AUT Journal of Electrical Engineering
10.22060/EEJ.2024.23296.5601

Persian Car License Plate Recognition using Deep Convolutional Neural Networks

Hossein Noori[1] (h.noori@vru.ac.ir)

[1]Assistant professor, Electrical Engineering, Vali-e-Asr University of Rafsanjan, Rafsanjan, Iran.

Abstract:

Car license plate recognition plays a very important role in intelligent transportation systems. In this paper, a new car license plate recognition algorithm is proposed. To recognize the car license plate, first, a new dataset in different light conditions is collected and labeled. Then a new convolutional neural network is proposed to find the position of a car license plate in an image, named plate position detection network (PPDN). The characters on the plate are separated by a proposed algorithm after the position of the plate is determined. Since the Iranian car license plate includes both digits and letters, two other convolutional neural networks are proposed: one for digit recognition, named digit recognition network (DRN), and the other for letter recognition, named letter recognition network (LRN). All three networks are trained with related datasets. Simulation results show that the PPDN reaches 99.86% training accuracy, 99.75% validation accuracy and 99.53% test accuracy. Also, DRN reaches 100% training accuracy, 99.6% validation accuracy, and 99.51 % test accuracy. In addition, LRN reaches 100% test accuracy, 99.45% validation accuracy and 99.05% test accuracy. Finally, the accuracy of the network for plate recognition on the test data is 99.21%.

Keywords:

Deep learning, object detection, pattern recognition, intelligent transportation system.

1. Introduction

An Intelligent Transportation System (ITS) is a crucial component of modern society, and effective transportation system management requires the use of Automatic License Plate Recognition (ALPR). This technology enables the detection and recognition of license plates, making it a valuable tool for identifying vehicles subject to traffic bans and intercity tolls, which has garnered attention from numerous organizations.

ALPR typically consists of three stages: the first involves detecting the position of the license plate, the second focuses on character segmentation (CS), and the final stage is the recognition of the segmented characters. Among these steps, the initial stage—license plate detection (LPD)—is the most critical for the optimal and effective performance of ALPR. Performance can be significantly influenced by various factors, including weather conditions, lighting variations, and camera angles. Therefore, achieving a robust LPD is essential for mitigating the impact of these variables.

Recent research in car license plate recognition has largely fallen into two categories: traditional approaches [1] - [10], and deep learning-based methods [11]- [27]. Traditional approaches rely on manually defined features for plate position detection and character recognition, such as color, shape, aspect ratio, and corners, which are then input into classifiers like support vector machines (SVM) or artificial neural networks. In contrast, deep learning methods utilize convolutional neural networks (CNNs) to perform both feature extraction and classification simultaneously.

To enhance accuracy under varying light conditions while maintaining algorithmic speed, we propose a new method for recognizing Iranian car license plates. This study introduces a new dataset that is combined with an existing one to improve reliability and accuracy. Our proposed algorithm consists of four sections and incorporates three convolutional neural networks (CNNs) to identify plate positions and subsequently separate and recognize characters.

Initially, we present a CNN designed specifically for detecting the position of Iranian car license plates. Following this, we introduce an innovative algorithm for character segmentation based on approximate positioning and image histograms. Once the characters are separated, another CNN is utilized to recognize the individual digits and letters. Simulation results demonstrate that our proposed algorithm achieves superior accuracy and precision compared to existing methods, while also being faster than state-of-the-art algorithms. The main contributions of this paper are outlined as follows:

- A new dataset has been collected and combined with an existing one to enhance the reliability and accuracy of the proposed method.
- A new convolutional neural network, featuring innovative layer connections and novel combinations of various attributes, is proposed, and trained to detect plate positions under varying light conditions.
- A new algorithm is proposed for character separation, utilizing image histogram and the approximate locations of characters.
- A new convolutional neural network is proposed and trained to recognize the digits on car license plates.
- A new convolutional neural network is proposed and trained to recognize the letters on car license plates.
- The effectiveness of the proposed algorithm is validated through six criteria: accuracy, precision, recall, F1-score, mean Average Precision (mAP), and implementation time (speed).

The remainder of this paper is organized as follows: related works are discussed in Section 2. Section 3 describes the collected dataset for PPDN. Section 4 introduces the PPDN, while Section 5 addresses character recognition, featuring three subsections: character separation, a discussion on DRN, and an introduction to LRN. In Section 6, the new algorithm is proposed. Section 7 presents the simulation results, discusses hyperparameter selection and implementation details, and compares the outcomes with existing algorithms. Finally, Section 8 concludes the paper.

2. Related Work

This section reviews several algorithms for car license plate recognition. In [1], a color-based method identifies Iranian license plates by detecting a distinctive blue strip and analyzing geometric features. Characters are classified using a hybrid decision tree and SVM, achieving 94% accuracy, but it struggles under varying lighting conditions. Authors in [2] present an algorithm using the HSV color space, separating foreground and background via a hue histogram and employing features like rectangularity and edge density. Its accuracy ranges from 60% to 97%, indicating inconsistency. Authors in [3] also use color features for plate location, while [4] maps features from the HSV color space to a fuzzy set for detection. In [5], a color edge detection algorithm is proposed but has above 90% accuracy and high implementation time. In [6] authors introduce a cascade classifier utilizing Adaboost to detect multiple Iranian license plates, though it is complex and performs poorly in low light. Reference [7] improves upon this with a two-phase learning algorithm.

In [8], a new algorithm for Iranian car license plate detection is introduced. The method first detects the plate position using seven features, which is then converted into a 32×30-pixel image. The characters in this image are recognized using K-nearest neighbor (KNN) and support vector machine (SVM) techniques. The authors considered 26 features for KNN and 19 features for SVM. While the method achieves a maximum accuracy of 97.03%, it is complex to implement. In [9], a fast and accurate algorithm for Iranian car license plate detection is proposed. The authors [9] utilized two datasets containing 67 and 492 images, respectively. They first enhanced the contrast of the images and then focused on locating the numbers to determine the car license plate’s position. The reported accuracy for the datasets is 100% and 99.95%, with implementation times of 109 milliseconds and 17.5 milliseconds, respectively. However, the algorithm [9] only detects the plate’s position and does not recognize any digits or letters. Additionally, the size of their dataset is relatively small, with around 500 samples. In [10], the authors propose an algorithm that extracts features using the histogram of oriented gradients (HOG) along with geometric features, followed by a feature selection process using a novel entropy-based method. Classification is conducted using a support vector machine (SVM), achieving a maximum accuracy of 99.5%. Traditional algorithms in this category require human intervention for feature determination, making them subjective and more complex than deep learning-based approaches. Additionally, they generally produce poorer results compared to deep learning methods. However, traditional algorithms do not require a dataset or labeling.

In deep learning-based approaches, a trained network is provided with just an image, allowing it to recognize the plate position or the characters on the plate. Recently, deep neural networks have gained significant attention due to their powerful capabilities. The authors in [11] propose a network called VSnet for car license plate recognition. This method [11] consists of two components: VertexNet for license plate position detection and SCR-Net for license plate recognition, integrated through a resampling-based cascaded approach. In VertexNet [11], the authors introduce an efficient integration block to extract spatial features from license plates, while SCR-Net employs a horizontal encoding technique for left-to-right feature extraction. Additionally, they propose a weight-sharing classifier for character recognition. In [12], the authors propose a method for detecting cars in images using You Only Look Once version 2 (YOLOv2), followed by another network for detecting the plate's position. They introduce [12] a new convolutional neural network architecture called Warped Planar Object Detection Network (WPODNET) for plate position detection. Finally, characters are recognized using an optical character recognition network (OCR-NET). The proposed networks in [12] were trained and tested on five datasets, achieving an accuracy of 89.33%.

In [13], the authors aimed to enhance the character recognition step in car license plate recognition. They proposed using You Only Look Once (YOLO) to eliminate the need for both character segmentation and traditional OCR. The method achieved an accuracy of 99.2% in recognizing plate characters. In [14], the authors propose a method for car license plate detection (not recognition) using a two-branch convolutional neural network. In this approach [14], cars are detected utilizing high-level features, while license plates are identified using low-level features. The highest accuracy achieved in [14] was 96.78%.

The authors in [15] present an ALPR system based on the YOLO object detector, designed to be robust to varying conditions such as light and background changes. Their two-step approach [15] for character segmentation and recognition was trained and tested on two datasets: the SSIG dataset, which consists of 2,000 frames from 101 vehicle videos, achieving a recognition rate of 93.53% with 47 frames per second (FPS), and the UFPR-ALPR dataset, comprising 150 videos and 4,500 frames, where the system achieved 78.33% accuracy. In [16], the authors utilize two successive YOLOv3 networks to automatically detect and recognize Persian characters from input images. The system, trained on 5,719 images, achieves an accuracy of 95.05% in 119.73 milliseconds per image[17]. In [18], a new dataset for car license plate detection is introduced, along with two shallow convolutional neural networks that achieve a maximum accuracy of 99.09%. In [19], the authors introduce a new dataset and propose an algorithm based on YOLO and YOLOv2 for detecting car license plates under poor weather conditions, demonstrating that YOLOv2 outperforms YOLO. In [20], the authors propose an encoder-decoder network to highlight digits on Iranian car license plates, followed by recognition using a recurrent neural network that does not require character segmentation. The highest reported accuracy is 94.19% on a test database of 4,000 images. It is important to note that the Iranian license plates include a letter along with seven digits, which the method in [20] cannot recognize. In [21], the authors propose a method to locate Chinese car license plates using convolutional neural networks. In [22], an algorithm based on YOLO is introduced for automatic license plate detection. After processing to separate the characters, a ResNet model [22] is used for recognition, achieving a maximum accuracy of 80%. There are similar works in [23-25].

