



# License plate identification and recognition in a non-standard environment using neural pattern matching

Imran Shafi<sup>1</sup> · Imtiaz Hussain<sup>2</sup> · Jamil Ahmad<sup>3</sup> · Pyoung Won Kim<sup>4</sup> · Gyu Sang Choi<sup>5</sup> · Imran Ashraf<sup>5</sup> · Sadia Din<sup>5</sup>

Received: 2 April 2021 / Accepted: 31 May 2021 / Published online: 10 June 2021  
© The Author(s) 2021

## Abstract

Non-standard license plates are a part of current traffic trends in Pakistan. Private number plates should be recognized and, monitored for several purposes including security as well as a well-developed traffic system. There is a challenging task for the authorities to recognize and trace the locations for the certain number plate vehicle. In a developing country like Pakistan, it is tough to have higher constraints on the efficiency of any license plate identification and recognition algorithm. Character recognition efficiency should be a route map for the achievement of the desired results within the specified constraints. The main goal of this study is to devise a robust detection and recognition mechanism for non-standard, transitional vehicle license plates generally found in developing countries. Improvement in the character recognition efficiency of drawn and printed plates in different styles and fonts using single using multiple state-of-the-art technologies including machine-learning (ML) models. For the mentioned study, 53-layer deep convolutional neural network (CNN) architecture based on the latest variant of object detection algorithm-You Only Look Once (YOLOv3) is employed. The proposed approach can learn the rich feature representations from the data of diversified license plates. The input image is first pre-processed for quality improvement, followed by dividing it into suitable-sized grid cells to find the correct location of the license plate. For training the CNN, license plate characters are segmented. Lastly, the results are post-processed and the accuracy of the proposed model is determined through standard benchmarks. The proposed method is successfully tested on a large image dataset consisting of eight different types of license plates from different provinces in Pakistan. The proposed system is expected to play an important role in implementing vehicle tracking, payment for parking fees, detection of vehicle over-speed limits, reducing road accidents, and identification of unauthorized vehicles. The outcome shows that the proposed approach achieves a plate detection accuracy of 97.82% and the character recognition accuracy of 96%.

**Keywords** Big data · Intelligent systems · Vehicle license plate recognition system · Intelligent transportation system · Convolutional neural network

✉ Imran Ashraf  
imranashraf@ynu.ac.kr

✉ Sadia Din  
sadiadin@yu.ac.kr

Imran Shafi  
imranshafi@ceme.nust.edu.pk

Imtiaz Hussain  
imtiazhussain901@gmail.com

Jamil Ahmad  
jamil@ieee.org

Pyoung Won Kim  
pwkim@inu.ac.kr

Gyu Sang Choi  
castchoi@ynu.ac.kr

<sup>1</sup> College of Electrical and Mechanical Engineering, National University of Sciences and Technology, Islamabad, Pakistan

<sup>2</sup> Abasyn University Islamabad Campus, Islamabad, Pakistan

<sup>3</sup> Hazara University Mansehra, Dhodial, Pakistan

<sup>4</sup> College of Education, Incheon National University, Incheon, Korea

<sup>5</sup> Department of Information and Communication Engineering, Yeungnam University, Gyeongsan, Korea

## Introduction

With the increase in the number of vehicles and the deployment of traffic monitoring cameras, license plate identification has become a topic of great interest. License plate recognition helps to identify vehicles and assists in vehicle tracking and activities analysis for surveillance and security purposes. As the demand for traffic management and safe vehicle monitoring systems continues to grow, vehicle handling has become a potential research challenge. Predominantly, it is done manually by a human agent which is laborious, time-consuming, and erroneous. On the other hand, an automated system can identify vehicle license plates efficiently, in real-time [1]. It is very difficult to check all vehicles on the road through a manual approach. Traffic control and owner identification are big problems globally and are actively researched for developing intelligent transport systems (ITS). These systems will provide the required innovative services in the modes of transport and traffic management. ITS will enable its users to have smart coordination and safe transport networking along with intelligent system implementation.

The vehicle license plate recognition (VLPR) system detects the name plate details of the vehicles through the installed cameras at different points on roads. It also plays an important role in the ITS which has several practical applications like vehicle tracking, payment for parking fees, detection of over-speed vehicles, controlling road accidents, and identification/tracking of illegal/counterfeit vehicles. However, several challenges hinder the performance of automatic license plate recognition (ALPR) systems. For example, several approaches are limited by the changes in the illumination. Although some approaches focus on the developing systems that can work both during the day and night time [2,3], yet the performance is not the same during the day and night. Many approaches tend to be computationally complex and inappropriate to deploy in real-world scenarios [4,5]. Real-world scenarios involve occlusions, rotations, and complex scenarios involving traffic jams, as shown in Fig. 1, which degrade the performance of most of the existing approaches.

Similarly, many existing approaches show up to the mark accuracy and efficiency for indoor stationary. Along with that there is involvement of motion at a lesser pace involving different scenarios but these situations do not perform well when used outdoor moving vehicles with varying speed [6,7]. Recent work has been done in this field of number plate recognition along with number detection methods with the advanced tools and analysis. Automatic Number Plate Recognition Using Convolution Neural Network has been researched for the detection of moving car speed as well as the judgment of the red-light violation on the traffic routes [8]. Along with this stream of research, another major developed has been done in the Character Segmentation Method



Fig. 1 A cluttered traffic jam of vehicles in Pakistan



Fig. 2 A number plate shop picture

to Increase Character Recognition Accuracy for Turkish Number Plates that will be used for accuracy of the recognition of number plate sensing and detection [9]. Additional challenges are hampering the success of such systems in developing countries like the use of multiple types of license plates, hanging the plates at different heights, different colors and sizes of the plates, and the use of non-standard fonts and templates. An illustration of such problems is shown in Fig. 2.

Single-stage deep-learning-based ALPR systems provide much more stability to the system analysis that will carry out the whole mechanism towards the recognition studies. Along with that good accuracy should be considered for the corresponding plate number however, often involve a high computational cost that is not affordable for real-time operations without edge computing [10]. On the other hand, multi-stage approaches can be deployed easily for real-time operations due to their less computational complexity, with moderate accuracy. Therefore, this study presents a multi-stage license plate recognition approach for non-conventional plates found in developing countries like Pakistan. This study makes the following contributions in brief

- This study presents a deep-learning-based license plate detection and recognition approach. In the initial stage, the location of the plate is found to detect the plate and then the detected plate area is further processed to identify the contents of the plate.

- For plate recognition, a 53-layer deep CNN architecture is adopted which is based on the latest object detection algorithm-YOLOv3 which is abbreviated as "You Only Look Once, Version 3", this is based on the algorithm of real-time object detection. It is used to identify specific objects in videos, live streams, or images. CNN structure is used based on deep learning to detect number plates with high accuracy and robust output.
- Extensive experiments are carried out with different approaches including YOLOv3, ResNet-50, YOLOv2, Darknet-19, SSD-300, and SSD-7 to investigate the performance of the proposed approach.

