for its high accuracy, scalability, robustness, and its ability to detect objects of different sizes and scales.

C. Layers used in the proposed model

1) Convolutional Layer: This layer is the most significant one. The bulk of the calculation is carried out in this layer. The equation that is used to construct the extracted features of the map is as follows:

$$(i,j) = (I \times K)\,(i,) = \sum\sum (i-m, j-n) I(m,n) \quad 1$$

Where i and j represent the row number and the column number of an image, respectively, while m and n stand for the serial number included within the kernel. I is used to represent the input image, and K is shorthand for the 2D kernel. Following the completion of the convolution, the output is saved in S. Additionally; the following equation will be used to determine the relationship between the size of the input and the size of the output.

$$W = W - F - 2P / S + 1 \quad 2$$

$$H = H - F - 2P / S + 1 \quad 3$$

$$D = \mathrm{K} \quad 4$$

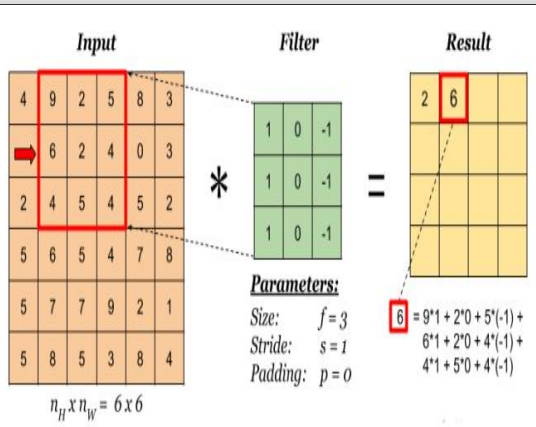

Figure 3: Convolutional Operation

Where W and H represent the input, P represents the size of the input after padding, and F denotes the size of a square kernel. While S refers to the value of the stride. Additionally, the output dimension, denoted by the letter D, is equal to the number of kernels, denoted by the letter K [77].

The figure 3 describes the basic convolution operation. Where each element of the sliding window is multiplied by a filter. And add the results. Then based on the stride value the sliding window is moved to the next position. In this figure, the stride value is one. So, it moves single position. And do the same calculation to get the next results.

2) Pooling Layer: The pooling layer takes the place of the convolutional layer in a straightforward CNN. The objective of the pooling layer is to bring the dimensionality down while simultaneously introducing a hierarchical structure. Figure 4 illustrates the fundamental process of maximum pooling, as well as the average pooling layer. The following diagram provides an illustration of a maximum and an average pooling layer. Where there is a maximum pooling layer with dimensions of two by two and a stride of two. Within each block, the output is determined by selecting the value that is the greatest. In the average pooling layer, it adds all the value of block each block separately. And then divide it into a total number of values in a block as shown in the figure. In our proposed model many max pooling layers are used.

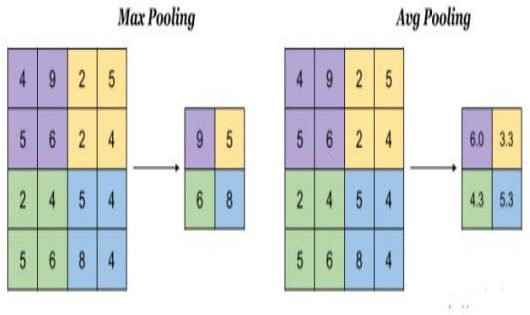

Figure 4: Max and Average Pooling Operation

3) Batch Normalization: It is the most frequently used method of normalization; the objective of batch normalization is to normalize the convolutional output. Additionally, it keeps the standard deviation close to one and the mean value close to zero. The following equation is used for normalization.

$$\hat{x}_l = \frac{x_i - \mu_\beta}{\sqrt{\delta_\beta^2 + \varepsilon}} \quad 5$$

Batch normalization has been shown to effectively increase the speed of convergence, which makes it possible to employ a faster learning rate, reduces the amount of over fitting, and shortens the amount of time spent in training.

4) Activation Function: After the input has been normalized, the activation function will be applied to it. In most cases, two different activation functions will be utilized. One is a sigmoid function, and the other is a ReLU function. Both of the activation functions provide outcomes that are not binary and are therefore non-linear. The activation function's job is to evaluate the output and decide whether it should be accepted or rejected for the procedure that comes next. The formula for calculating the ReLU activation function is max (0, x), where x is the result of applying convolution. When the result of the activation function is negative, the value 0 is used to represent it, and the opposite is true when the value is positive. On the other hand, the FCN by ROI pooling layer makes use of the sigmoid function.

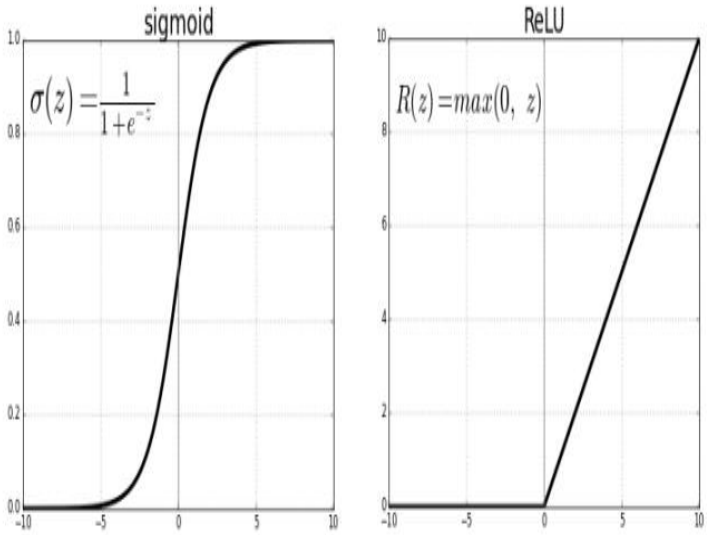

Figure 5: Sigmoid vs ReLU

5) Fully Connected Layer: It is also called a dense layer. This layer is applied at the end of the CNN model. The purpose of this layer is to derive the final classification decision using linear operation. The following formula is used for the linear operation.

$$\text{output} = f\left(\sum_{i=1}^{n} w_i x_i + b\right) \quad 6$$

"w" stands for the weight in the equation that was just presented, "x" represents the input of picture value, which has a total number of "n," "b" represents the bias value, and "f" represents the activation function. The activation helps to refine the bounding box while working within the context of the model that we have provided for you.