In [26], the authors employ transfer learning to detect and recognize characters on license plates. They [26] use edge detection to locate the plates, which is sensitive to noise, and apply various image processing techniques to separate the characters. Finally, they recognize the characters using AlexNet, achieving an accuracy of 98.5%. A new deep learning approach for car license plate recognition is proposed in [27]. It first uses the YOLOv4-tiny model to detect the license plate, followed by a recurrent neural network with connectionist temporal classification for character recognition. The end-to-end processing time is 0.435 seconds, with an accuracy of 75.14%.

This paper proposes a novel algorithm for Iranian car license plate recognition, featuring three new convolutional neural networks: PPDN for plate position detection, DRN for digit recognition, and LRN

10.22060/EEJ.2024.23296.5601

for letter recognition. Additionally, a simple character segmentation algorithm is introduced. To ensure reliable results, PPDN is trained, validated, and tested on two datasets: one collected by the author from Kerman and Rafsanjan cities under various lighting conditions to ensure robustness against light variations, and another available in [28, 29]. The DRN is trained, validated, and tested on two datasets: the Persian handwritten digit dataset (HODA) [30] and an Arabic digit dataset [31]. The LRN is trained, validated, and tested using the dataset from [32]. The proposed algorithm achieves 99.53% accuracy for plate position detection, 99.51% for digit recognition, 99.05% for letter recognition, and 99.21% for overall plate recognition. It takes just 0.133 seconds to recognize a car license plate, making it suitable for online traffic control.

3. Dataset Explanation

In this paper, a new algorithm based on deep learning is proposed for Iranian car license plate recognition which is fully automatic. To this goal, first, the position of license plate should be detected. To determine the license plate position, a new convolutional neural network is proposed (i.e. PPDN), a dataset is needed to train PPDN. In this paper, a new dataset is collected and combined with another existing dataset which is explained in following.

3-1- Collected dataset

To collect images, two students of Vali-e-Asr Rafsanjan University (VRU) are trained, they collected pictures from cars in two cities: Kerman, and Rafsanjan utilizing cell phone Huawei Honor 9X. They collected 1400 pictures from different cars in different times and weather conditions. Fig. 1 shows some samples of this dataset in different conditions. As Fig. 2 shows, there were some samples including two or more car license plates. Also, there were some images in which the plate was unreadable or blurred, Fig. 3 shows a sample including unreadable and blurred car license plates. Beside this, the camera was not in the line of car license plate in all images, and also, the plate was not horizontally aligned in images, Fig. 4 shows two samples of these.

AUT Journal of Electrical Engineering
10.22060/EEJ.2024.23296.5601

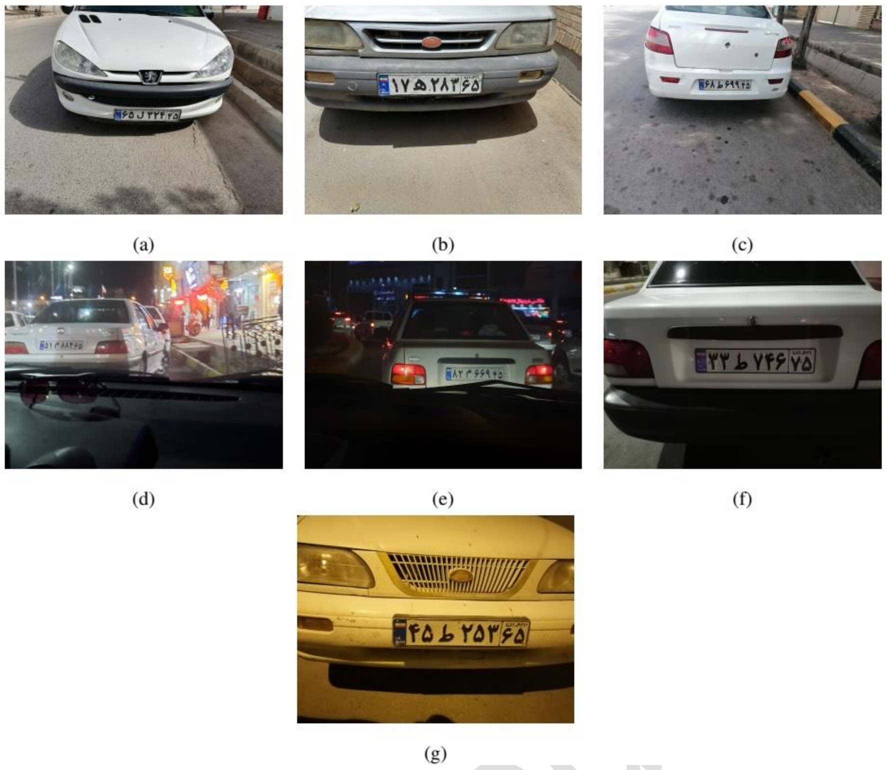


Figure 1: Car image in different time of day (a) morning (b) noon (c) evening (d) night without light (e) night with car light (f) night with a wight lamp of a post (g) night with a yellow lamp of a post.

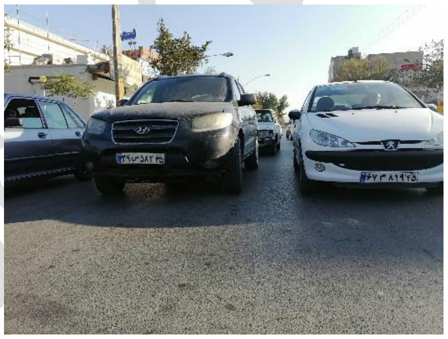


Figure 2: Image including more car license plate

In order to focus on the car license plate recognition in the entrance gates, all images including more than one car license plate, blurred or unreadable car license plate and oblique plates are eliminated. Deleting such samples decreases the size of collected dataset to 1280 images. Then to increase the dataset size, the collected dataset is combined with another existing the dataset [28], which is collected in Isfahan, Iran. After combination of the collected dataset with the one in [28], the size of dataset reached to 1500 samples, from them 150 (10%) samples are allocated for validation and the network is trained with the others (1350 samples). To test the network, 127 other samples are taken with Xiaomi POCO X3 NFC.

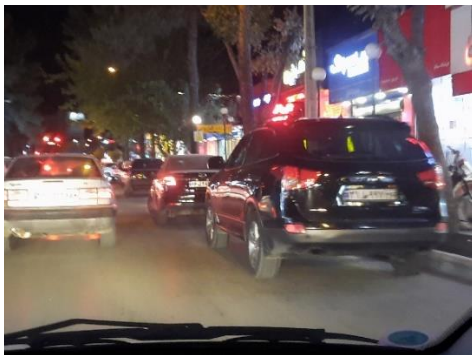


Figure 3: Image including blurred license plate

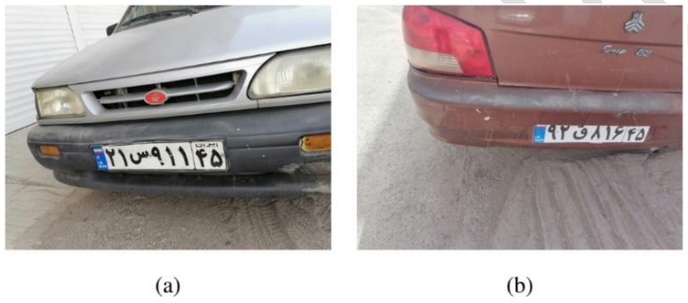


Figure 4: Different angle of camera (a) camera is not horizontally aligned (b) the image is taken from top but is readable

3-2- Pre-processing

All images in the data set are taken by a cell phone Huawei Honor 9X which includes 4 cameras: standard, wide, ultrawide and depth measurement. Huawei Honor 9X takes 4000 × 3000 pixel images (12 Mega pixels). As we know, these images are extremely large to feed into a convolutional neural network. Therefore, a downsampleing is performed to resize the images to 256 × 256.

Next, images in the dataset are labeled by a binary image in such a way that car license plate position's pixels are shown with white color and other sections of the image are shown with black color. Fig. 5 shows an original image, labeled image, and detected position of car license plate.