The rest of the paper has been organized in the following manner. The next section describes research works from the literature which are closely linked to the current study. The proposed methodology and its working mechanism are discussed in the third section. The fourth section discusses the experimental results while in the end, the concluding remarks are given in the fifth section.

## Literature review

The wide use of personal vehicles introduced several challenges during the last two decades including efficient traffic control, tracking traffic rules violating vehicles, finding theft cars, and security surveillance. These tasks require the detection and recognition of the license plate of the vehicles. Consequently, a large body of works can be found in the literature that focuses on license plate identification.

The authors divide the task of license plate recognition into red, green, and blue channels for segmentation in [11]. The image in each channel is used as inputs for CNN to learn the hierarchical features, then the results are passed to SVM for generating the target probability label values. Similarly, CNN-based strategies are used by [12] for a multi-directional license plate detection system using the multi-directional you only look once (MD-YOLO) framework based on CNN. The proposed framework can process multi-directional problems with the best detection accuracy. MD-YOLO predicts the center coordinate of every object with width and height. The informative system is trained by the authors in [13] which is based on a deep-learning approach divided into three parts: detection, cutting, and role recognition. The detection of the license plates has been done in many pre-treatment steps and after that the application of CNN model has been implemented. After this, finally, all the shapes (A–Z) and digits (0–9) are identified. A special kind of feed-forward multi-layer perception is discussed by [14]. The proposed supervised model can perform automatic feature extraction to perform license recognition. The main purpose of the

implemented method to increase the performance of character recognition using the customized CNN.

License plate recognition tasks vary with the change in the characters used in different languages, so various approaches are presented accordingly. For example, the authors present an effective recognition system for Malaysian license plates in [15]. The license plate recognition can be performed in both one line and double lines using a smearing algorithm. Plate localization accuracy of 97.4%, character segmentation accuracy of 96%, and the character recognition accuracy 76% are achieved. The authors addressed the problem of license plate detection using the inception method in [16]. The proposed approach aims at identifying license plates from dull and low-intensity images. Problems related to plate identification are focused such as blob extraction, segmentation, and character recognition. The approach can provide a recognition accuracy of 98.40%.

Similarly, the authors present an approach for license plate recognition for Australian conditions in [17]. It deals with variation in the appearance of the number of plates caused using different designs and colors from different states. Besides, the plates put in the covers and the use of non-standard materials pose an extra challenge. The proposed approach identifies different plates with a 0.95 accuracy, using image processing and an artificial neural network-based approach. A license plate recognition is presented in [18] which is based on scale-invariant feature transform (SIFT) features. This study has been implemented to detect the number plate. The approach is used to identify the number plates with Chinese characters. Factors considered are illumination variation, partial occlusion, and defective character. The approach provides a 96% candidate filtration accuracy and 100% character segmentation accuracy.

An automatic number plate recognition for real-world application for the United Kingdom number plate recognition in [19]. Several real-world aspects are incorporated in the system including the screw fixing, font variability, spacing, color, and different symbols present on the number plates. A two-stage license plate recognition approach is presented in [20] where the plate localization is carried out by fuzzy disciplines while the recognition is done using neural networks. Images taken from various scenes are used to evaluate the performance of the proposed approach. The system achieves 97.9% accuracy for the license plate location while the plate identification rate is 95.6%.

The authors propose a sensors platform control a barrier that contains an image-based license plate recognition of the vehicle in [21]. The proposed approach performs plate recognition in challenging environments such as variations in license plate background, use of different fonts, and deformations. For this purpose, characters are recognized using a standard optical character recognition pipeline. The approach achieves 98%, 96%, and 93% accuracy for

license plate extraction, character segmentation, and character recognition, respectively. An approach is presented in [22] for multiple plate recognition using a single frame. The focus is to enhance the recognition accuracy for characters varying in plates or the size of the plates in the captured images. For plate detection, Spanish and Indian license plates are used, while for the recognition, only the Spanish plates are used.

A novel algorithm for vehicle license plate identification is proposed in [23] which uses adaptive image segmentation technique and connected component analysis. In addition, a character recognition neural network is adopted for character recognition. Results indicate that 96.5% of the plates can be segmented correctly with the proposed approach. Similarly, plate recognition accuracy is 89.1%. Research shows that plate recognition results are affected due to limitations in the distance, angle of view, illumination conditions, and background complexity. Better recognition accuracy can be achieved if these factors can be controlled.

Despite the accuracy of the above-cited research works, these approaches are limited by several factors. For example, several approaches focus on license plate recognition in static or less motion environments including the underground parking areas where the driving speed is limited. Similarly, other focus on single-stage deep-learning approaches where the plate recognition accuracy is high, however, the deployment in real-world environments is challenging due to computational complexity. Similarly, many important factors are not investigated extensively such as the use of non-conventional number plates with different fonts and sizes, and color schemes. This study aims at developing an approach which could recognize license plates in the challenging environment in developing countries like Pakistan with higher accuracy.

## Proposed methodology

Figure 3 shows the research methodology diagram proposed in the current study and all the steps are described in detail in subsequent sections. Methodology suggests about the usages of big data and big science surveys for the achievement of the required designs including the data science. These are done with the help of comparative reviews of the already designed mechanisms that are being implemented in this field of data mining and its classification as per localization and segmentations [24,25].

## Data gathering

At the first stage of the methodology, images of Pakistani local vehicles are collected from several different resources. Like many other developing countries, in Pakistan, license plate vendors do not follow a specific pattern regarding the size, font, and color, etc. In addition, people design license

plates to their liking and several other traditions including putting the names, pictures, and quotations are placed on or around the license plate number [26]. After gathering data, the label\_img software is used to make annotations of license plates and labels. This research uses two types of datasets: first is the vehicles and license plate dataset while the second is the numbers and character dataset.

## Vehicle license plate dataset

The vehicle license plate dataset consist of 2131 images and have two folders of images and annotations. The image folder contains 2131 images of Pakistani vehicles that do not follow a specific pattern. The annotation folder contains coordinates ( $x$ -min,  $y$ -min,  $x$ -max, and  $y$ -max) and these coordinates indicate the location of the license plate from the vehicle image.

## Character dataset

The character dataset consists of 571 plate character images with two folders of images and annotations. The image folder contains license plates and the annotation folder contains coordinates ( $x$ -min,  $y$ -min,  $x$ -max, and  $y$ -max), which represent the label obtained during segmentation. In this research, 37 labels are used (A–Z), (0–9), and (N/A) where N/A refers to not attempted [27]. In Pakistan, approximately eight types of license plates are used and the background color, font color, and font size are different for each style.