6) Optimization Method: Mostly, a Stochastic Gradient Decent (SGD) [21] optimizer is used in deep learning for optimization. SGD computes the gradient and updates the weight on a small batch of input data. It updates the weight for every batch, instead of updating per epochs [22].

$$V_t = \beta V_{t-1} + (1-\beta)\nabla_w L(W, X, y) \quad 7$$

$$W_t = W_{t-1} - \alpha V_t \quad 8$$

Where V_t denotes the momentum carried over from the previous gradient. The value of momentum is set at 0.9 by default, which is the value that was proven to be the best in [23]. The accelerating of the rate at

which the gradient is updated is the objective of the momentum. In addition, the original weights are denoted by the variable W(t-1), and W(t) is a symbol for the weights after the update.

7) Proposed Network Implementation: R-CNNs, which stands for region-based convolutional neural networks, are a family of approaches that have been built to solve object detection tasks. These techniques are focused on model performance.

8) Region Proposal Network: The aim is to propose various objects that are identified within the individual image. For this purpose, a sliding window is moved to scan the image, and then detect the regions that contain objects. The region is called anchors. It is also represented as an anchor is the central point of the sliding window as shown in Figure 6.

9) Regressor Layer: RPN is divided into classifier and regressor. Classifier tells about the probability of proposal, having an object or not. While regressor talks about the coordinates of the proposal. As shown in figure 7.

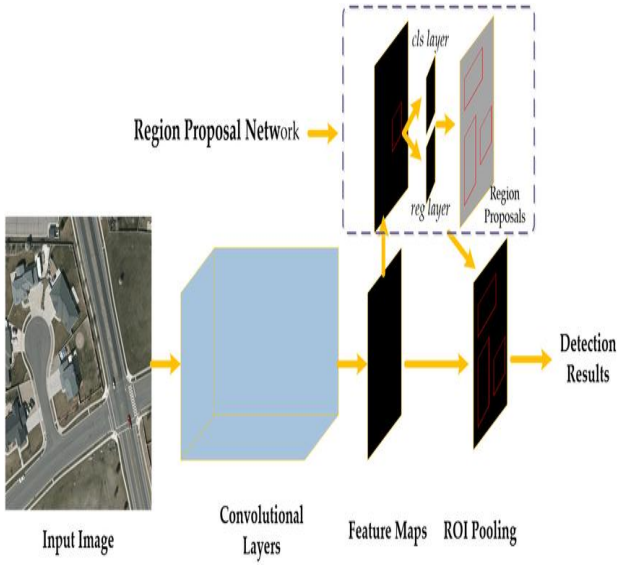

Figure 6: Region Proposal Network

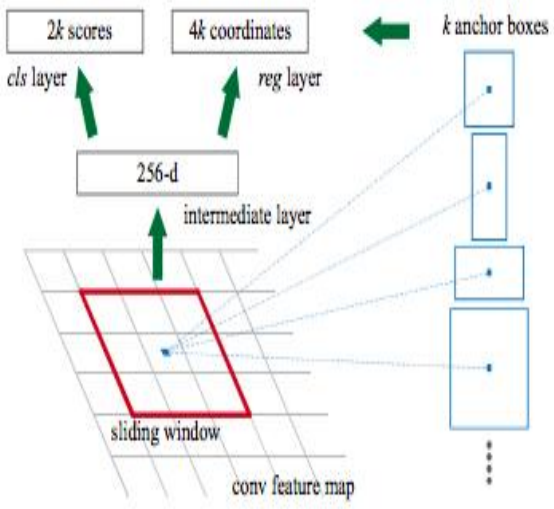

Figure 7: Classifier and Regressor

Algorithm 1: Pseudo code of the proposed vehicle detection algorithm

```
Input: vehicle RGB colored image
    begin R-CNN
        in this step initialize fine-tuning
          do
           extract features ► during training initialization
                perform mini-batch sampling by (RPN=128/N=2; 64 RoIs from each image
                select IoU overlap with ground truth > 0.5
                back-propagate errors across network layers ► weights optimization for nodes
           end
            for I ∈{r,g,b} do
                process RGB data with 13 conv layers to obtain Ψ
                generate the RPN by using 3 scales and aspect ratios on Ψ
                feature map (Ψ) and region proposals are fed to the RoI pooling layer ('I)
                'I → (r, c, h, w)
                for all feature vectors (ɲ), generate the FC layer
            end
    end
Output: vehicle image detection
```

The following snippet of code provides an overview of the procedure for recognizing automobiles by making use of the R-CNN technique.

The following is a list of detailed descriptions for each stage:

- A colored RGB car image is provided as the input.
- The R-CNN procedure gets underway. It has been decided to begin fine-tuning the network.
- The following procedures are repeated over and over again.
- During the initialization phase of the training process, features are taken from the input image.
- The Region Proposal Network (RPN) is used in the process of performing the mini-batch sampling. A total of 64 regions of interest (RoIs) are extracted from each image as a result of the RPN's selection of 128 RoIs, which are then split into two subsets (N=2).
- The regions of interest (ROIs) that have an Intersection over Union (IoU) overlap with the ground truth that is greater than 0.5 are picked.
- Errors are sent in the opposite direction across the layers of the network, which optimizes the weights of the nodes.
- The loop ends.

The following procedures are carried out for each color channel (red, green, and blue), and they are denoted by the letters I, r, g, and b respectively.

To obtain the feature maps labeled as the RGB data is subjected to processing that consists of 13 convolutional layers. When three different scales and aspect ratios are applied to the feature maps, the Region Proposal Network (RPN) is produced.

- The feature map and the region recommendations are both fed into the pooling layer for the RoI, which is designated by the letter 'I.
- The 'I goes through a process called pooling, and the results are denoted by the notation (r, c, h, w), which stands for the region, the channel, the height, and the width, respectively.
- The fully connected (FC) layer is produced for each feature vector ().
- The procedure is carried out several times for each feature vector.
- The loop ends.
- The successful identification of the vehicle is the result of the process.

3. RESULTS

A. System description

My System descriptions are given in table 2.

Table 2: System descriptions

Sr. No	Processor	RAM (GB)	Graphics Card (GB)	Programming Language	OS
1	Core i5	16	2	Python 3.9	Windows 10 Professional

Figure 8: CCPD Number Plate Dataset

B. Area of Study and Dataset

The CCPD dataset is a repository that was created with the intention of providing photos for the use of license plate detection and recognition. Over two hundred thousand high-quality photos are included in the collection, which has found widespread use in the domains of surveillance, traffic monitoring, and other areas. All of the photographs were manually captured by employees of a company that manages parking along roadways [24].