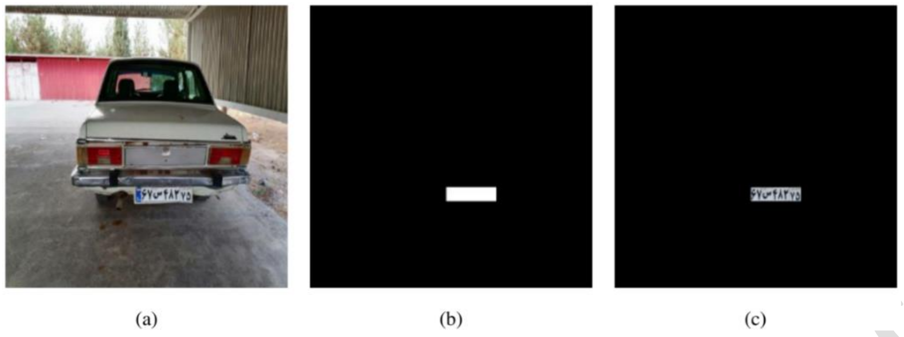


Figure 5: Image and its label (a) Original image (b) Label of original image (c) Detected place of car license plate

4. Proposed Plate Position Detection Network (PPDN)

In this section, a new convolutional neural network architecture is proposed to find the car license plate position. As it is discussed in section 3-2 and the Fig. 5 shows the input and output of the PPDN should be images with the same size. Fig. 6 shows the architecture of the proposed network. As Fig. 6 shows, the proposed architecture includes three parts: the goal of the first part is to find features from low to high level. The second part's goal is to approximate primary position of the car license plate from detected features. Finally, the third part is designed to finalize the plate position.

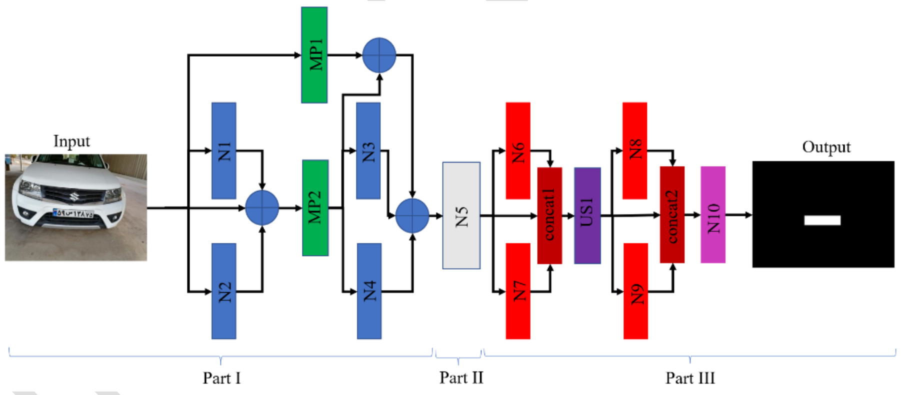


Figure 6: Architecture of the proposed convolutional neural network

In Fig. 6, each blue rectangle represents a three-layer convolutional neural network whose parameters are shown in Tables 1 and 2. These networks are considered to extract low level features of the original image. After features are extracted in N1 and N2, they are combined with original image (in Add layer) in order to extract high level features by N3 and N4. It is necessary to mention that each blue circle in Fig. 6

shows an Add layer. Each green rectangle represents a Maxpooling layer in Fig. 6, which tries to decrease the size of input to speed up the performance. The step of these two Maxpooling layers (green rectangles) is 4, which means they reduce the size of input image from 256×256×3 to 64×64×3. The main idea of the first part of the proposed architecture is that N1 and N2 finds the low-level features, and they are combined with each other and the original image for higher level features detection. The N4 and N3 get the output of N1 and N2 and try to find high level features (higher level than the previous layers). Again, original image, low- and high-level features are combined to fed into second part of the proposed architecture.

Table 1. Parameters of Network N1 and N2

Layer Type	Kernel size	# of kernels	stride	Input size	Output size	Activation function
Convolution	3	8	1	256×256×3	256×256×8	ReLU
Convolution	3	16	1	256×256×8	256×256×16	ReLU
Convolution	3	3	1	256×256×16	256×256×3	ReLU

Table 2. Parameters of Network N3 and N4

Layer Type	Kernel size	# of kernels	stride	Input size	Output size	Activation function
Convolution	3	16	1	256×256×3	256×256×16	ReLU
Convolution	3	32	1	256×256×16	256×256×32	ReLU
Convolution	3	3	1	256×256×32	256×256×3	ReLU

The gray rectangle in Fig. 6 shows a shallow sequential network, which gets the features and tries to find a primary estimation of plate’s position. The architecture of this network is shown in the Fig. 7, and its parameters are shown in Table 3. The goal of this part is to reconstruct a primary plate’s position using a combination of low- and high-level features and the original image. It is necessary to mention that in Fig. 7, C1 and C2 show ordinary convolutional layers, MP3 and MP4 show Maxpooling layers, US2, US3, and US4 shows the upsampling layers. Finally, CT1 and CT2 show the transposed convolution layers.

Table 3. Parameters of N5 network

Layer Type	Kernel size	# of kernels	stride	Input size	Output size	Activation function
Maxpooling	4	-	1	64×64×3	16×16×3	-
Convolution	3	256	1	16×16×3	16×16×256	ReLU
Maxpooling	2	-	1	16×16×256	8×8×256	-
Convolution	3	512	1	8×8×256	8×8×512	ReLU
Upsampling	2	-	1	8×8×512	16×16×512	-
Transposed Convolution	3	128	1	16×16×512	16×16×128	ReLU
Upsampling	2	-	1	16×16×128	32×32×128	-
Transposed Convolution	3	64	1	32×32×128	32×32×64	ReLU
Upsampling	4	-	1	32×32×64	128×128×64	-

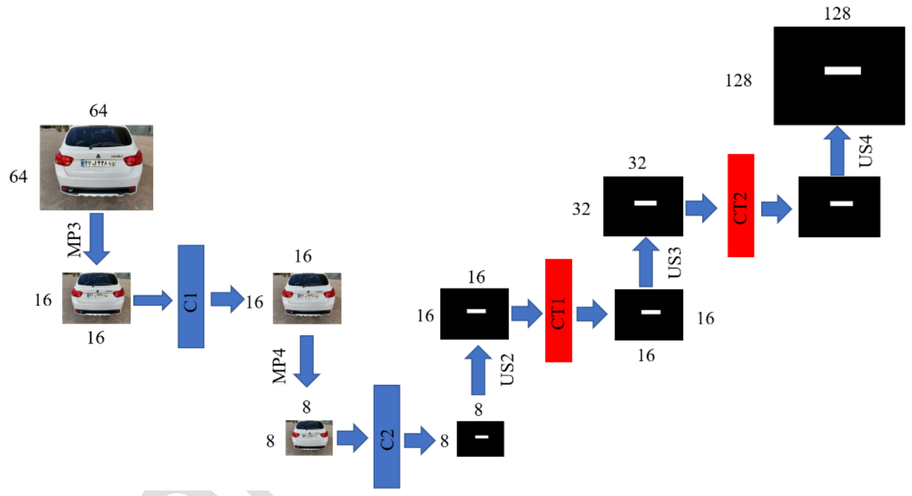


Figure 7: Architecture of the proposed N5 convolutional neural network

As it is stated before, the third part of the PPDN tries to find the accurate place of the car license plate. Red rectangles in the Fig. 6, show transposed convolutional neural networks that try to produce the accurate plate’s position mask from a primary plate’ position mask. Parameters of N6 and N7 are shown in Table 4 and parameters of N8 and N9 are shown in Table 5. These networks perform the inverse of the N1 to N4. The dark red rectangles in Fig. 6 shows concatenation layers which only put the feature maps on each other as a stack. The purple rectangle in Fig. 6 (US1 layer) shows the upsampling layer with the step of 2. Dark pink rectangle in Fig. 6 shows a two-layer sequential network whose parameters are depicted in Table 6.

Table 4. Parameters of Network N6 and N7

Layer Type	Kernel size	# of kernels	stride	Input size	Output size	Activation function
Transposed Convolution	3	64	1	128×128×64	128×128×64	ReLU
Transposed Convolution	3	32	1	128×128×64	128×128×32	ReLU
Transposed Convolution	3	16	1	128×128×32	128×128×16	ReLU

Table 5. Parameters of Network N8 and N9

Layer Type	Kernel size	# of kernels	stride	Input size	Output size	Activation function
Transposed Convolution	3	32	1	256×256×96	256×256×32	ReLU
Transposed Convolution	3	16	1	256×256×32	256×256×16	ReLU
Transposed Convolution	3	8	1	256×256×16	256×256×8	ReLU

Table 6. Parameters of Network N10

Layer Type	Kernel size	# of kernels	stride	Input size	Output size	Activation function
Transposed Convolution	3	10	1	256×256×112	256×256×10	ReLU
Transposed Convolution	3	1	1	256×256×10	256×256×1	ReLU

After the car license plate’s position is detected, the plate’s characters should be recognized, which is performed using two stages. First, characters on the license plate are separated and then they are recognized with two CNNs.

5. Character Recognition

To recognize the characters, a new algorithm is proposed to separate the characters. After separation, two convolutional neural networks are proposed to recognize digits and letters on the license plate.

5-1- Character separation

To separate the characters on the detected car license plate, a new algorithm is proposed in this paper. In this paper, to have better insight, the outputs of the proposed network in Section 4 are resized to 1024 × 1024-pixels images, but the proposed algorithm is implemented with 256 × 256-pixel images.