## Pre-processing

The purpose of pre-processing is to improve the quality of image data by suppressing unnecessary distortion or enhancing image functions that are important for further processing. As a result, distorted pixels can usually return to the average of adjacent pixels. The image pixel values are converted to  $416 \times 416$  sized gray images and fed into the neural networks. This procedure is done to avoid the unnecessary density of neural networks. For image pre-processing, following steps are carried out:

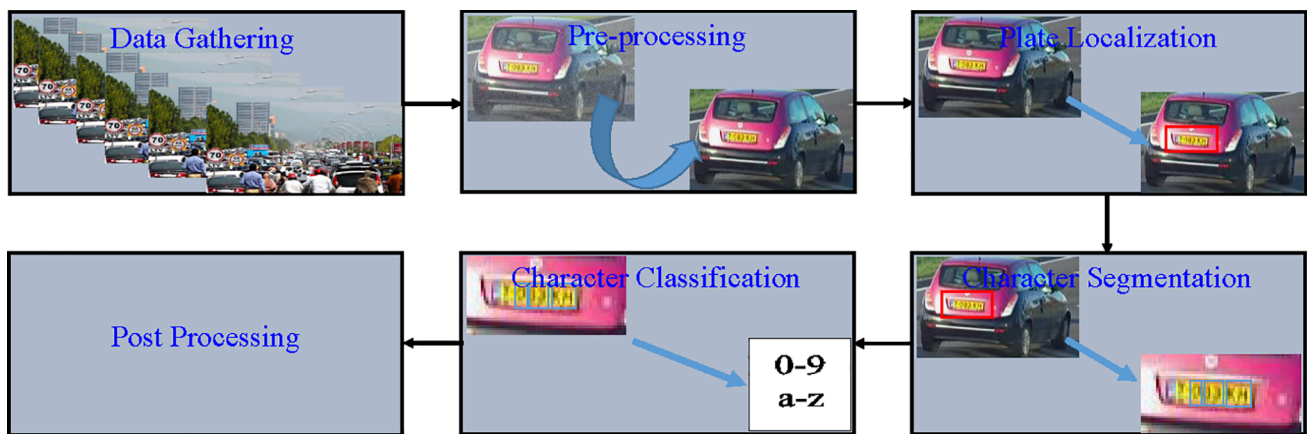
### Read images

In this step, first of all, we read the image. Images are in several different formats such as jpg, png, and MPEG.

### Resize image

In this step, the images are converted into a lower size using the Open CV library. Reducing the sizes helps to decrease the feature vector and computational complexity for neural networks.





**Fig. 3** Architecture of the proposed approach for vehicle plate identification

### Convert BGR2RGB

Datasets contain color images in blue, green, and red (BGR) which are converted to red green blue (RGB). Open cv read images are in BGR format, but our model works better with RGB, so the images are converted accordingly [28].

### Image normalization

Image normalization aims at preparing the data by putting all images into a common statistical distribution regarding the pixel values. For this purpose, the image array is divided by 255 to normalize the image. Image normalization for its pixel values is also called intensity normalization [29].

### Noise removal

Images often contain noise or blur called Gaussian noise. Gaussian noise (also called Gaussian blurring) is the result of using a Gaussian function that renders the image light. This is a widely used effect in graphic software to reduce image noise. The visual effect of this technique looks like a translation screen, with a slightly different buoyancy effect produced by a focus lens or an item under normal render. Gaussian optimism is the pre-processing step used in computer vision algorithms to improve image textures in different compartments [30].

### License plate localization

License plate localization (LPL) and character segmentation (CS) are important in the license plate (LP) recognition system. In LPL, for the analysis of this mechanism, histogram equalization is adapted. Hybrid binarization technique is also being suggested to divide the characters. Proposal of algo-

rithms are done to gain the localization and segmentation [31].

The input image is divided into  $M \times M$  grid cells. The cell in the grid is responsible for predicting the object on which the object is centered [32]. The output of the model will be the vector for  $M \times M (B \times 5 + C)$ , where  $B$  shows the defined bounding box and the confidence score for every grid cell and  $C$  represents the class probability of the predicted bounding box. Every Bounding box consists of five components ( $b_{bx}$ ,  $b_{by}$ ,  $b_{bw}$ ,  $b_{bw}$ ) and the confident score box  $C$  [33]. The center of the object represented by this ( $b_{bx}$ ,  $b_{by}$ ) coordinates relative to the grid cell position, and this ( $b_{bw}$ ,  $b_{bw}$ ) coordinate shows the bounding box width and height relative image dimension [16]. YOLOv3 predicts that there are multiple bounding boxes per grid cell, but selects a bounding box with the greatest union over (IOU) intersection with the true ground, which is called non-maximum suppression.

### Character segmentation

The segment license plate is skewed in several images which are segmented using space detection in the horizontal projection. We frequently use adaptation thresholding filters to increase the plate area before segmentation is performed. The threshold adaptation is used to separate the absence of light foreground from the illuminated background [21]. After thresholding, the horizontal projection  $P_x(x)$  of license plate ( $f(x, y)$ ) is calculated. We apply this projection to find the horizontal boundary between the segmented characters. Boundaries correspond to the peaks in a horizontal projection graphic [34]. Figure 4 shows the results of applying the thresholding.



Fig. 4 License plate after applying adaptive thresholding

## Feature extraction

The license plate's segments contain extra space and other unwanted elements in addition to alphanumeric characters. Since the segment is processed by an adaptive threshold filter, it carries only black and white pixels. Adjacent pixels are grouped into larger sections, one of which is characters. Our objective is to break the segment into sections and hold just a segment that represents the occurring at fixed intervals character. Figure 5 illustrates this concept.