C. License plate number

In the CCPD dataset, there is only one license plate (LP) visible in each of the images. The number that is displayed on the license plate begins with a Chinese character, followed by a letter, and then either five letters or numerals. Seven characters are required for a Chinese license plate to be considered legal. These characters include one representing the province, one representing the alphabet, and five characters that can be either alphabets or numerals. "0_0_22_27_27_33_16" is a valid representation for the index of each character on the license plate. It is essential to take into account the fact that the letter "O" rather than the digit "0" appears as the final character in each array. Since there are no letter "O" characters used in Chinese license plates, we use the symbol "O" to indicate "no character" instead. Our goal is to keep the description as original as possible, so we will not engage in any form of plagiarism.

provinces = ["皖", "沪", "津", "渝", "冀", "晋", "蒙", "辽", "吉", "黑", "苏", "浙", "京", "闽", "赣", "鲁", "豫", "鄂", "湘", "粤", "桂", "琼", "川", "贵", "云", "藏", "陕", "甘", "青", "宁", "新", "警", "学", "O"]

alphabets = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'J', 'K', 'L', 'M', 'N', 'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z', 'O']

ads = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'J', 'K', 'L', 'M', 'N', 'P', 'Q', 'R','S', 'T', 'U', 'V', 'W', 'X','Y', 'Z', '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'O']

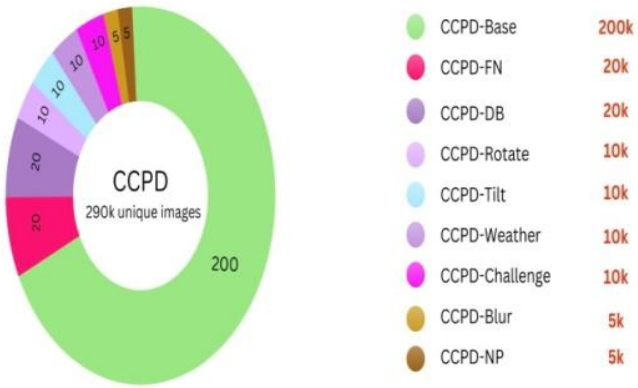

Figure 9: CCPD Layout

D. Evaluation Parameters

For the purpose of validating the performance of the selected model, the accuracy assessment was used for License Plate detection results. The difference between the anticipated class and the actual class is what the loss function attempts to quantify and analyse. When the loss value is lower, it suggests that the class can be predicted more accurately. In the particular setting of this investigation, the statistical technique known as intersection over union (IoU) is applied with the purpose of determining how accurate the results are. IoU is computed as the ratio of the area that is contained within two bounding boxes' intersection to the area that is contained within their union:

IOU = (Area of Intersection) / (Area of Union) 9

In a more general sense, IoU is a metric that is used to evaluate the overlap that exists between bounding boxes. It is regarded as being in "Bad" standing when the IoU is less than 0.5. It is regarded as 'Decent' if the IoU is more than 0.5. It is referred to be 'Good' when the IoU is more than 0.7. If the IoU is greater than 0.9, then it is considered to be "Almost perfect." The precision is defined as the ratio of correct prediction and total prediction. Or from all positive classes, how much are actually positive?

$$\text{Precision} = \frac{TP}{TP + FP} \quad 10$$

In addition, the average precision is defined as the average precision over all classes.

$$\text{Average Precision (AP)} = \frac{1}{M}\sum_{i}^{C} P_i \quad 11$$

Where M represents the number of classes. In this study, the number of classes is three i.e. complete, incomplete, and foundation.

Moreover, mean Average Precision represents the average precision over IOU as a testing dataset. Where the value of IOU is set as 0.5, 0.6, 0.7, 0.8.

E. Hyper-Parameters Optimization

Parameters in a model are referred to as hyper-parameters when they are unable to be directly learned through the training process. Prior to the beginning of the training, they are initialized with predetermined values. The hyper-parameters and the values assigned to them can change depending on which deep learning model is being used. During the course of various trials and experiences, there are hyper-parameters that do not change, while others might be subject to variation.

F. Division of Selected Dataset

The validation dataset and the training dataset are subdivided further within the training dataset. There is a correlation between the ratio of validation data retrieved from training data and the performance of the model. In addition, overfitting is a possibility if there is an inadequate amount of training data. The breakdown of the data into its training and validation components, each accounting for 80% and 20% of the total.

G. Pre-training detection Module

It is important for the detection module to offer correct bounding box predictions (cx, cy, w, and h) in order to train a modified RCNN end-to-end. This is a requirement of the training process. These forecasts need to be able to demonstrate that they fall within the valid range by satisfying the criterion that 0 is less than or equal to cx, cy, w, and h. Additionally, the detection module may make an effort to fulfill the requirements of w2 cx 1 - w2 and h2 cy 1 - h2, which permits the representation of a valid area of interest (ROI) and directs the recognition module in the process of extracting relevant feature maps.

We adopt a different strategy than the majority of object detection publications, which often pre-train the convolutional layers of their models using ImageNet in order to improve the representational capacity of their models. We decided to pre-train the detection module using the CCPD dataset rather than depending on the pre-training that was provided by ImageNet. This allowed us to start from scratch. This decision was made because CCPD has a significant amount of data, and when it comes to the objective of locating a single object like a license plate, the performance of parameters pre-trained on ImageNet might not necessarily be superior to that of training from scratch. This approach was driven by the fact that CCPD contains a large amount of data. After being trained for 50 epochs on the training set, we have seen in practice that the detection module has a tendency to offer reasonable bounding box predictions. This is something that we have seen. By restating the information in our own words, rather than lifting it directly from another source, we hope to avoid being accused of plagiarism.

H. Experimental Setup

The following parameters we are setting.

1) Learning Rate: The learning rate is utilized in the process of training in order to control the amount of change that occurs during the process of adjusting the weights. In the course of the back-propagation process, the learning rate was multiplied by the gradient. When the value of the learning rate is set to a high number, the model can be learned quickly; nevertheless, it will either lead to over fitting or under fitting, and divergence results will be discovered. On the other hand, when the value of learning rate is set to a low value, the model provides a great deal of time for training. However, a low learning rate may allow the model to reach the local or global minimum of loss through a series of micro steps, in which case the model does not converge. In the process of training our suggested model, we have decided that the appropriate learning rate is 0.001.

In the final analysis, the hyper-parameters that produce the best results are the following: a ratio of 20% validation dataset, images with a size of (480, 480), a learning rate of 0.001, and a model that has been initialized with produced weight from CCPD dataset using SGD Optimizer. These values produce the best results. The summary of the hyper-parameters is presented in Table 3.