Then the output of PPDN, which is a 3-channel color image, transferred to gray image (it is necessary to mention that the output of the PPDN is a binary mask which should be multiplied by all 3 color channels of the original image to find the plate). The gray image is converted to a binary image utilizing a threshold which is set to 139 in this paper. Fig. 8 shows the process of converting the output of the proposed network in section 4 to a binary image. Next, the binary image is cropped to find the plate, only. To crop the image, only sections of binary image equivalent to white sections of the network output are selected, Fig. 9 shows the cropped upsampled image of detected plate. To the best of our knowledge, one of the best algorithms to separate the characters, is finding columns whose sum is maximum, as Fig. 10 shows. Since the characters are black and the background is white if summation of columns is calculated, there is a local maximum where there is no character. In addition, since there are 8 characters on the plate, 9 local maxima are needed to separate the characters. But as it can be seen from Fig. 10, it might be more than 9 local maxima in the plate of column summation of binary image. To overcome this problem, it is proposed to consider only maxima in the proper locations. As Fig. 11 shows, if the width of car license plate is y, each digit occupies 0.4y in length and the letter occupies 0.8y. Therefore, if only the maxima near these locations are considered, the characters can be separated as Fig. 12 shows.

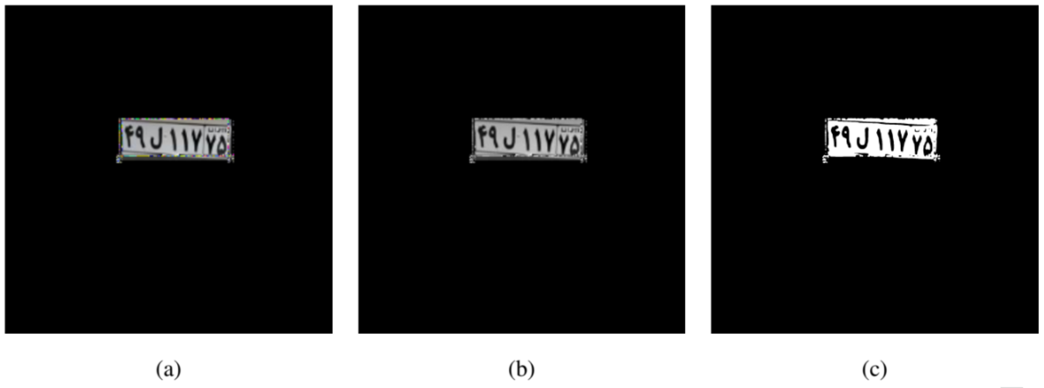


Figure 8: Converting Network output to binary image (a) Output of the plate position detection network (b) gray image (c) binary image

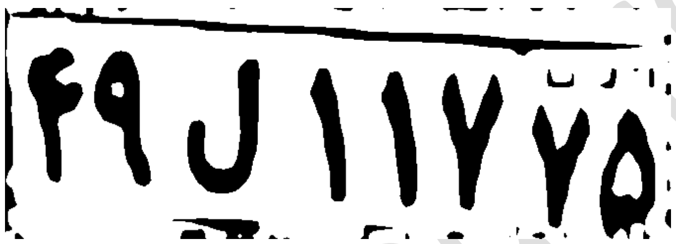


Figure 9: Cropped image of binary plate

After the characters are separated, they should be recognized to recognize the car license plate. To this goal two convolutional neural networks are proposed in the following sections.

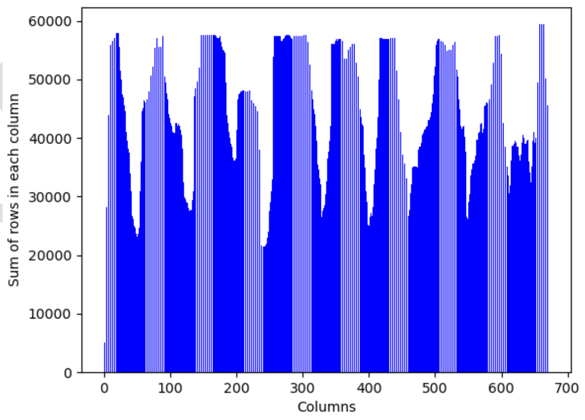


Figure 10: Histogram (sum) of columns of the cropped image of plate

10.22060/EEJ.2024.23296.5601

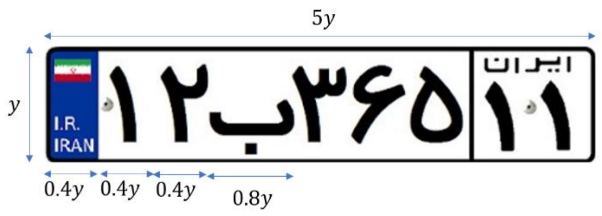


Figure 11: Approximate locations in car license plate

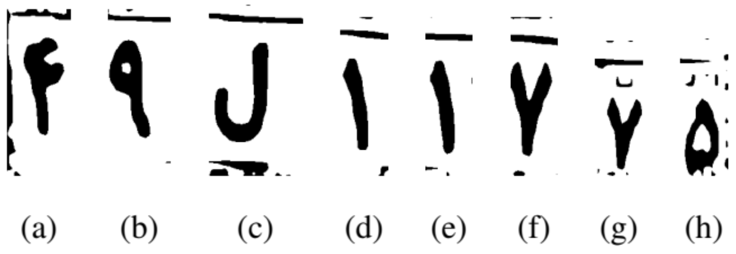


Figure 12: Separated characters for fig. 9

5-2- Digit Recognition Network (DRN)

In this section, a new convolutional neural network is proposed to recognize the separated digits on the license plate, called DRN. The proposed architecture is shown in Fig. 13. As Fig. 13 shows the input to the network is the one of separated digits of the previous section and the output is a digit between 0 to 9. Table 7 shows the parameters of DRN. Layers DN1 to DN10 in Fig. 13 are dilated convolution instead of ordinary convolutional layers. Since larger features can be detected using dilated convolution layers with the same number of parameters as ordinary convolution layers and therefore the more accuracy can be obtained. DA1 layer which is a concatenation layer tries to summarize all features for the next layers. The layers DD1 and DD3 operate as classifier. In addition, after layer DD1, a Dropout layer with parameter 0.2 and after DD2 a Dropout layer with parameter 0.3 is considered to avoid overfitting. Also, the activation function of last layer of DRN is softmax, to produce a probability distribution on the possible outputs.

10.22060/EEJ.2024.23296.5601

Table 7. Parameters of digit recognition network (DRN)

Layer Name	Layer Type	Kernel size	# of kernels	stride	Input size	Output size	Activation function	Dilation
DN1	Convolution	3	16	1	24×28×1	24×28×16	ReLU	1
DN2	Convolution	3	32	1	24×28×16	24×28×32	ReLU	1
DN3	Convolution	3	64	1	24×28×32	24×28×64	ReLU	1
DN4	Convolution	3	128	1	24×28×64	24×28×128	ReLU	1
DN5	Convolution	3	16	1	24×28×1	24×28×16	ReLU	2
DN6	Convolution	3	32	1	24×28×16	24×28×32	ReLU	2
DN7	Convolution	3	64	1	24×28×32	24×28×64	ReLU	2
DN8	Convolution	3	16	1	24×28×1	24×28×16	ReLU	3
DN9	Convolution	3	32	1	24×28×16	24×28×32	ReLU	3
DN10	Convolution	3	64	1	24×28×32	24×28×64	ReLU	3
DA1	Concatenation	-	-	-	Output of DN4, DN7 and DN10	24×28×257	-	-
DN11	Convolution	3	64	2	24×28×257	11×13×64	ReLU	1
DN12	Convolution	3	128	2	11×13×64	4×5×128	ReLU	1
DN13	Convolution	3	256	2	4×5×128	1×1×256	ReLU	1
Flatten	Flatten	-	-	-	1×1×256	256×1	-	-
DD1	Dense	500	1	1	256×1	500×1	ReLU	-
DD2	Dense	100	1	1	500×1	100×1	ReLU	-
DD3	Dense	10	1	1	100×1	10×1	softmax	-

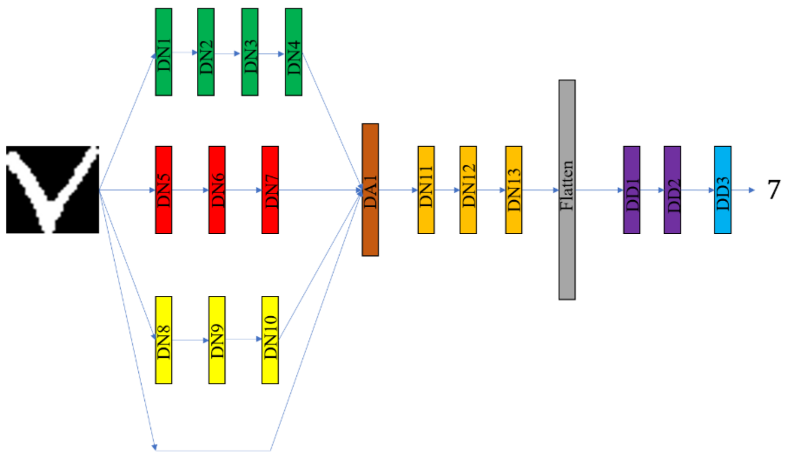


Figure 13: Architecture of the proposed network to digit recognize

The DRN is trained on two datasets: Persian handwritten digit dataset (HODA dataset) [30] and Arabic Handwritten digit dataset [31] to be more reliable. The trained DRN is tested on the separated digits of plates. The HODA dataset includes handwritten digits written by B.Sc. student in Iran and they are scanned at resolution of 200 Dot per Inch (DPI). The size of image of this dataset are variable. The HODA dataset includes 102352 samples from them 60000 and 20000 samples are utilized for training and testing, respectively. There are also 22352 samples which can be utilized for evaluation, for more information on HODA dataset, interested readers are referred to [30]. Hoda dataset includes images with different sizes. To find the proper size of network's input, rows and columns of all images are added, respectively; and then it is divided by the number of all images, this leads to 24 rows and 28 columns. Therefore, to have images with the same size, all images are resized to (24, 28) and saved. Next, to normalize the data, the type of each value in each image matrix is changed to float32 and then is divided by the maximum value of that image (255). Finally, all images are assigned to a variable with size of (60000, 24, 28, 1) for train data and (20000, 24, 28, 1) for test data.