Give this class a discrete function in a coordinate system by  $f(x, y)$  like that  $[0, 0]$  that shows the top-left intersection of segmentation and bottom of the right corner represent of  $[1, 1]$  width, height,  $w$  and  $h$  show segmentation dimensions. A 1 in  $f(x, y)$  represents black pixel and 0 represents white pixel space.  $P$  is all adjacent pixel  $[x, y]$  in which express extending components and  $[x, y]$  pixel is classified in a class to the  $P$  if it is found  $(x', y')$  one pixel from the piece in that case  $(x', y')$  and  $(x, y)$  these are neighbors of each other [24]. Figure 6 shows the results of applying horizontal seg-

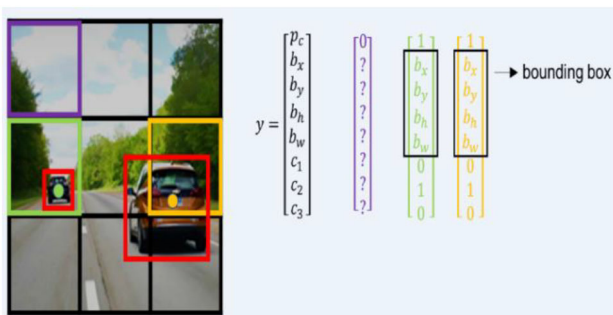


Fig. 5 Object bound boxes

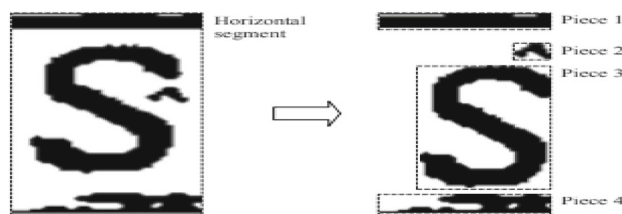


Fig. 6 License plate horizontal segmentation

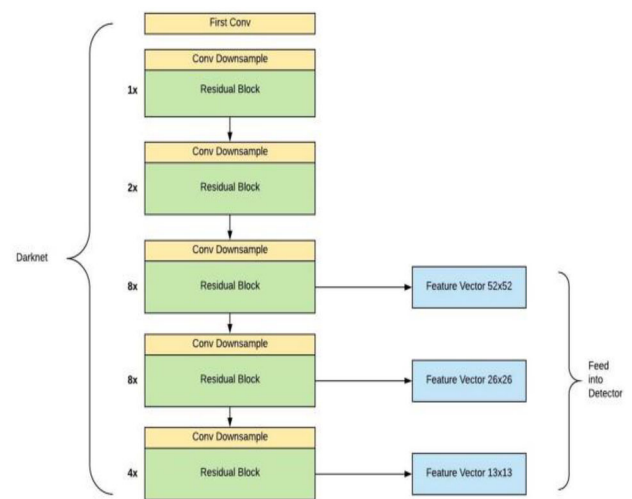


Fig. 7 CNN architecture methodology diagram

mentation.

$$[x, y] \in P \exists [x', y'] \in P : [x, y] N_4 [x', y'] \quad (1)$$

## Character classification

The current study uses the Darknet-53 CNN architecture for feature extraction and classification. The figure describes all the steps from input to output in methodology. Each layer of CNN consists of the following. The first convolutional layer 'Conv' is followed by the activation function. Additionally, it contains a pooling layer and a fully connected layer 'FC'. FC layer fundamentally works the same as a feed-forward neural network. We use many filters to apply the convolutional product on input and then use the activation function  $\psi$  [13].

## Convolutional layer

As a convolutional product, the two-dimensional matrix is the addition of an element-wise product. In general, images can mathematically be expressed as a tensor in the following dimension:

$$\dim(\text{image}) = n_H, n_W, n_C \quad (2)$$

where  $n_H$  and  $n_W$  represent the height and width of the image while  $n_C$  denotes the number of channels.

In RGB images, for example,  $n_C = 3$ , there are three types of color red, green, and blue. In practice, filter  $K$  is square and has an odd dimension representing  $f$ , which allows every pixel to the place of the center in the filter, all components around it. When running the product conventional filter or kernel ( $K$ ) it is necessary to have the same number of the channel as the image so that we can use the ordinary filter

on every channel [35]. Therefore, filter dimensions are as follows:

$$\dim(\text{filter}) = f, f, n_C \quad (3)$$

The product of the convolution between image and filter is a two-dimension matrix, in which every component is an addition of the product of the elements of the cube (filter) and the given image sub-cube, as shown in Fig. 8.

### Pooling layer

This step involves the down-sampling of image characteristics by summarizing the information it contains. The operation is performed on a per-channel basis, so only the dimensions ( $n_H, n_W$ ) are affected while  $n_C$  remains intact. Given an image, we move smoothly a filter behind a certain step without learning any parameters and then utilize a function to the selected components:

$$\dim(\text{conv}(I, K)) = \begin{cases} \left( \lfloor \frac{n_H+2p-f}{s} + 1 \rfloor, \lfloor \frac{n_W+2p-f}{s} + 1 \rfloor \right) & s > 0 \\ (n_H + 2p - f, n_W + 2p - f, n_C) & s = 0. \end{cases} \quad (4)$$

In practice, we use a square filter with a size of  $f$ , usually set to  $f = 2$ , and  $s = 2$ .

### Fully connected layer

A fully connected layer contains a limited number of neurons that accept an input vector and return another vector. In general, considering the  $j$ th node of the  $i$ th layer we hold the following equation:

$$A_j^{[i]} = \sum_{l=1}^{n_{i-1}} W_{j,l}^{[i]} a_l^{[i-1]} + b_j^{[i]}, \quad (5)$$

$$a_j^{[i]} = \psi^{[i]}(Z_j^{[i]}). \quad (6)$$

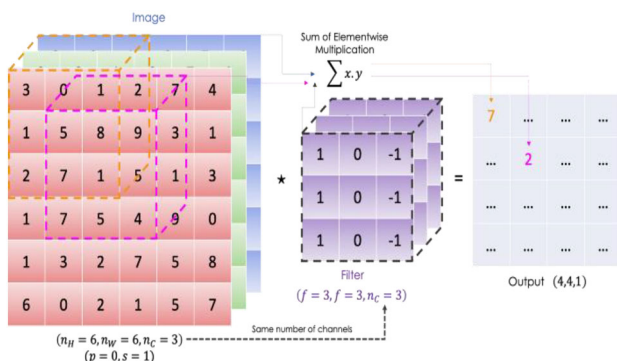


Fig. 8 Convolutional product on the two-dimension matrix

The input  $a^{[i-1]}$  might be the result of the convolution or a pooling layer with the dimension

$$(n_H^{[i-1]}, n_W^{[i-1]}, n_C^{[i-1]}), \quad (7)$$

while the vector having the dimension

$$(n_H^{[i-1]} * n_W^{[i-1]} * n_C^{[i-1]}), \quad (8)$$

which leads to

$$n_{i-1} = (n_H^{[i-1]} * n_W^{[i-1]} * n_C^{[i-1]}). \quad (9)$$

The learned parameters at the layers include weights  $w_{j,l}$  with  $n_{i-1} * n_l$  parameters and  $n_l$  bias parameters.

The operations of a fully connected layer are illustrated in Fig. 9.