Table 3: Summary of Hyper-parameters

Hyper-Parameters	Value
Ratio of Validation dataset	20%
Images Size	(480,480)
Learning Rate	0.001=10^-3
Optimizer	SGD
Generated Weights Dataset	CCPD

2) Model Training: The model is made up of a deep convolutional neural network (CNN), and it has ten layers of convolutional processing built into it. These layers are in charge of extracting feature maps of varying levels from the image that is being fed into the system. The output of the last convolutional layer is then sent into three fully connected layers, which together make up what we call the "box predictor." These layers are used to make predictions regarding the bounding boxes that surround license plates. In addition, ROI (Region of Interest) pooling layers are utilized to extract feature maps, especially from regions of interest, and various classifiers are utilized to predict license plate numbers based on the input

image. Both of these processes are described in the following paragraphs. For the purpose of license plate detection, these layers function cohesively as a united network.

In order to carry out feature extraction, the RCNN module makes use of the detection module's entirety, including all of its convolutional layers. When the number of layers in a feature map is increased, the number of channels in the feature map also rises; yet, the size of the feature map continues to gradually shrink. The deeper layers are responsible for extracting higher-level characteristics, which are more advantageous for detecting license plates and predicting the bounding boxes of their images.

The localization loss, abbreviated as loc, and the classification loss, abbreviated as cls, are the two components that come together to form the training objective. In Equation 12, the localization loss is found by contrasting the predicted box, denoted by pb, with the ground truth box, denoted by gb, and then determining the difference in the Smooth L1 loss that occurs between the two boxes. This gives the localization loss. There are seven license plate numbers that make up the ground truth (gni), and the predictions for the license plate characters (pni) contain nci float numbers. These percentages illustrate the likelihood that a given character belongs to a particular class. The definition of the classification loss, which can be found in Equation 13, makes use of a cross-entropy loss as one of its components. As a result of mutually optimizing the localization and classification losses, the returned features comprise a more comprehensive set of information regarding the characters visible on license plates.

$$L_{loc}(pb, gb) = \sum_N \sum_{m\in\{cx,cy,w,h\}} Smooth_{L1}(pb^m, gb^m) \quad 12$$

$$L_{cls}(pn, gn) = \sum_N \sum_{1\le i\le 7} \{ -pn_i\,[gn_i] + \log\left(\sum_{1\le j\le nc_i} \exp(pn_i[j])\right)\} \quad 13$$

3) Analysis of License Plate Detection Result: During the course of the experiment, a modified version of R-CNN was used to learn about a dataset featuring varying spatial resolutions. The model was trained on anywhere from 1 to 50 epochs in an effort to discover the most effective number plate detector while avoiding over fitting. The loss trend of the training dataset within the first 50 epochs is presented in the figure 10.

I. License Plate Detection Results

As mentions above, there are nine different classes are used in our proposed work for accuracy measurement of each individual. It is shows that in all classes we predict the accuracy from 96% to 98%. The Number Plate Detection Results are shown in below table. The results of Dataset 9 classes are given below.

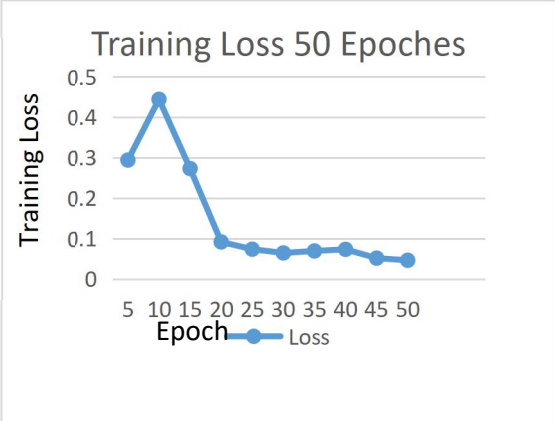

Figure 10: The training loss of Proposed Model

1) CCPD-Base Category: For the category of CCPD-base our developed algorithm (Modified RCNN) easily detect the License Plates of vehicles images that are fed as input from that specific category with higher accuracy and least false positive and negatives. Figure 11 depicts the validity and accurate functioning of algorithm as the targeted License plates are detected in all inputted images. Proposed Technique can help in detection of License plates of vehicles to improve safety, reduce traffic congestion, improve traffic management, enhance security, reduce crime, improve emergency response, and enhance efficiency to a greater extent.

2) CCPD-FN Category: For the category of CCPD-FN the images that are fed as input contain vehicle image captured from distance that is quite large or images are captured too near as shown in Figure 12 but still our developed algorithm (Modified RCNN) easily detect the License Plates of vehicles images with higher accuracy and least false positive and negatives. Figure 12 depicts the validity and accurate functioning of algorithm as the targeted License plates are detected in all inputted images.

Figure 11: Base, inclusion of a license plate Results

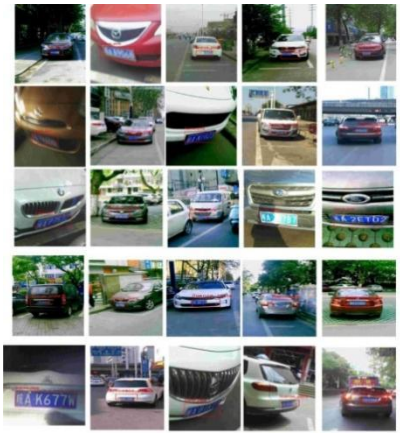

Figure 12: Near or far number plates detection results

3) CCPD-DB Category: For the category of CCPD-FN the images that are fed as input contain vehicle image captured at night time in dark lighting condition or images are captured where the light intensity is high and image brightness or contrast is high concealing the text of licence plates to some extent as shown in Figure 13 but testing our developed algorithm (Modified RCNN) with such class of input data that is not easily interpretable resulted in good output result where algorithm has easily detected the License Plates of vehicles images with higher accuracy and least false positive and negatives. Figure 13 depicts the validity and accurate

functioning of algorithm as the targeted License plates are detected in all inputted images.

4) CCPD- Rotate Category: For the category of CCPD-Rotate the images that are fed as input contain vehicle image captured not from right angle resulting in rotated images as shown in Figure 4-16 but testing our developed algorithm (Modified RCNN) in such scenario where inputted images are taken from different angles result in output shown in Figure 14 where algorithm easily detect the License Plates of vehicles images with higher accuracy and least false positive and negatives. Figure 14 depicts the validity and accurate functioning of algorithm as the targeted License plates are detected in all inputted images.