Also, Arabic Handwritten digit dataset includes 60000 training images and 10000 test images. All images are in the same size (28,28). To have the same size with HODA, all samples are resized to (24, 28), for more information, interested reader is referred to [31].

Fig. 14 shows the training accuracy of DRN. As Fig. 14 shows, the network reaches to 100 % accuracy for trained data and as Fig. 15 shows the proposed network leads to validation accuracy higher than 99.60% in only 500 epochs. In addition, DRN reaches to 99.51% for separated digits on the 127 unseen images.

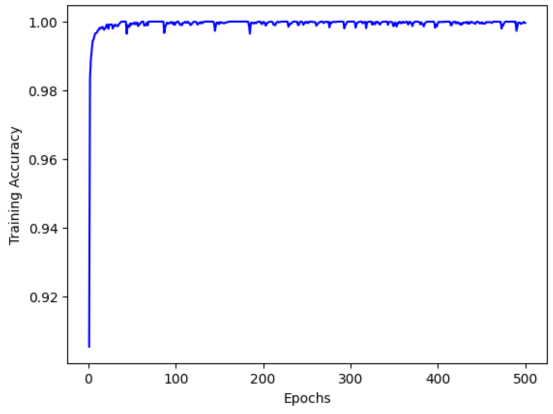


Figure 14: Training accuracy of DRN

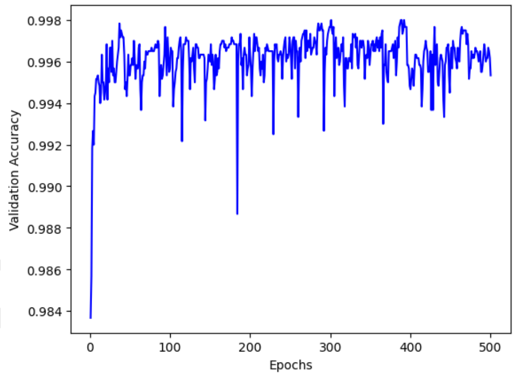


Figure 15: Validation accuracy of DRN

5-3- Letter Recognition Network (LRN)

In this section, a new sequential convolutional neural network is proposed to recognize the letter on the car license plate, called LRN. Table 8 shows the parameters of the LRN. The LRN is trained on the dataset in [32] and tested by the separated letters on the plate of 127 unseen images. Figs. 16 and 17 show the training accuracy and loss, respectively. As Figs. 16 and 17 shows the training accuracy tends to 100% and the network tries to eliminate the loss in 100 epochs which shows that the network is trained properly. The LRN' input size is 32 × 32, which means the separated character from section 5-1 is resized to 32 × 32. Also, the size of output of the network is 18, which is equal to number of letters and signs on the Iranian plates as explained in [32].

Table 8. Parameters of letter recognition network (LRN)

Layer Type	Kernel size	# of kernels	stride	Input size	Output size	Activation function
Convolution	3	32	1	32×32×1	30×30×32	ReLU
Convolution	3	64	1	30×30×32	28×28×64	ReLU
Maxpooling	2	-	1	28×28×64	14×14×64	-
Convolution	3	128	1	14×14×64	12×12×128	ReLU
Maxpooling	2	-	1	12×12×128	6×6×128	-
Convolution	3	128	1	6×6×128	4×4×128	ReLU
Maxpooling	2	-	1	4×4×128	2×2×128	-
Flatten	-	-	-	2×2×128	1×512	-
Dense	-	512	1	1×512	1×512	ReLU
Dense	-	18	1	1×512	1×18	softmax

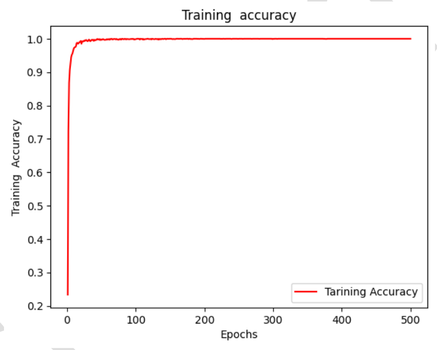


Figure 16: Training accuracy of the proposed network for letter detection

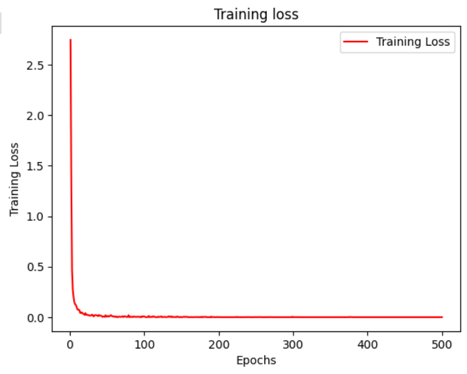


Figure 17: Training loss of the proposed network for letter detection

6. Algorithm to Recognize Car License Plate

After explaining all required steps to car license plate recognition, an algorithm is needed to use these steps to recognize a car license plate. First, an image from a scene including a car is taken. The image is resized to 256×256, then it is fed to PPDN. Next, the output and input of PPDN are multiplied to obtain the license plate of the car. Next, characters on the plate are separated. The third separated section is fed into LRN and seven other sections are fed to DRN. Finally, the results of the digit and letter recognition networks are gathered in proper order to recognize the car license plate.

7. Simulation Results

In this section, first, the hyperparameters of the proposed convolutional neural networks are discussed. Next, some experimental results are presented. Finally, we compare the proposed algorithm with several state-of-the-art algorithms.

7-1- Hyperparameters of PPDN and Discussion

The PPDN, proposed in Section 4, is trained on the combined dataset described in Section 3 over 3000 epochs using Google Colab with a Tesla T4 GPU. The network comprises 2,261,281 trainable parameters and is trained in multiple steps. In each step, the model and its weights are saved, allowing them to be loaded and reused in subsequent steps. The ’Adam’ algorithm is employed as the optimizer with a variable learning rate, initially set at 0.001 and decreased by a factor of 10 every 1000 epochs. The ’categorical cross-entropy’ function is used as the loss function, and a batch size of 5 is utilized during training. The training accuracy achieved is 99.86%, while the validation accuracy is 99.75%.

The trained network is evaluated on 127 images that are not part of the training or validation datasets (some of these images are depicted in Fig. 21), resulting in an average accuracy of 99.53%, as defined in Eq. (2). Additionally, the recall rate is calculated at 98.91%, as detailed in Eq. (1). It is important to note that a sample is considered to have incorrect plate recognition if the intersection of the detected plate’s position and the ground truth plate’s position is less than 90%.

$$\mathrm{Precission} = \frac{number\ of\ \ true\ detected\ pixels\ by\ \ the\ network\ \ as\ the\ place\ of\ \ plate}{Number\ \ of\ \ detected\ pixels\ by\ \ the\ network\ \ as\ the\ place\ of\ \ plate} \quad (1)$$

10.22060/EEJ.2024.23296.5601

$$\mathrm{Re}call = \frac{number\ of\ \ true\ detected\ pixels\ by\ the\ network\ as\ the\ place\ of\ \ plate}{Number\ of\ \ pixels\ in\ the\ ground-truth} \qquad (2)$$

Fig. 18 displays several images fed into the PPDN, along with the corresponding input and output results of the network from different cities. This figure illustrates the input-output multiplication to provide better insight into the network's performance. As shown in Fig. 18, the network accurately identifies the location of the license plate. It is important to note that the implementation time for detecting the position of the car license plate is approximately 16 milliseconds. Additionally, the separation algorithm averages 8 milliseconds, resulting in a total processing time of less than 24 milliseconds for each image in Fig. 18.