### Training and testing

As per the default method, the dataset is divided into three categories training, validation, and testing. The modeled convolutional neural network is trained on training dataset that contains learning examples with the desired target. For validation, 2131 images are used with the trained model. Training data go through the pre-processing method involving BGR to RGB conversion, resizing images, the feature extracting, and then training. The trained model is later used for testing on the validation dataset. The training loss of the YOLOv3 algorithm is as follows:

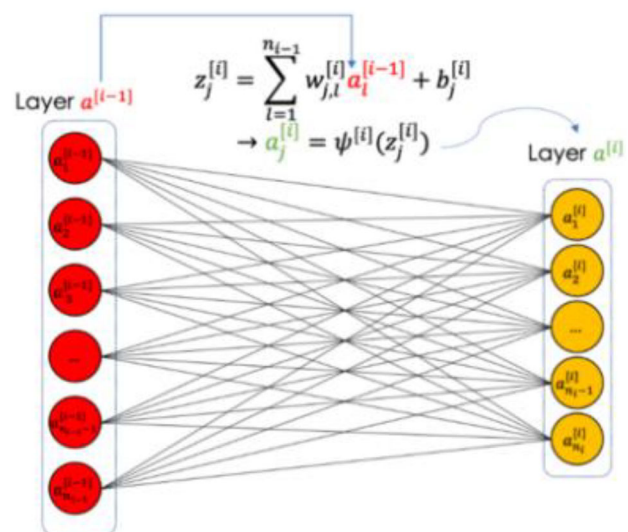


Fig. 9 Fully connected CNN

$$\begin{aligned}
& \lambda_{\text{Coord}} \sum_{i=0}^{s^2} 1_{ij}^{\text{obj}} [(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2] \\
& + \lambda_{\text{Coord}} \sum_{i=0}^{s^2} \sum_{j=0}^B 1_{ij}^{\text{obj}} \left[ \left( \sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left( \sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right] \\
& + \sum_{i=0}^{s^2} \sum_{j=0}^B 1_{ij}^{\text{obj}} (C_i - \hat{C}_i)^2 + \lambda_{\text{Coord}} \sum_{i=0}^{s^2} \sum_{j=0}^B 1_{ij}^{\text{obj}} (C_i - \hat{C}_i)^2 \\
& + \sum_{i=0}^{s^2} 1_{ij}^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{P}_i(c))^2
\end{aligned} \quad (10)$$

## Results and discussion

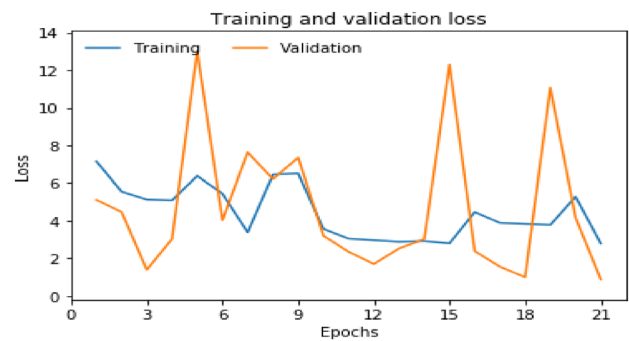
The experiments are performed using different network architectures including YOLOv3, Mask R-CNN, YOLOv2, SSD-300, and SSD-7. In the end, the results are compared using the mean average precision (mAP) metric. The section also describes the details of implementing the methodology, generating and analyzing graphs, and comparison of the existing approaches with the results achieved through the proposed approach.

### Experiment using YOLOv3

YOLOv3 is a fully convolutional network whose final output is generated by applying a  $1 \times 1$  kernel to the feature map [36]. In YOLOv3, detection is accomplished by applying a  $1 \times 1$  detection kernel on three different-sized features in three different locations in the network. 1st detection is performed through the 82nd layer. First, 81 layers image is down-sampled from the network, and 81 layer has 32 strides. If our image is  $416 \times 416$ , the resulting feature map is  $13 \times 13$ . A test is performed here using the  $1 \times 1$  detection kernel, which gives us the detection capabilities of  $13 \times 13 \times 255$ . The second detection is performed at layer 94, resulting in a  $26 \times 26 \times 255$  detection feature map, and the third and last detection is performed at the 106th layers, detection feature map is  $52 \times 52 \times 255$ . The training and validation loss result of YOLOv3 is shown in Fig. 10.

In this process, we use intersection over union (IOU) values after non-max suppression (NMS) as very necessary values for the assessment (confidence scores for specific class). A threshold of 0.3 is used in this and subsequent experiments. It means that if the intersection over union values is greater than 30% then identification is considered successful. Values below 50% will be logged as a detection fault.

The license plate dataset contains 2131 samples for training, 425 samples for validation. The proposed architecture is trained using 70 epochs. Results are provided in Table 1.



**Fig. 10** Training and validation loss of YOLOv3

**Table 1** YOLOv3 plate detection accuracy

Using IoU	0.5
Using object threshold	0.3
Non-maximum suppression	0.5
Plate	0.9782
mAP	0.9782

The character dataset contains 571 character samples for training and 114 samples for validation. A total of 37 labels are used in this training model and each label represents its results in Table 2. Character dataset training results are obtained using an IoV of 0.5, the threshold of 30, and NMS of 0.50.

### Experiment using mask R-CNN

The Mask regional convolution neural network (R-CNN) is a two-phase framework [37]. The 1st phase scans the image and produces a proposal (areas that may contain objects). The 2nd phase classifies the proposal and creates bounding boxes and masks. Both phases are associated with the backbone (standard CNN). The Mask R-CNN architecture consists of convolutional and average pooling layers, followed by a flattened convolutional layer, then two fully connected layers,

**Table 2** License plate label results

0:	0.9972	1:	0.9683	2:	1.0000	3:	0.9730
4:	0.9344	5:	0.9987	6:	0.9298	7:	1.0000
8:	0.9962	9:	0.9259	A:	1.0000	B:	0.9945
C:	1.0000	D:	1.0000	E:	0.9225	F:	1.0000
G:	0.9091	H:	1.0000	I:	1.0000	J:	1.0000
K:	0.8333	L:	1.0000	M:	1.0000	N:	0.8889
NA:	0.9821	O:	0.8421	P:	1.0000	Q:	1.0000
R:	1.0000	S:	1.0000	T:	0.8889	U:	1.0000
V:	1.0000	W:	0.9091	X:	1.0000	Y:	1.0000
Z:	1.0000	MAP	0.9701				



and at the end a soft-max classifier. The Mask R-CNN architecture is trained using 40 epochs to achieve the best accuracy results. The architecture accuracy depends on epochs and batch size. During training, each epoch represents a model loss. Iterations are the number of batches or steps required to complete the partitioned training packets required for an epoch. Figure 11 shows the training and validation loss of Mask R-CNN.

The accuracy of the plate dataset using Mask R-CNN is given in Table 3. The values used for IoU, threshold, and NMS are 0.5, 30, and 0.50, respectively.

The character dataset accuracy for each label is shown in Table 4. The labels contain A–Z, 0–9, as well as, 37 not attempted labels in the character dataset.