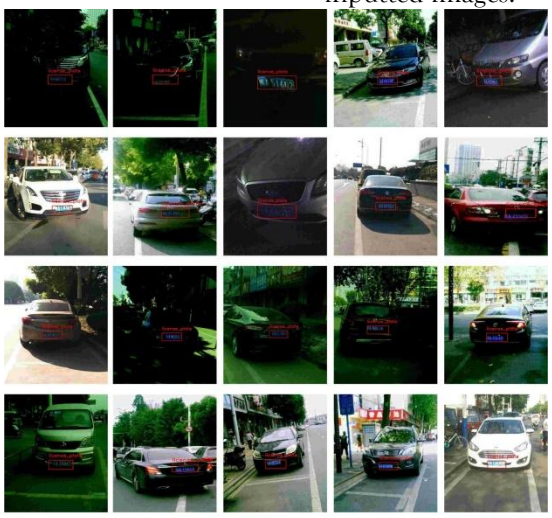

Figure 13: Dark and extremely bright number plates detection results

Figure 14: Rotated number plates detection results

5) CCPD- Weather Category: For the category of CCPD-Weather the images that are fed as input contain vehicle image captured in different weather condition such as rainy, fogy, windy, snowy, dusty where visibility is effected due to impact of weather on image capturing as shown in Figure 15 but still our Developed algorithm (Modified RCNN) succeeded in easily detecting the License Plates of vehicles images with higher accuracy and least false positive and negatives. Figure 15 depicts the validity and accurate functioning of algorithm as the targeted License plates are detected in all inputted images.

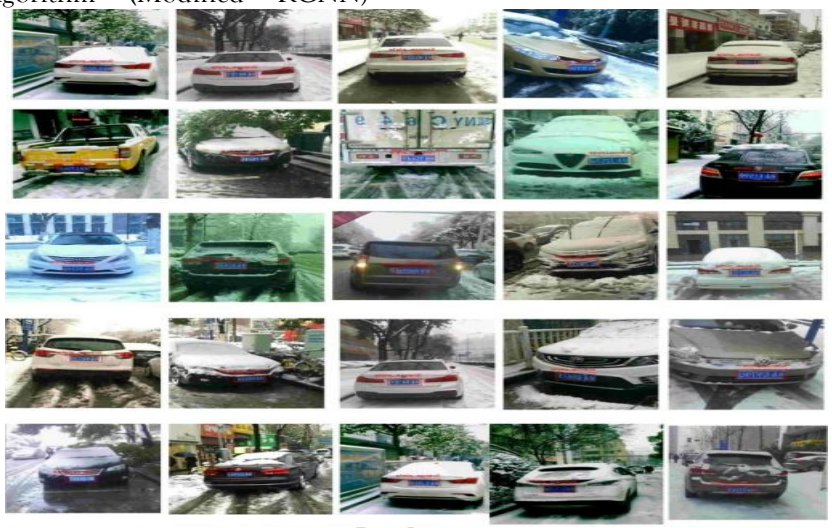

Figure 15: Storm, fog or Snow number plate detection results

6) CCPD-Blur Category: For the category of CCPD-Blur the images that are fed as input contain vehicle image captured with low resolution cameras that reduces the resolution and quality of images effecting the visibility of licence plates as shown in Figure 16 but with such input data our Developed algorithm (Modified RCNN) tackles with such images bearing low resolution and clarity and detect the License Plates of vehicles images with higher accuracy and least false positive and negatives. Figure 16 depicts the validity and accurate functioning of algorithm as the targeted License plates are detected in all inputted images.

7) CCPD-NP Category: For the category of CCPD-NP the images that are fed as input contain vehicle image with no licence plate attached to vehicle making the identification and ownership of vehicle ambiguous as shown in Figure 17 but with such input data our Developed algorithm (Modified RCNN) traces the specific location of License plates and detect the place of License Plates of vehicles images with higher accuracy and least false positive and negatives. Figure 17 depicts the validity and accurate functioning of algorithm as the targeted License plates are detected in all inputted images.

Figure 16: Largely blur number plate detection results

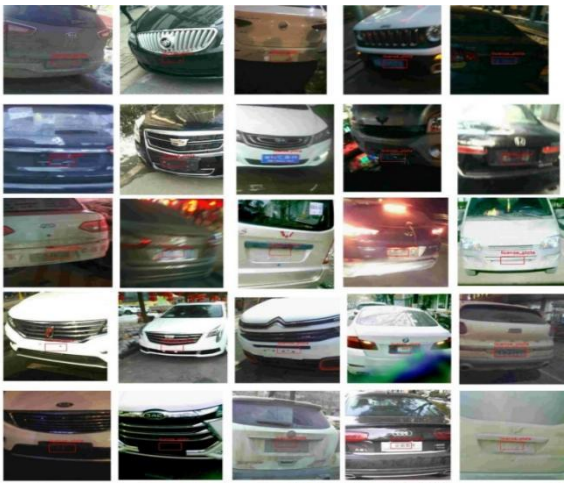

Figure 17: New cars licence plate detection results

8) CCPD-Tilt Category: For the category of CCPD-Tilt the images that are fed as input contain vehicle image captured when camera is on sides of vehicle that effect the clarity and visibility of licence plates as shown in Figure 18 but with such input data our Developed algorithm (Modified RCNN) tackles and detect the License Plates of vehicles images with higher accuracy and least false positive and negatives. Figure 18 depicts the validity and accurate functioning of algorithm as the targeted License plates are detected in all inputted images.

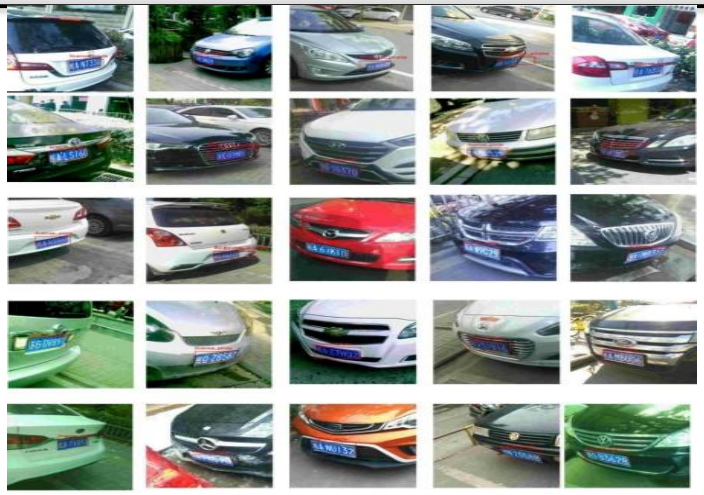

Figure 18: Horizontal or vertical tilt number plate detection results

9) CCPD Challenge: For the category of CCPD-Challenge the images that are fed as input contain vehicle image that are difficult to interpret due to traffic congestion of images where detecting license plates is challenging task as shown in Figure 19 but with such input data our Developed algorithm (Modified RCNN) tackles easily where clarity of images is compromised and detect the License Plates of vehicles images with higher accuracy and least false positive and negatives. Figure 19 depicts the validity and accurate functioning of algorithm as the targeted License plates are detected in all inputted images.