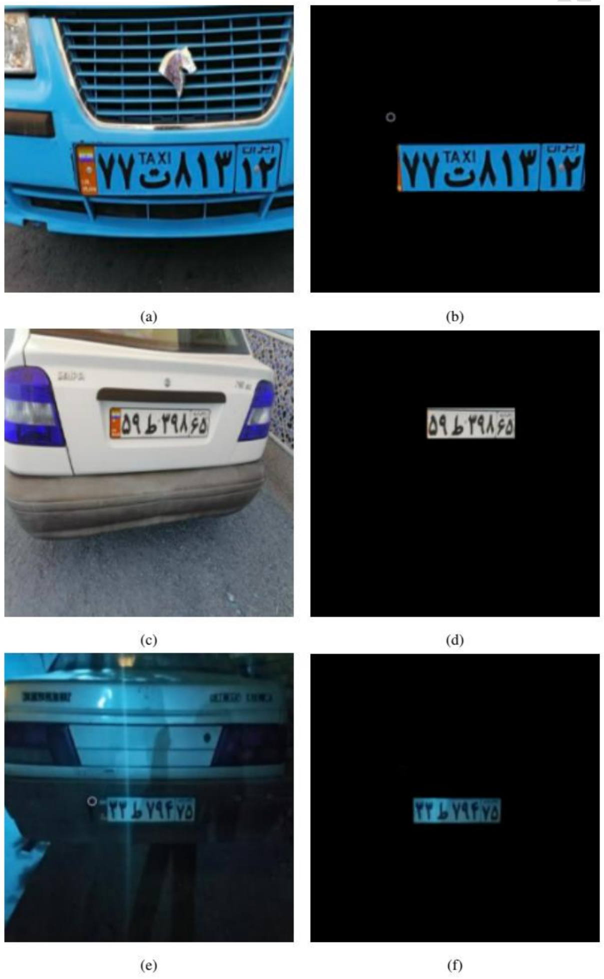


Figure 18: (a),(c) and (e) input to the network (b),(d) and (f) multiplication of input and output of the network

Fig. 19 presents an example of the network's improper performance. As depicted, while the proposed network successfully identifies the location of the car license plate, it also incorrectly labels other regions as potential license plate positions. Furthermore, the last digit of the car license plate is not fully detected.

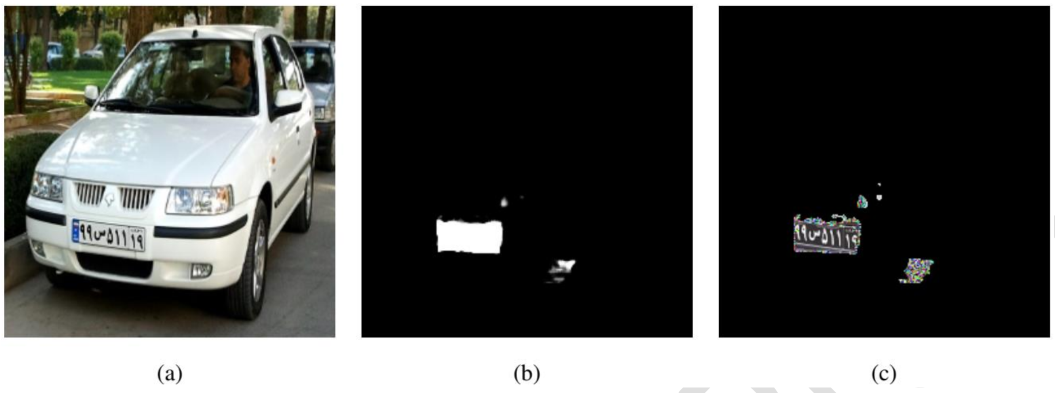


Figure 19: An example that the proposed network doesn't have proper performance (a) input to the network (b) output of the network (c) multiplication of input and output of the proposed network

7-2- Hyperparameters of DRN and Exploration

The digit recognition network, outlined in Section 5-2, is trained on the HODA dataset [30] and [31] for 500 epochs using Google Colab with a Tesla T4 GPU. The proposed network consists of 848,666 trainable parameters. The 'Adam' optimizer, with a learning rate of 0.0001, is employed to optimize the network, while 'categorical cross-entropy' is used as the loss function. The batch size for training is set to 256, and the validation split is configured to 0.1, meaning that 10% of the training data is allocated for validation.

Fig. 14 illustrates the accuracy of the DRN throughout the training process. As shown, the DRN achieves a training accuracy of 1.0000 (100%). Fig. 15 highlights the validation accuracy during training, where the DRN attains an average validation accuracy of 99.6% and a test accuracy of 99.51%. Fig. 20 presents examples of instances where the network produces incorrect predictions. However, it is important to note that the proposed network generates fewer erroneous results for car license plate digit detection because the characters on car plates are not handwritten and do not exhibit variations in font or style.

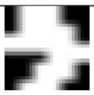
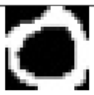
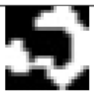
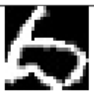
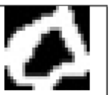
handwritten digit					
True Label	0	0	5	5	5
Detected label	3	5	0	4	0

Figure 20: Samples of false recognition by the proposed digit detection network

The model takes approximately 14 milliseconds to predict a single digit. Given that an Iranian car license plate contains 7 digits, the total time required to predict all digits on a car license plate is less than 100 milliseconds (0.1 seconds).

7-3- Hyperparameters of LRN and Discussion

The letter recognition network, presented in Section 5-3, is trained on the dataset from [32] for 500 epochs using Google Colab with a Tesla T4 GPU. The LRN comprises 512,722 trainable parameters. The 'RMSprop' optimizer, with a learning rate of 0.0001, is utilized for network optimization, and 'categorical cross-entropy' is employed as the loss function. The batch size during training is set to 20, along with a validation split of 0.1, indicating that 10% of the training data is reserved for validation.

Fig. 16 illustrates the accuracy of the LRN during the training process, demonstrating that the network achieves a training accuracy of 1.0000 (100%). Additionally, Fig. 17 shows the loss value of the LRN throughout training, indicating that the loss tends toward zero. The validation accuracy is measured at 99.45%, while the test accuracy is recorded at 99.05%. Furthermore, the LRN takes approximately 11 milliseconds to predict a single letter.

7-4- Evaluation of the Proposed Algorithm

To evaluate the accuracy of the proposed algorithm, we collected 127 images of cars under various lighting conditions using a Xiaomi POCO X3 NFC. We applied the proposed method to these images. Fig. 21 displays some of the collected data across different lighting conditions, as well as the results of applying the proposed algorithm. In Figs. 21-(a) to 21-(e), the lighting conditions vary, while Fig. 21-(f) shows a different plate color. As observed, the proposed algorithm performs effectively across these conditions. It is important to note that a sample is considered an incorrect prediction if the intersection between the detected plate position and the ground truth plate position is less than 90%. To calculate the accuracy, we divide the number of correctly recognized plates by the total number of plates (127). The resulting accuracy is 99.21%, indicating that the proposed algorithm correctly predicted 126 out of 127 samples.

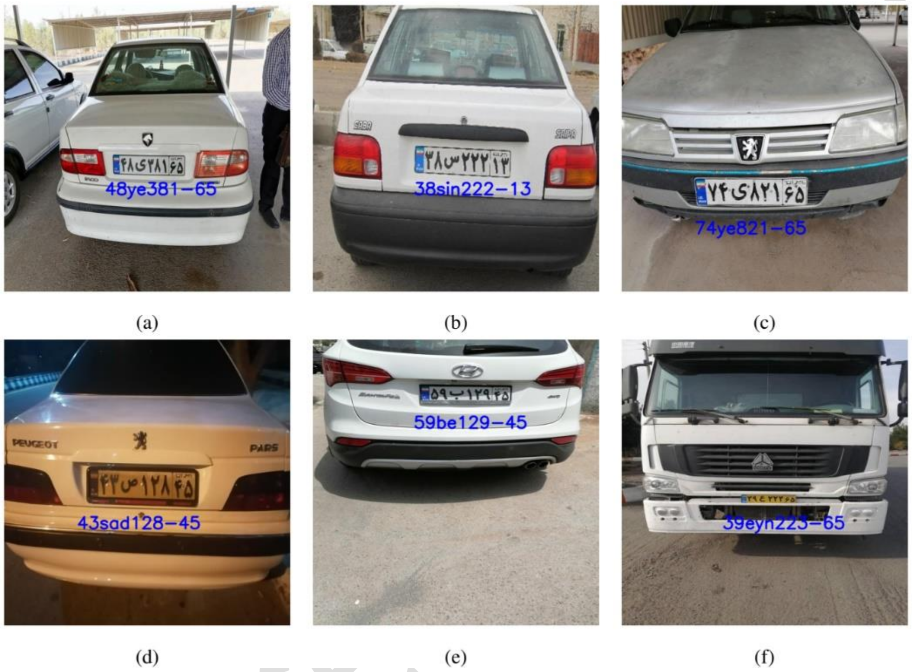


Figure 21: Some exprimental results of the proposed algorithm: (a) morning light in carport (b) evening (c) evening in carport (d) night (e) noon (f) plate with different color

To evaluate the computational complexity, the implementation time of the proposed algorithm, including the processing time of PPDN, DRN, LRN, and character separation, was compared with state-of-the-art algorithms. By adding the implementation time of the proposed networks and character separation algorithm, the proposed algorithm requires an average of 0.133 seconds to accurately predict an Iranian car license plate. This demonstrates that the proposed algorithm is sufficiently fast for online traffic control applications.