### Experiment using YOLOv2

YOLO version two removes all fully connected layers and predicts the bounding box using an anchor box. Remove a pooling layer to improve the output resolution. Currently, the size of images  $416 \times 416$  is used for training for network detection. And get  $13 \times 13$  the output of feature map, is obtained  $32 \times$  down-sampled [38]. This model uses Darknet-19 architecture and has 19 convolutional layers. First, YOLOv2 uses the Dark net's variant, which originally had 19-layer networks trained on Image-net. It also stacks 19 layers for detection tasks, providing a 19-layer fully convolutional underlying architecture for YOLOv2. That is why YOLOv2 is faster than YOLOv3. YOLOv2 is a CNN model made by 24 convolutional layers followed by 2 fully connected layers. Each epoch represents a model loss. Iterations

are the number of batches or steps required to complete the partitioned training packets required for epochs. Figure 12 shows the training and validation loss of YOLOv2.

Two datasets (plate dataset and character dataset) are trained by Darknet-19 architecture with 50 epochs. The accuracy of the plate dataset using YOLOv2 is given in Table 5.

The character dataset accuracy for each label is shown in Table 6. The labels contain A–Z, 0–9, as well as, 37 not attempted labels in the character dataset. The values for IoU, threshold, and NMS are 0.5, 30, and 0.50, respectively.

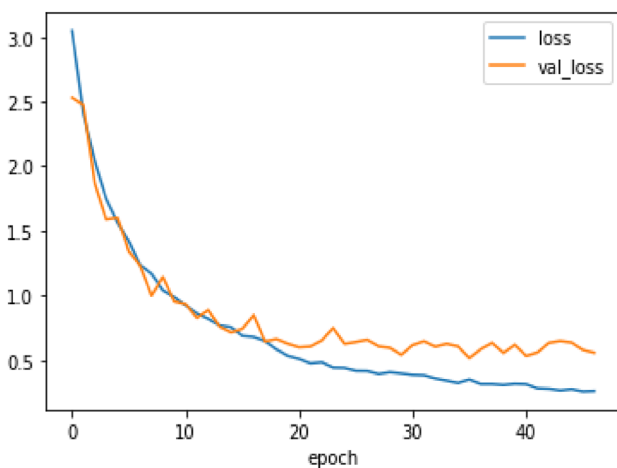
### Experiments using SSD-300

The SSD-300 object detection compile of two parts where the first part is the feature extraction map while in the second part convolution filters are applied to detect objects. SSD-300 uses VGG-16 to feature extraction maps. The VGG-16 CNN architecture consists of convolution layers, max-pooling layers, activation layers, and fully connected layers. In which 13 convolution layers, 5 max-pooling layers, and 3 dense layers sum up to 21 layers but only 16 of them are weight layers. Convolution 1 has as many filters as 64 while Convolution 2 has 128 filters, Convolution 3 has 256 filters while Convolution 4 and Convolution 5 each have 512 filters. The VGG-16 is trained by 38 epochs to achieve the best accuracy results. But the result of SSD-300 is very low because it needs more datasets to achieve good accuracy [39].

The training and validation loss plot given in Fig. 13 shows that the loss starts from 75.01 and reaches 70 in 30 epochs while validation loss starts from 85 and moves down to 80. Table 7 provides the accuracy of SSD-300 for plate detection and character recognition.

### Experiments using SSD-7

A small seven-layer version that provides relatively quick training from scratch, even on mid-tier GPUs, but sufficient for less complex object detection tasks and tests. This architecture works as SSD-300 but this is smaller compared to SSD-300. There are 2 reasons for superior efficiency with a single-shot approach. The regional proposal network and classification and location calculations are fully integrated. This minimizes redundant computing. A single shot is powerful for any number of objects in the image, and its calculated load is based on the number of anchor points. The SSD-7 architecture is trained by 60 epochs to achieve the best accuracy results. SSD-7 architecture accuracy depends on a huge amount of dataset and 1000 epochs. During the training of SSD-7 architecture, each epoch represents a model loss. Iterations are the number of batches or steps required to complete the partitioned training packets required for epochs. Two datasets (plate dataset and characters dataset) are trained by SSD-7 architecture with 60 epochs. But the accuracy of



**Fig. 11** Training and validation loss of Mask R-CNN

**Table 3** Plate detection accuracy using Mask R-CNN

Plate	0.8625
Overall map	0.8624

**Table 4** License plate label results using Mask R-CNN

0:	0.4905	1:	0.9213	2:	0.8873	3:	0.8758
4:	0.8496	5:	0.6481	6:	0.7769	7:	0.9025
8:	0.8741	9:	0.8867	A:	0.6411	B:	0.8846
C:	0.9200	D:	0.9130	E:	0.8750	F:	0.9487
G:	0.8709	H:	1.0000	I:	1.0000	J:	0.9387
K:	0.8636	L:	0.8521	M:	1.0000	N:	0.9705
NA:	0.4545	O:	P:	0.8823	Q:	1.0000	
R:	0.9729	S:	0.8095	T:	0.7619	U:	0.8775
V:	1.0000	W:	0.920000	X:	1.0000	Y:	1.0000
Z:	0.5200	MAP	0.86085				

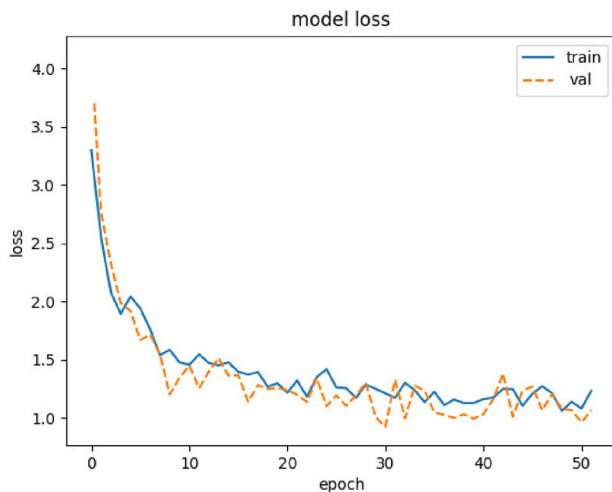
**Fig. 12** Training and validation loss of YOLOv2. The  $x$  and  $y$  axis represents the number of epochs and loss, respectively**Table 5** Plate detection accuracy using YOLOv2

Plate	0.8624
Overall map	0.8624

this model is very low because our datasets are small. Training and validation loss of SSD-7 model is shown in Fig. 14.

The accuracy of this experiment is very low because SSD-7 requires a large dataset to achieve better results. If we training it on a small dataset then accuracy is very low. The plot presents the loss of both, training set and validation set during 30 epochs of training the training loss of the proposed model starts from 16.9401 loss to 6.0305 and the second yellow line shows the validation loss which starts from 13.3453 loss then down to 6.98. Character recognition and license plate detection accuracy of SSD-7 is provided in Table 8.