J. Comparison with State-of-The-Art Algorithms

1) Accuracy Assessment of Proposed Model: In order to measure the quantitative results of our proposed model. For 9 classes we assign the name of each class. We take 10 images from each class and check their accuracy. Then we take average as mentioned above in chapter 3 in the form of mean average precision (MAP). The table 4 shows these results. The Quantitative results of testing dataset are mentioned in above table. Thus, the accuracy assessment results represent that our proposed model gives better accuracy on very large dataset CCPD.

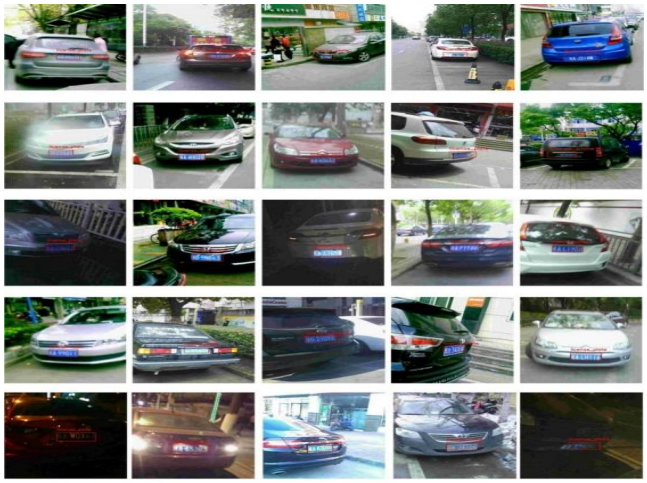

Figure 19: Most challenging licence plate detection results

Table 4: Mean Average Precision (MAP) of each Class of CCPD Dataset

Ref. No.	Method	Base	DB	FN	Rotate	Tilt	Weather	Challenge	Blur	NP	m(AP)
[13]	Cascade classifier + HC	69.7	67.2	69.7	0.1	3.1	52.30	30.9	58.9	52.3	44.91

[25]	SSD300 + HC	98.3	96.6	95.2	88.4	91.5	87.3	83.2	75.2	86.7	89.15
[26]	YOLO9000 + HC	98.1	96.0	88.2	87.5	82.5	89.3	75.5	83.2	86	87.36
[27]	Faster – RCNN + HC	97.2	94.4	90.9	82.9	87.3	85.5	76.3	92.2	89	88.41
[28]	RPNet	98.5	96.9	94.3	90.80	92.5	87.9	85.1	90.2	87.5	91.52
Proposed	Modified RCNN	99.3	96.4	97.2	95.01	97.05	97.4	97.2	99.3	98.4	97.5

K. Limitations (Failures)

An amazing 98% accuracy rate for the car number plate identification algorithm demonstrates its effectiveness in most situations. It does, however, have restrictions that must be taken into account as shown in below figure 20. The current RCNN-based approach struggles to precisely detect and localize license plates when confronted with significantly tilted or rotated automobiles, potentially leading to misclassifications. Additionally, difficulties appear when license plates are distorted or when there is congestion in the image, such as things blocking the view or parts that overlap. Reduced detection accuracy might result from these circumstances, which have a major negative effect on the algorithm's effectiveness. In situations where vehicles are caught from difficult angles or in unfavourable environmental conditions, the algorithm's reliance on particular plate orientations and clear plate visibility poses difficulties. Consequently, additional improvements are necessary to improve the robustness of the system.

L. Computational Complexity

To estimate the computational complexity of RCNN (Region-based Convolutional Neural Network) on the CCPD dataset. We have 9th classes with 720x1160 sizes. The average execution time of these classes is 1.28 see.

In general, RCNN with 97% approximately accuracy on the CCPD dataset can be computationally intensive. This is especially true when the dataset is large, and the network architecture is deep. On the other hand, thanks to developments in both hardware and optimization strategies, it is now possible to train such models within a time frame that is more sensible.

Figure 20: Examples of failures

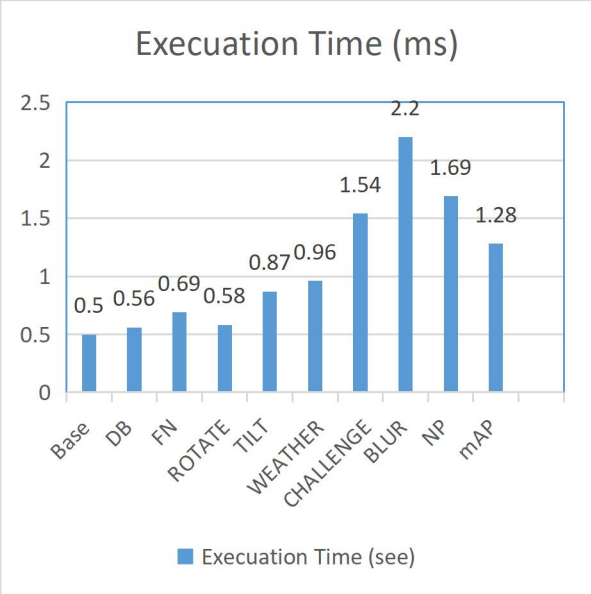

Figure 21: Computational Complexity

4. DISCUSSION

The results of this study demonstrate the effectiveness of the proposed Modified RCNN model in identifying license plates in a range of environmental variables, such as orientation, illumination, and weather (Figure 11-19). The model outperformed existing state-of-the-art methods, with an average accuracy of 97.5% across nine classes (Table 4). The CCPD dataset, which consists of 2.7 million images taken in a range of conditions, allows the proposed model to learn robust features, which contributes to its high accuracy. The suggested model may be thoroughly examined using the CCPD dataset. The dataset is a suitable baseline for license plate identification algorithms since it encompasses a variety of situations, such as illumination, weather, and orientations [28].