7-5- Comparing the proposed algorithm with the state-of-the-art algorithms

To ensure a fair comparison, the proposed algorithm was compared with the algorithms presented in [13], [17], [16], [23], [26], [20], [32], and [24]. All algorithms were simulated and tested on a Lenovo Legion 7 laptop equipped with an Intel(R) Core(TM) i7-10750H CPU @ 2.60GHz, 32.0 GB of RAM, and an Nvidia GeForce RTX 2070 Max-Q design GPU. Also, this section compares the proposed algorithm with state-of-the-art algorithms across six criteria: accuracy, precision, recall, F1-score, mAP, and implementation time.

Table 9 presents the plate recognition, plate detection, digit recognition, letter recognition accuracy, and runtime of the aforementioned algorithms. The first, second, and third highest values are highlighted in red, blue, and green, respectively. Note that plate position detection accuracy is not shown for algorithms that directly identify characters without prior plate localization. As evident from the table, the proposed algorithm achieves the highest accuracy in all detection and recognition processes. Moreover, its runtime of approximately 133 milliseconds ranks as the third fastest among the compared algorithms to the best of our knowledge. However, it's important to note that the proposed algorithm's performance deteriorates significantly when dealing with plates rotated relative to the horizon.

Table 9. Comparing accuracy of the proposed method with the state-of-the-art

Algorithm	Plate recognition	Plate position detection	Digit detection	Letter detection	Run time
Ref. [13]	99.18	-	89.91	79.83	238.89
Ref. [17]	98.70	98.08	87.54	98.66	230
Ref. [16]	95.03	97.60	97.7	97.7	121.12
Ref. [23]	96.7	-	97	97	67.89
Ref. [25]	90.65	87.81	87.22	87.22	435
Ref. [20]	94.19	-	95.2	95.8	414
Ref. [32]	92.23	-	96.57	98.29	461
Ref. [24]	97.9	96.2	94.5	94.5	4437
Ref. [26]	998.20	98.50	99.40	99.02	5612
Ref. [27]	75.14	87.82	87.22	86.16	435
Proposed	99.21	99.53	99.51	99.05	133

Table 10 presents the precision for the proposed and state-of-the-art algorithms. Precision measures the accuracy of predictions. As shown in the table, the proposed networks for plate position detection, digit recognition, and letter recognition outperform existing networks. While the proposed approach achieves the second-best overall performance in car plate recognition, as indicated in Table 10, it demonstrates superior accuracy in the individual components.

Table 10. Comparing precision of the proposed method with the state-of-the-art

Algorithm	Plate recognition	Plate position detection	Digit detection	Letter detection
Ref. [13]	99.05	-	90.01	80.22
Ref. [17]	99.8	99.01	98.14	98.72
Ref. [16]	93.85	96.43	97.9	97.9
Ref. [23]	95.9	-	96.81	96.23
Ref. [25]	87.81	87.22	86.92	86.43
Ref. [20]	93.10	-	94.72	94.38
Ref. [32]	91.17	-	95.41	98.09
Ref. [26]	97.52	97.21	97.93	98.26
Ref. [27]	91.51	90.16	89.21	89.42
Ref. [24]	97.54	95.12	96.57	98.29
Proposed	99.13	99.37	99.16	99.21

Table 11 compares the recall for the proposed method and state-of-the-art algorithms. Recall measures the algorithm’s ability to identify positive samples. The proposed method excels in plate position detection and letter recognition, and it also achieves the second-best results in digit recognition and overall plate recognition. These results indicate that the proposed method effectively estimates the true plate position and accurately identifies characters in nearly all samples.

Table 11. Comparing recall of the proposed method with the state-of-the-art

Algorithm	Plate recognition	Plate position detection	Digit detection	Letter detection
Ref. [13]	99.11	-	87.76	75.13
Ref. [17]	98.26	97.81	85.10	97.8
Ref. [16]	94.30	96.62	99.4	98.90
Ref. [23]	95.21	-	94.3	94.21
Ref. [25]	87.74	85.14	86.17	87.01
Ref. [20]	93.29	-	94.11	93.87
Ref. [32]	91.24	-	95.31	97.65
Ref. [24]	95.98	95.12	98.03	98.12
Ref. [26]	97.22	96.49	98.89	98.23
Ref. [27]	83.69	89.53	88.47	86.04
Proposed	98.91	98.25	98.44	98.95

To assess the reliability of the results across different samples within the datasets, Table 12 presents the F1-score for each class. The F1-score, a harmonic mean of precision and recall, provides a comprehensive measure of model performance. As shown in Table 12, the proposed approach outperforms all methods in plate position detection and both digit and character recognition. In terms of overall plate recognition, the proposed method achieves the second-best F1-score. These results demonstrate the robustness and effectiveness of the proposed approach across diverse samples.

Table 12. Comparing F1-score of the proposed method with the state-of-the-art

Algorithm	Plate recognition	Plate position detection	Digit detection	Letter detection
Ref. [13]	99.08	-	88.87	77.59
Ref. [17]	99.02	98.40	91.16	98.26
Ref. [16]	94.07	96.52	98.64	98.4
Ref. [23]	95.55	-	95.54	95.21
Ref. [25]	87.78	86.17	86.54	86.72
Ref. [20]	93.19	-	94.41	94.12
Ref. [32]	91.21	-	95.36	97.86
Ref. [24]	96.75	95.12	97.29	98.20
Ref. [26]	97.36	96.84	98.40	98.24
Ref. [27]	87.42	89.84	88.83	87.69
Proposed	99.02	98.81	98.80	99.08

Table 13 presents the mean Average Precision (mAP) for the proposed and state-of-the-art algorithms. The proposed method demonstrates superior performance in plate position detection and digit recognition, indicating that the proposed networks outperform existing methods. This improvement is attributed to the two-section architecture of the plate position detection network, which enables accurate plate position estimation, and the deep design of the digit recognition network, which facilitates the identification of larger and higher-level features. While the proposed approach achieves the second-best results in letter recognition and overall plate recognition, it still demonstrates strong performance and reliability.

The proposed algorithm was compared with several state-of-the-art algorithms, as illustrated in Fig. 22. This figure showcases an image captured in low-light night conditions, where a shadow divides the license plate into a dark upper half and a lighter lower half. Such challenging conditions significantly hinder character recognition. As the results demonstrate, only the proposed method successfully detected and recognized the car license plate, highlighting its effectiveness in handling adverse lighting conditions. Finally, as evident from Tables 9 to 13, the proposed method consistently ranks among the top performers in plate recognition, plate position detection, and both digit and letter recognition. Additionally, Table 9 reveals that the proposed method has the third-highest speed. Based on these findings, it can be concluded that the proposed method offers a compelling balance of performance and efficiency compared to other state-of-the-art approaches.

Table 13. Comparing mAP of the proposed method with the state-of-the-art

Algorithm	Plate recognition	Plate position detection	Digit detection	Letter detection
Ref. [13]	97.12	-	85.46	73.87
Ref. [17]	99.6	98.12	99.02	99.78
Ref. [16]	93.02	94.13	93.76	92.11
Ref. [23]	90.28	-	91.76	91.5
Ref. [26]	75.14	78.99	79.20	80.05
Ref. [20]	90.10	-	91.9	90.79
Ref. [32]	93.51	-	95.72	93.49
Ref. [24]	83.0	85.1	85.9	86.1
Proposed	99.49	99.08	99.2	98.96

8. Conclusion

In this paper, a new dataset is collected and then is combined with an existing dataset. All samples in the combined dataset are labeled in such a way correct position of car license plate is shown. Next, a new convolutional neural network is proposed and trained to find the position of a car license plate. Then, the characters on the found plate are separated. After separation, each digit is recognized by a proposed convolutional neural network for digit recognition (DRN), and also, the letter is recognized by another proposed convolutional neural network (LRN). Finally, all the results are gathered to recognize the plate. The proposed algorithm is fast in such a way that it needs about 0.133 second totally. As the simulation results confirm the accuracy of the proposed algorithm is 99.21% on unseen data.

In future, we try to label the collected dataset in such a way that characters of plates are shown in an image directly; and we design a new CNN to find all characters directly which means that the future work does not need to character separation. Consequently, the speed of detection can be increased.