### Performance analysis of various models

The performance of the implemented models to detect and recognize license plates is compared. For this purpose, both the plate detection accuracy and character recognition accu-

**Table 6** License plate label results using YOLOv2

0:	0.8018	1:	0.7977	2:	0.8139	3:	0.8409
4:	0.7876	5:	0.7555	6:	0.8006	7:	0.8020
8:	0.8736	9:	0.8771	A:	0.7615	B:	0.9883
C:	0.8423	D:	0.9166	E:	0.8552	F:	0.8581
G:	0.9058	H:	0.8539	I:	0.9278	J:	0.8202
K:	0.9210	L:	0.7867	M:	0.9295	N:	0.9662
NA:	0.3092	O:	0.8661	P:	0.7398	Q:	1.0
R:	0.8463	S:	0.8434	T:	0.8425	U:	0.7407
V:	0.8469	W:	0.7972	X:	0.9439	Y:	0.8448
Z:	0.7931	MAP	0.8051				

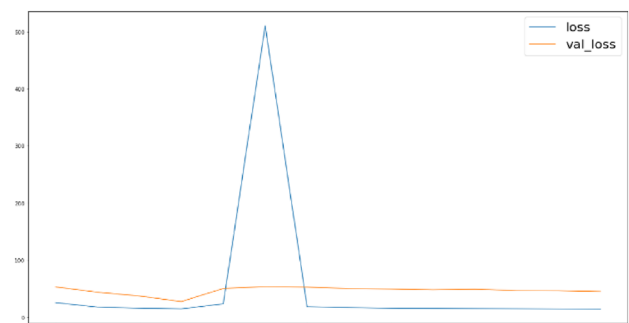
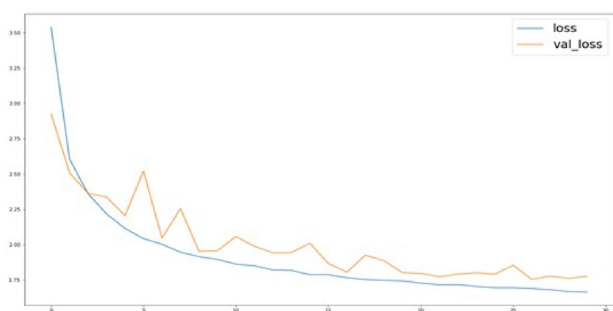
**Fig. 13** Training and validation loss of SSD-300**Table 7** Plate detection accuracy using SSD-300

Plate detection	0.3834
Character recognition	0.4067

rary is evaluated. Results indicate that YOLOv3 achieves the highest accuracy with the methodology adopted in the current study. The achieved accuracy for YOLOv3 is 0.9782 each for plate detection and character recognition. Mask R-CNN achieves the second-best accuracy of 0.9624 for plate detection, however, its character recognition accuracy is only 0.8401 which is very low as compared to YOLOv3.



**Fig. 14** Training and validation loss of SSD-7. The  $x$  and  $y$  axis represents the number of epochs and loss, respectively

**Table 8** Plate detection and character recognition accuracy of SSD-7

Plate detection	0.0047
Character recognition	0.0040

**Table 9** Comparison of results for different models for plate detection

Classification model	Epochs	Loss	mAP
<i>Plate detection accuracy</i>			
YOLOv3	70	2.18	0.9782
Mask R-CNN	40	0.1167	0.9624
YOLOv1	50	4.67	0.8624
SSD-7	60	6.0303	0.8624
SSD-300	38	6.981	3.86
<i>Character recognition accuracy</i>			
YOLOv3	70	0.1768	0.9782
Mask R-CNN	40	0.1482	0.8401
YOLOv1	50	0	0.8051
SSD-7	60	5.9190	0.0017
SSD-300	38	2.14	0.4067

**Fig. 15** Versatility in the color and size of the license plates



## Real-time testing

Apart from the experimental results discussed in the preceding sections, the proposed methodology is tested in a real-time setting. Experiments are performed on the images of Pakistani vehicles taken in real-time. As shown in Fig. 15, the versatility in the car plate size, its hung position, and varying colors pose a huge challenge. Due to this heterogeneity, where different types of license plates are used, character recognition becomes a very difficult task.

The input image contains license plate characters in two rows. It is very difficult for most of the existing methods discussed in the literature while there are two rows on the license plate. The vehicle license plate has a white background and black characters on the plate. When we input this image in our license plate recognition system, first our system detects the license plate, then characters recognition is processed. The results are shown in Fig. 16.

Figure 17 shows that our proposed technique is capable of detecting two license plates from one input image. The picture has the one vehicle in the image but with an additional license plate on key chain. The output shows extra license plates detected with the original one.

## Conclusion

With the increase in the number of vehicles, vehicle tracking has become an important research area for efficient traffic control, surveillance, and finding stolen cars. For this purpose, efficient real-time license plate detection and recognition are of great importance. Due to the variation in the background and font color, font style, size of the license plate, and non-standard characters, license plate recognition is a great challenge in developing countries like Pakistan. To overcome such issues, this study applies a deep-learning

Number Plate Character Recognition



Fig. 16 Detection of license plate with two rows



Fig. 17 Detection of two license plates in a single image

strategy to improve license plate recognition efficiency. The proposed methodology is tested on a huge image data set consisting of nine different types of license plates found in Pakistan. The collected images have been captured under various lighting/contrast conditions, distance from the camera, varying angle of rotation, and validated to produce a high recognition rate. The approach can be effectively used by law enforcement agencies and private organizations to improve homeland security. Future work may include training and validation of the existing algorithm using the hybrid classifier method and improvement of the robustness of the license plate recognition system in varying weather conditions.