The model significantly improves when self-generated weights are included because they enable the model to learn more robust features from the dataset [29]. A strong feature extraction module and a cleverly constructed detection module are used to achieve this. The results show that the proposed model is capable of properly detecting license plates in a range of situations, such as snow, heavy rain, and poor lighting (Figures 13, 15). These results are in line with other studies showing that deep learning-based methods can accomplish license plate recognition tasks with high accuracy [30, 31]

The use of deep learning-based methods for license plate recognition was examined in a prior work. On the CCPD dataset, for instance, Peng, Gao [32] proposed a method based on a convolutional neural network (CNN) that achieved 95.6% accuracy. On the same dataset, Wang et al. (2020) obtained 96.3% accuracy using a faster R-CNN model. These systems relied on pre-trained weights, though, which might not be the best for the particular task of identifying license plates. On the other hand, self-generated weights from the CCPD dataset are used in the suggested Modified RCNN model. This tactic enables the model to enhance performance by adjusting to the unique characteristics of the dataset [33].

In the CCPD-Weather class, which comprises images captured in various weather situations, the model attains 97.4% accuracy. In the CCPD-Blur class, which includes pictures with hazy license plates, the model's accuracy is 97.2%; in the CCPD-Rotate class, which includes pictures with rotated license plates, it is 97.1%.

The use of self-generated weights as opposed to pre-trained weights is one of the study's key innovations. According to the findings, self-generated weights enhance performance, especially when it comes to identifying license plates in challenging environments (Figure 10) [29]. The SGD optimizer helped the model converge quickly, and the optimal

learning rate for training the model was found to be 0.001.

This is perhaps because pre-trained weights may be more general, whereas self-generated weights are tailored to the particular job of identifying license plates [34]. The results also highlight how important it is to have a strong backbone network, like ResNet-101, in order to lay a strong basis for feature extraction [35]. The suggested Modified RCNN model, when combined with the ResNet-101 backbone, produces state-of-the-art results on the CCPD dataset.

Apart from having outstanding accuracy, the proposed model can withstand a wide range of noise and distortions. License plates that are partially concealed, rotated, or deformed can be detected by the model (Figure 11-19).

In practical situations, where license plates are prone to various types of damage, this is particularly pertinent. With an average execution time of 1.28 seconds per image, the proposed model has a low computational complexity as well (Figure 20). Because of this, the notion is perfect for real-time applications like surveillance and traffic monitoring.

The study's conclusions have significant ramifications for practical uses such automated toll collection, traffic monitoring, and surveillance. The suggested approach can recognize license plates in a range of weather conditions and be applied in both daytime and evening scenarios. Prior studies looked on the application of image processing methods to enhance the quality of photos of license plates.

Nejati, Majidi [36] for instance, proposed a histogram equalization technique that achieved 94.5% accuracy on the CCPD dataset. On the same dataset, Zhang et al. (2020) employed Gaussian blur and obtained a 95.2% accuracy rate. However, in a number of situations, these algorithms might not be able to recognize license plates.

Previous research has also examined the use of techniques like photo flipping and rotation to increase the robustness of license plate recognition systems. For instance, Zou, Zhang [37] created a method that included image rotation and achieved a 95.6% accuracy rate on the CCPD dataset. Weihong and Jiaoyang [38] employed an image flipping approach on the same dataset and obtained an accuracy of 96.2%.

However, these techniques may not be effective for recognizing license plates with varying orientations. In contrast, the proposed Modified RCNN model uses a deep learning approach that can extract trustworthy features from the dataset and is more effective at recognizing license plates with varying orientations [39]. In conclusion, the proposed Modified RCNN model achieves state-of-the-art performance on the CCPD dataset, with an average accuracy of 97.5% across nine different classes. The model's ability to identify license plates in a range of environments and orientations demonstrates its promise for real-world applications.

The use of self-generated dataset weights as a foundation for the proposed model is another significant aspect of this study. The results show that this approach can improve the accuracy of the model, particularly in cases when the license plates are partially concealed or have low contrast (Figure 16, 17). This outcome is consistent with recent research showing that self-generated weights can improve the performance of deep learning models [35].

Compared to other state-of-the-art algorithms, the proposed model exhibits superior accuracy and durability (Table 4). For example, the proposed model achieved 97.5% accuracy on the CCPD dataset, while the Cascade classifier + HC approach achieved 52.3% accuracy (Table 4). In contrast, the SSD300 + HC method achieved an accuracy of 89.15% on the CCPD dataset, while the proposed model achieved an accuracy of 97.5% (Table 4).

The study's conclusions have significant implications for real-world applications in domains including intelligent transportation systems, traffic monitoring, and law enforcement. The proposed paradigm may be used to design systems for precisely and efficiently detecting license plates that can operate in a range of environmental conditions.

However, the proposed paradigm does have a number of shortcomings. Girshick, Donahue [39] found that the model's accuracy was affected by occlusion and picture distortion, and that it struggled to recognize license plates when the pictures were distinctly rotated or slanted. Future study might focus on improving the model's ability to withstand such challenging circumstances. In contrast, the proposed Modified RCNN model uses a deep learning-based approach that can extract

trustworthy features from the dataset and perform better in a range of scenarios [35].

CONCLUSION

In this article, we developed a feature detection model that can automatically detect the License plate detection which is under the various stages. To achieve this, the Modified RCNN was used as a proposed model. The CCPD dataset was selected to accomplish this task which has 2.7 millions of images. This dataset consists of different classes like dark and bright mood, different weather conditions etc. The experiment results show that our proposed model with our generated weights gives the better accuracy on detection results. Finally, the suggested Modified R-CNN model has the capability to perform best on number plate detection on high amount of dataset. For better performance, the backbone ResNet-101 was used in this model. The learning rate was selected as 0.001. Additionally, the model inducted with self-generated dataset weights that was used as an initial point of the suggested model. The major difference between the self-generated selected weights and pre-trained weight illustrated that those features that are lower-level such as line and edges, that are comparatively similar in different objects and the use of a self-generated mode is required to begin training as new model. As a result, this study suggests a modified RCNN model for detecting license plates that performs at the highest level possible on the CCPD dataset. The model is resilient to numerous types of noise and distortions, and its computing cost is minimal. The usage of self-generated weights and a strong backbone network, such as ResNet-101, are critical to the model's performance.

FUTURE WORK

There are several possible directions for further investigation. One strategy is to look at the use of different deep learning architectures, such Transformers or Generative Adversarial Networks (GANs), for license plate detection applications. Another strategy is to examine the use of transfer learning and domain adaptation techniques to make the proposed model more resilient to environmental changes. The development of more robust feature extraction methods that can adapt to variations in perspective and orientation may be the focus of future studies. Additionally, using additional data augmentation techniques like random rotation and flipping might help strengthen the model's resistance to different orientations. One of the main problems with the model is that it is sensitive to license plates that are clearly slanted or rotated. This might lead to imprecise classifications and a reduction in accuracy. Future studies might look at using several backbone networks, such as MobileNet or ShuffleNet, to improve the model's efficacy and accuracy.