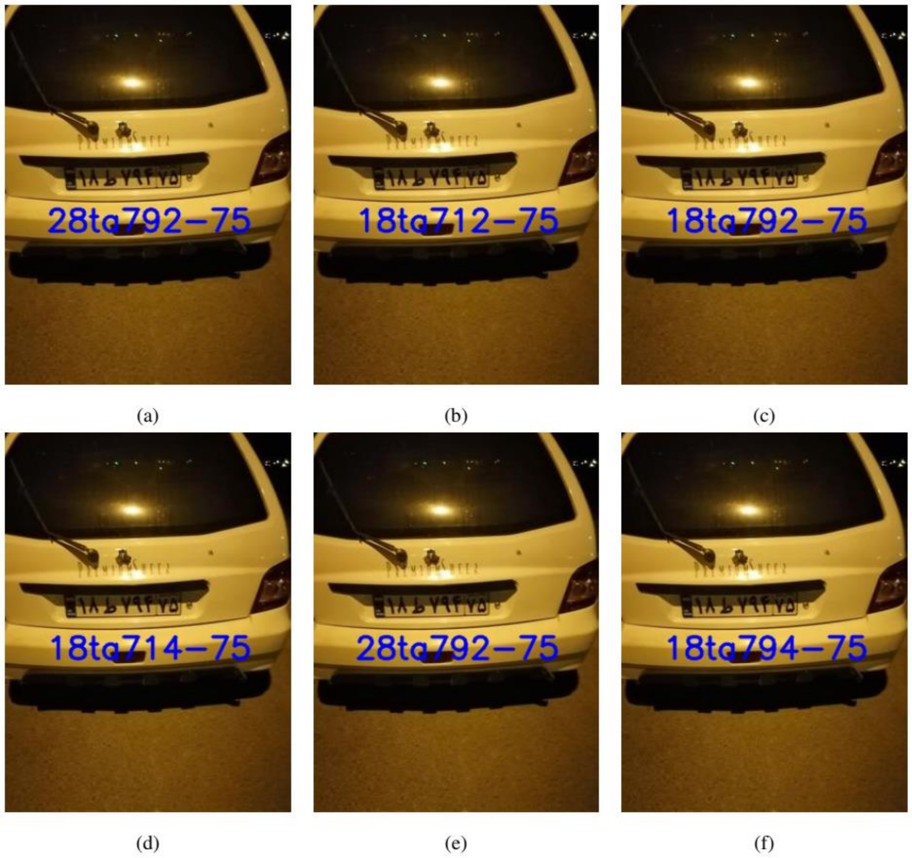


Figure 22: Comparison the result of the proposed algorithm with several of state-of-the-art algorithms: (a) ref. [23] (b) ref. [24] (c) ref. [26] (d) ref. [27] (e) ref. [32] (f) Proposed algorithm

References

[1] A.H. Ashtari, M.J. Nordin, M. Fathy, An Iranian license plate recognition system based on color features, IEEE transactions on intelligent transportation systems, 15(4) (2014) 1690-1705.
[2] W. Jia, H. Zhang, X. He, M. Piccardi, Mean shift for accurate license plate localization, in: Proceedings. 2005 IEEE Intelligent Transportation Systems, 2005., IEEE, 2005, pp. 566-571.
[3] X. Shi, W. Zhao, Y. Shen, Automatic license plate recognition system based on color image processing, in: International conference on computational science and its applications, Springer, 2005, pp. 1159-1168.
[4] F. Wang, L. Man, B. Wang, Y. Xiao, W. Pan, X. Lu, Fuzzy-based algorithm for color recognition of license plates, Pattern Recognition Letters, 29(7) (2008) 1007-1020.
[5] V. Abolghasemi, A. Ahmadyfard, An edge-based color-aided method for license plate detection, Image and Vision Computing, 27(8) (2009) 1134-1142.

10.22060/EEJ.2024.23296.5601

[6] F. Ramazankhani, M. Yazdian-Dehkordi, Iranian license plate detection using cascade classifier, in: 2019 27th Iranian conference on electrical engineering (ICEE), IEEE, 2019, pp. 1860-1863.
[7] F. Ramazankhani, M. Yazdian-Dehkordi, Iranian Vehicle License Plate Detection based on Cascade Classifier, Signal and Data Processing, 18(3) (2021) 77-90.
[8] S.S. Tabrizi, N. Cavus, A hybrid KNN-SVM model for Iranian license plate recognition, Procedia Computer Science, 102 (2016) 588-594.
[9] S. Poursiyah, H. Salami, M.R. Mohebbi, H. Tabatabaee, An Effective and Fast Iranian License Plate Detection Using Statistical and Geometrical Approaches, Advances in Science and Technology. Research Journal, 12(4) (2018).
[10] M.A. Khan, M. Sharif, M.Y. Javed, T. Akram, M. Yasmin, T. Saba, License number plate recognition system using entropy-based features selection approach with SVM, IET Image Processing, 12(2) (2018) 200-209.
[11] Y. Wang, Z.-P. Bian, Y. Zhou, L.-P. Chau, Rethinking and designing a high-performing automatic license plate recognition approach, IEEE Transactions on Intelligent Transportation Systems, 23(7) (2021) 8868-8880.
[12] S.M. Silva, C.R. Jung, License plate detection and recognition in unconstrained scenarios, in: Proceedings of the European conference on computer vision (ECCV), 2018, pp. 580-596.
[13] A.R. Rashtehroudi, A. Shahbahrami, A. Akoushideh, Iranian license plate recognition using deep learning, in: 2020 international conference on machine vision and image processing (MVIP), IEEE, 2020, pp. 1-5.
[14] S.-L. Chen, C. Yang, J.-W. Ma, F. Chen, X.-C. Yin, Simultaneous end-to-end vehicle and license plate detection with multi-branch attention neural network, IEEE Transactions on Intelligent Transportation Systems, 21(9) (2019) 3686-3695.
[15] R. Laroca, E. Severo, L.A. Zanlorensi, L.S. Oliveira, G.R. Gonçalves, W.R. Schwartz, D. Menotti, A robust real-time automatic license plate recognition based on the YOLO detector, in: 2018 international joint conference on neural networks (ijcnn), IEEE, 2018, pp. 1-10.
[16] A. Tourani, A. Shahbahrami, S. Soroori, S. Khazaee, C.Y. Suen, A robust deep learning approach for automatic iranian vehicle license plate detection and recognition for surveillance systems, IEEE Access, 8 (2020) 201317-201330.
[17] M. Shahidi Zandi, R. Rajabi, Deep learning based framework for Iranian license plate detection and recognition, Multimedia Tools and Applications, 81(11) (2022) 15841-15858.
[18] H. Gholamalinejad, H. Khosravi, Irvd: A large-scale dataset for classification of iranian vehicles in urban streets, Journal of AI and Data Mining, 9(1) (2021) 1-9.
[19] G.-S. Hsu, A. Ambikapathi, S.-L. Chung, C.-P. Su, Robust license plate detection in the wild, in: 2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), IEEE, 2017, pp. 1-6.
[20] S. Rakhshani, E. Rashedi, H. Nezamabadi-pour, License plate recognition using deep learning, Journal of Machine Vision and Image Processing, 6(1) (2019) 31-46.
[21] C. Xu, H. Zhang, W. Wang, J. Qiu, License plate recognition system based on deep learning, in: 2020 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA), IEEE, 2020, pp. 1300-1303.
[22] J. Hendryli, D.E. Herwindiati, Automatic license plate recognition for parking system using convolutional neural networks, in: 2020 International Conference on Information Management and Technology (ICIMTech), IEEE, 2020, pp. 71-74.
[23] A. Ebrahimi, A. Amirkhani, A. A Raie, M.R. Mosavi, Car license plate recognition using color features of Persian license plates, Journal of Advances in Computer Research, 6(4) (2015) 27-38.

[24] M. Rahmani, M. Sabaghian, S.M. Moghadami, M.M. Talaie, M. Naghibi, M.A. Keyvanrad, IR-LPR: A Large Scale Iranian License Plate Recognition Dataset, in: 2022 12th International Conference on Computer and Knowledge Engineering (ICCKE), IEEE, 2022, pp. 053-058.
[25] G. Karimi, Z. Ali Mohammadi, S. Najafi, S. Mousavi, D. Motiallah, Z. Azimifar, Real-Time Mobile Mixed-Character License Plate Recognition via Deep Learning Convolutional Neural Network, in: International Conference on Artificial Intelligence and Smart Vehicles, Springer, 2023, pp. 77-93.
[26] D. Giveki, O. Dalvand, H. Rastegar, Introducing a new dataset (Iranian Plate) for Iranian licence plate recognition, Journal of Machine Vision and Image Processing, 9(2) (2022) 81-95.
[27] S. Hatami, M. Sadedel, F. Jamali, Iranian License Plate Recognition Using a Reliable Deep Learning Approach, arXiv preprint arXiv:2305.02292, (2023).
[28] K. Branch, A novel morphological method for detection and recognition of vehicle license plates, American Journal of Applied Sciences, 6(12) (2009) 2066-2070.
[29] S.H.M. Kasaei, S.M.M. Kasaei, Extraction and recognition of the vehicle license plate for passing under outside environment, in: 2011 European Intelligence and Security Informatics Conference, IEEE, 2011, pp. 234-237.
[30] H. Khosravi, E. Kabir, Introducing a very large dataset of handwritten Farsi digits and a study on their varieties, Pattern recognition letters, 28(10) (2007) 1133-1141.
[31] A. El-Sawy, M. Loey, H. El-Bakry, Arabic handwritten characters recognition using convolutional neural network, WSEAS Transactions on Computer Research, 5(1) (2017) 11-19.
[32] A. Tourani, S. Soroori, A. Shahbahrami, A. Akoushideh, Iranis: A large-scale dataset of iranian vehicles license plate characters, in: 2021 5th International Conference on Pattern Recognition and Image Analysis (IPRIA), IEEE, 2021, pp. 1-5.