**Conflict of interest** The authors declare that there is no conflict of interests.

**Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

## References

- Shashirangana J, Padmasiri H, Meedeniya D, Perera C (2020) Automated license plate recognition: a survey on methods and techniques. *IEEE Access* 9:11203–11225
- Rio-Alvarez A, de Andres-Suarez J, Gonzalez-Rodriguez M, Fernandez-Lanvin D, López Pérez B (2019) Effects of challenging weather and illumination on learning-based license plate detection in noncontrolled environments. *Scientific Programming*, vol. 2019,
- Chen Y-T, Chuang J-H, Teng W-C, Lin H-H, Chen H-T (2012) Robust license plate detection in nighttime scenes using multiple intensity ir-illuminator. In: *IEEE international symposium on industrial electronics*. IEEE 2012, pp 893–898
- Wen Y, Lu Y, Yan J, Zhou Z, von Deneen KM, Shi P (2011) An algorithm for license plate recognition applied to intelligent transportation system. *IEEE Trans Intell Transp Syst* 12(3):830–845
- Raghunandan K, Shivakumara P, Jalab HA, Ibrahim RW, Kumar GH, Pal U, Lu T (2017) Riesz fractional based model for enhancing license plate detection and recognition. *IEEE Trans Circuits Syst Video Technol* 28(9):2276–2288
- Wang D, Tian Y, Geng W, Zhao L, Gong C (2020) Lpr-net: Recognizing Chinese license plate in complex environments. *Pattern Recogn Lett* 130:148–156
- Azam S, Islam MM (2016) Automatic license plate detection in hazardous condition. *J Vis Commun Image Represent* 36:172–186
- Roy S (2020) Automatics number plate recognition using convolutional neural network. *Azerbaijan Int J High Perform Comput* 3(2):234–244
- Çavdaroglu GÇ, Gökmen M (2021) A character segmentation method to increase character recognition accuracy for Turkish number plates. Preprint. <https://doi.org/10.20944/preprints202104.0440.v1>
- Sarfraz MS, Shahzad A, Elahi MA, Fraz M, Zafar I, Edirisinghe EA (2013) Real-time automatic license plate recognition for cctv forensic applications. *J Real-Time Image Proc* 8(3):285–295
- Badr A, Abdelwahab MM, Thabet AM, Abdelsadek AM (2011) Automatic number plate recognition system. *Ann Univ Craiova-Math Comput Sci Ser* 38(1):62–71
- Bhardwaj ED, Gujrat E (2014) Automated Number Plate Recognition System Using Machine learning algorithms (Kstar). *Int J Res Edu and Sci Method* 2(6):15–21
- Hsieh C-T, Juan Y-S, Hung K-M (2005) Multiple license plate detection for complex background. In: *19th international conference on advanced information networking and applications (AINA'05) volume 1 (AINA papers)*, IEEE, vol 2, pp 389–392



14. Radzi SA, Khalil-Hani M (2011) Character recognition of license plate number using convolutional neural network. In: International visual informatics conference. Springer, New York, pp 45–55
15. Abdullah M, Bakhtan MAH, Mokhtar SA (2017) Number plate recognition of Malaysia vehicles using smearing algorithm. *Sci Int (Lahore)* 29(4):823–827
16. Bhushan B, Singh S, Singla R (2013) License plate recognition system using neural networks and multithresholding technique. *Int J Comput Appl* 84(5):45–50
17. Hamey LG, Priest C (2005) Automatic number plate recognition for Australian conditions. In: Digital image computing: techniques and applications (DICTA'05). IEEE p 14
18. Wang Y, Ban X, Chen J, Hu B, Yang X (2015) License plate recognition based on sift feature. *Optik* 126(21):2895–2901
19. Rhead M, Gurney R, Ramalingam S, Cohen N (2012) Accuracy of automatic number plate recognition (anpr) and real world UK number plate problems. In: 2012 IEEE international carnavan conference on security technology (ICCST). IEEE, 2012, pp 286–291
20. Chang S-L, Chen L-S, Chung Y-C, Chen S-W (2004) Automatic license plate recognition. *IEEE Trans Intell Transp Syst* 5(1):42–53
21. Ullah F, Anwar H, Shahzadi I, Ur Rehman A, Mahmood S, Niaz S, Mahmood Awan K, Khan A, Kwak D (2019) Barrier access control using sensors platform and vehicle license plate characters recognition. *Sensors* 19(13):3015
22. Menon A, Ommann B (2018) Detection and recognition of multiple license plate from still images. In: 2018 international conference on circuits and systems in digital enterprise technology (ICCSDET). IEEE, 2018, pp 1–5
23. Du S, Ibrahim M, Shehata M, Badawy W (2012) Automatic license plate recognition (alpr): a state-of-the-art review. *IEEE Trans Circuits Syst Video Technol* 23(2):311–325
24. Chen Y, Chen H, Gorkhali A, Lu Y, Ma Y, Li L (2016) Big data analytics and big data science: a survey. *J Manag Anal* 3(1):1–42
25. Zheng L, Sayed T, Mannering F (2020) Modeling traffic conflicts for use in road safety analysis: a review of analytic methods and future directions. *Anal Methods Accident Res* 29
26. Puranic A, Deepak K, Umadevi V (2016) Vehicle number plate recognition system: a literature review and implementation using template matching. *Int J Comput Appl* 134(1):12–16
27. Yu C, Xie M, Qi J (2008) A novel system design of license plate recognition. In: International symposium on computational intelligence and design, vol. 2. IEEE 2008, pp 114–117
28. Xie L, Ahmad T, Jin L, Liu Y, Zhang S (2018) A new cnn-based method for multi-directional car license plate detection. *IEEE Trans Intell Transp Syst* 19(2):507–517
29. Babu DM, Manvitha K, Narendra M, Swathi A, Varma KP (2015) Vehicle tracking using number plate recognition system. *Int J Comput Sci Inf Technol* 6(2):1473–1476
30. Puarunroj W, Boonsirisumpun N (2018) Thai license plate recognition based on deep learning. *Proc Comput Sci* 135:214–221
31. Guo J-M, Liu Y-F (2008) License plate localization and character segmentation with feedback self-learning and hybrid binarization techniques. *IEEE Trans Veh Technol* 57(3):1417–1424
32. Fahmy MM (1994) Automatic number-plate recognition: neural network approach. In: Proceedings of VNIS'94-1994 vehicle navigation and information systems conference. IEEE, pp. 99–101
33. Patel C, Shah D, Patel A (2013) Automatic number plate recognition system (anpr): a survey. *Int J Comput Appl* 69(9):21–33
34. Kumthekar A, Owhal MS, Supekar MS, Tupe MB (2018) Recognition of vehicle number plate using raspberry pi. *Int Res J Eng Technol* 5(4):391–394
35. Haider SA, Khurshid K (2017) An implementable system for detection and recognition of license plates in Pakistan. In: 2017 international conference on innovations in electrical engineering and computational technologies (ICIEECT). IEEE, pp 1–5
36. Farhadi A, Redmon J (2018) Yolov3: An incremental improvement. *Comput Vis Pattern Recogn (cite as)*
37. He K, Gkioxari G, Dollár P, Girshick R (2017) Mask r-cnn. In: Proceedings of the IEEE international conference on computer vision, pp 2961–2969
38. Nakahara H, Yonekawa H, Fujii T, Sato S (2018) A lightweight yolov2: a binarized cnn with a parallel support vector regression for an fpga. In: Proceedings of the 2018 ACM/SIGDA international symposium on field-programmable gate arrays, pp 31–40
39. Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu C-Y, Berg AC (2016) Ssd: single shot multibox detector. European conference on computer vision. Springer, New York, pp 21–37

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.