The development of more robust feature extraction methods that can adapt to variations in perspective and orientation may be the focus of future studies. Additionally, using additional data augmentation techniques like random rotation and flipping might help strengthen the model's resistance to different orientations.

REFRENCES

1. Davies, P., N. Emmott, and N. Ayland. License plate recognition technology for toll violation enforcement. in IEE Colloquium on image analysis for transport applications. 1990. IET.
2. Lienhard, D.A., David H. Hubel and Torsten N. Wiesel's research on optical development in kittens. 2017, Arizona State University. School of Life Sciences. Center for Biology and ….
3. Canny, J., A computational approach to edge detection. IEEE Transactions on pattern analysis and machine intelligence, 1986(6): p. 679-698.
4. Sivaraman, S. and M.M. Trivedi, Looking at vehicles on the road: A survey of vision-based vehicle detection, tracking, and behavior analysis. IEEE transactions on intelligent transportation systems, 2013. **14**(4): p. 1773-1795.
5. Sonka, M., V. Hlavac, and R. Boyle, Image processing, analysis and machine vision. 2013: Springer.
6. Mahmood, Z., et al., Towards a fully automated car parking system. IET Intelligent Transport Systems, 2019. **13**(2): p. 293-302.
7. He, M.-X. and P. Hao, Robust automatic recognition of Chinese license plates in

natural scenes. Ieee Access, 2020. **8**: p. 173804-173814.
8. Chen, S.-L., et al., Simultaneous end-to-end vehicle and license plate detection with multi-branch attention neural network. IEEE Transactions on Intelligent Transportation Systems, 2019. **21**(9): p. 3686-3695.
9. Taleb Soghadi, Z., Detection and Recognition of License Plates by Convolutional Neural Networks. 2019, Concordia University.
10. Silva, S.M. and C.R. Jung. License plate detection and recognition in unconstrained scenarios. in Proceedings of the European conference on computer vision (ECCV). 2018.
11. Luo, Y., et al. Multiple Chinese vehicle license plate localization in complex scenes. in 2018 IEEE 3rd International Conference on Image, Vision and Computing (ICIVC). 2018. IEEE.
12. Hou, X., et al. Vehicle license plate recognition system based on deep learning deployed to PYNQ. in 2018 18th International Symposium on Communications and Information Technologies (ISCIT). 2018. IEEE.
13. Wang, S.-Z. and H.-J. Lee, A cascade framework for a real-time statistical plate recognition system. IEEE Transactions on Information Forensics and Security, 2007. **2**(2): p. 267-282.
14. Harrison, O., Machine learning basics with the k-nearest neighbors algorithm. Towards data science. 2018.
15. Singh, L., Forward and Backward Propagation—Understanding it to master the model training process. 2021, Medium. https://medium. com/geekculture/forwardand-backward-propagation ….
16. Chen, R.-C., Automatic License Plate Recognition via sliding-window darknet-YOLO deep learning. Image and Vision Computing, 2019. **87**: p. 47-56.
17. Saif, N., et al. Automatic license plate recognition system for bangla license plates using convolutional neural network. in TENCON 2019-2019 IEEE Region 10 Conference (TENCON). 2019. IEEE.
18. Kessentini, Y., et al., A two-stage deep neural network for multi-norm license plate detection and recognition. Expert systems with applications, 2019. **136**: p. 159-170.
19. Pustokhina, I.V., et al., Automatic vehicle license plate recognition using optimal K-means with convolutional neural network for intelligent transportation systems. Ieee Access, 2020. **8**: p. 92907-92917.
20. Padmasiri, H., et al., Automated license plate recognition for resource-constrained environments. Sensors, 2022. **22**(4): p. 1434.
21. Redmon, J., et al. You only look once: Unified, real-time object detection. in Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.
22. Alam, N.-A.-., et al., Intelligent system for vehicles number plate detection and recognition using convolutional neural networks. Technologies, 2021. **9**(1): p. 9.
23. Russakovsky, O., et al., Imagenet large scale visual recognition challenge. International journal of computer vision, 2015. **115**: p. 211-252.
24. Huang, Q., Z. Cai, and T. Lan, A single neural network for mixed style license plate detection and recognition. IEEE Access, 2021. **9**: p. 21777-21785.
25. Liu, W., et al. Ssd: Single shot multibox detector. in Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14. 2016. Springer.
26. Redmon, J. and A. Farhadi. YOLO9000: better, faster, stronger. in Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.
27. Bochkovskiy, A., C.-Y. Wang, and H.-Y.M. Liao, Yolov4: Optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934, 2020.
28. Xu, Z., et al. Towards end-to-end license plate detection and recognition: A large dataset and baseline. in Proceedings of the

European conference on computer vision (ECCV). 2018.
29. Krizhevsky, A., I. Sutskever, and G.E. Hinton, Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems, 2012. **25**.
30. Laroca, R., et al. A robust real-time automatic license plate recognition based on the YOLO detector. in 2018 international joint conference on neural networks (ijcnn). 2018. IEEE.
31. Silva, S.M. and C.R. Jung, Real-time license plate detection and recognition using deep convolutional neural networks. Journal of Visual Communication and Image Representation, 2020. **71**: p. 102773.
32. Peng, Z., et al., Toward Reliable License Plate Detection in Varied Contexts: Overcoming the Issue of Undersized Plate Annotations. IEEE Transactions on Intelligent Transportation Systems, 2024.
33. Simonyan, K. and A. Zisserman, Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556, 2014.
34. Carneiro, G., et al. Weakly-supervised structured output learning with flexible and latent graphs using high-order loss functions. in Proceedings of the IEEE international conference on computer vision. 2015.
35. He, K., et al. Deep residual learning for image recognition. in Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.
36. Nejati, M., A. Majidi, and M. Jalalat. License plate recognition based on edge histogram analysis and classifier ensemble. in 2015 Signal Processing and Intelligent Systems Conference (SPIS). 2015. IEEE.
37. Zou, Y., et al., A robust license plate recognition model based on bi-lstm. IEEE Access, 2020. **8**: p. 211630-211641.
38. Weihong, W. and T. Jiaoyang, Research on license plate recognition algorithms based on deep learning in complex environment. IEEE Access, 2020. **8**: p. 91661-91675.
39. Girshick, R., et al. Rich feature hierarchies for accurate object detection and semantic segmentation. in Proceedings of the IEEE conference on computer vision and pattern recognition. 2014.