IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Revue d'Intelligence Artificielle
Vol. 37, No. 5, October, 2023, pp. 1331-1339
Journal homepage: http://iieta.org/journals/ria

Tesseract OpenCV Versus CNN: A Comparative Study on the Recognition of Unified Modern Iraqi License Plates

Suha Ahmed*, Akbas Ali, Ekhlas Naser

Department of Computer Science, The University of Technology, Baghdad 10066, Iraq

Corresponding Author Email: cs20.31@grad.uotechnology.edu.iq

https://doi.org/10.18280/ria.370526

Received: 25 June 2023
Revised: 25 July 2023
Accepted: 30 July 2023
Available online: 31 October 2023

Keywords:
convolution neural network, deep learning, license plate recognition, machine learning, tesseract

ABSTRACT

Various applications central to societal functioning, such as traffic control and parking management, are fundamentally rooted in License Plate Recognition (LPR). The type of license plate significantly impacts the effectiveness of these processes. This study focuses on the 2022 Unified Modern Iraqi license plates, which pose a unique challenge due to their recent design that incorporates the representation of governorate names with symbols. This new design introduces difficulties in accurately recognizing characters, leading to potential misinformation and unreliable applications. Furthermore, there is a dearth of recognition systems specifically tailored for these newly designed plates. In an attempt to surmount these hurdles, this paper introduces a comparative analysis of two models based on state-of-the-art machine and deep learning methods. The first model employs Tesseract by OpenCV for the recognition of characters on the detected plate, while the second model utilizes a nine-layer Convolutional Neural Network (CNN). The research contributes to the field by collating the plates into a dataset and recognizing them for the first time using these models. The results indicate a significant disparity in the performance of the two models, with the CNN model exhibiting superior accuracy in character recognition, surpassing 95.5%, while the Tesseract OpenCV model achieved a rate of merely 36%. This study underscores the potential of deep learning methods in augmenting license plate recognition systems, especially for novel designs like the 2022 Unified Modern Iraqi license plates.

1. INTRODUCTION

The license plate recognition (LPR) technology, which is derived from advanced intelligent transportation systems, plays a vital role in managing traffic, enhancing digital security surveillance, recognizing vehicles, and facilitating parking management in large cities [1-5]. The LPR technology is employed in various countries worldwide, particularly in smart cities, electronic garages, and mainly on roads, to capture data on speeding vehicles or track stolen cars. The license plate design for each country is created by its government and typically comprises a combination of letters, numbers, and words written in the country's language [6]. For instance, the license plates of new Iraqi vehicles are distinguished from the model that appeared in 2022 (the unified model) [7], where the Arabic and the Kurdish languages were excluded from the English language (the German model).

Until 2022, Iraqi license plates displayed the names of provinces as words. However, starting in 2022, the representation of regions on license plates was changed to pairs of symbols for each province, following the guidelines set by the Iraqi General Traffic Directorate, as shown in Figure 1.

The new Iraqi license plates pose a challenge to current LPR technologies. This challenge stems from the alteration in the design of certain license plate characters in forms different from their standard forms known in the standard dataset. Furthermore, the traditional displaying of the governorate's name has been substituted with symbols incorporated into the vehicle number itself. This departure from the established norm, where the actual name is replaced by symbolic representations, adds another layer of complexity to LPR systems.

Figure 1. The symbols used for car numbers in governorates of Iraq

Compounding this problem is the absence of a standardized dataset that could facilitate the seamless detection and identification of these new license plates. The absence of such a dataset impedes the efficiency and accuracy of LPR technologies when tasked with identifying and processing these novel license plates.

So in this research to solve those problems, two models were applied to recognize these plates. The first model was developed based on machine learning (ML) using the TESSERACT Open CV technique, and this is the first time that this technique is used to recognize the numbers and the governorate of the new Iraqi license plates. The second model is based on the latest methods of deep learning (DL) using CNN.

In this context, ML refers to the ability of a system to acquire knowledge and enhance its performance based on its past experiences [8]. On the other hand, DL Belongs to the category of machine learning that emphasizes learning successive layers of more complex data representations [9]. In particular, DL techniques have completely overshadowed other ML techniques since they produce better results.

It primarily serves the purpose of making informed decisions. but can also be employed in many other applications [3]. TESSERACT is a freely available and open-source optical character recognition (OCR) engine. Distributed under the Apache 2.0 license, it offers the capability to extract textual content from images, either directly or through an API for software developers. With extensive language support, Tesseract enables text extraction across multiple languages. Notably, it operates without a graphical user interface (GUI), focusing on providing powerful OCR functionality.

Py-Tesseract is a Python package that serves as a convenient interface for the Tesseract-OCR Engine. As a wrapper, it allows seamless integration with Tesseract functionalities within Python code. Additionally, Py-Tesseract is versatile enough to be utilized as an independent Tesseract invocation script. Leveraging the capabilities of the Pillow and Leptonica imaging libraries, Py-Tesseract can effectively process and extract text from a diverse range of image formats, such as jpeg, png, gif, BMP, tiff, and more [10].

The strength of the Tesseract OCR algorithm is its ability to perform real-time text recognition of captured license plate images, while its weakness is its sensitivity to different angles and character styles [11].

The second model introduced in this research uses the convolution neural network. One key advantage of utilizing neural networks is their ability to directly process raw pixel data and function as both feature extractors and classifiers. While deep learning approaches may require more computational resources compared to alternative methods such as template matching and statistical feature extraction, they typically offer improved accuracy in return [12]. Convolution-deep Neural networks find application in the recognition of visual patterns and types [13]. It is a revolution in computer vision applications to process multi-dimensional data, such as three two-dimensional arrays of pixels of the color image. Nowadays, CNNs are the most public deep neural architecture [14, 15]. Specifically, CNN is popular for license plate character recognition, and it is the most effective approach for implementing ALPR systems [3].

CNN's strength lies in playing a crucial role in improving the accuracy and efficiency of license plate recognition systems [16]. and from the deep learning techniques CNN has shown a promising result in achieving more efficient and optimal performance in LPR through her training with the training dataset used for the recognition of license plates in certain conditions, such as the non-standard design of the license plate, different environments, lighting, and weather conditions [17]. while the Weakness is Difficulty in recognizing multiple license plates within a single frame, especially when plates are in proximity or crowded areas. Inability to distinguish between the license plate area and vehicle grills, potentially leading to false positives or incorrect recognition of characters when obstructions [3].

The model encompasses various stages, including vehicle image acquisition, license plate detection, character segmentation from the license plate, and ultimately, character recognition. The detection of the license plate is a complex and challenging task that can have a direct impact on the effectiveness and precision of the subsequent steps [18]. In general, it is worth mentioning that the utilization of deep learning (DL) in automatic license plate recognition (ALPR) systems is currently limited, with only a few studies exploring its application in an end-to-end manner. Existing literature predominantly relies on manually engineered features such as HOG (Histogram of Oriented Gradients) and LBP (Local Binary Patterns). However, the data-driven features offered by DL techniques are typically more discriminative and have the potential to enhance ALPR performance [19].

The contribution of this research is represented by splitting and recognizing the new Iraqi license plates for the first time in two ways, depending on one of the machine learning methods using Tesseract and one of the deep learning methods using CNN, and then comparing the results between the two methods.

The rest of this paper is structured as follows: Section II presents some related works of this research. Section III represents the LPR System based on machine learning techniques in detail, and then in Section IV, the LPR system based on deep learning techniques is explained. The experiments and the results are given in Section V and discussed in Section VI. Finally, the conclusions are stated in Section VII.

2. RELATED WORKS

In this section, an overview of the current advancements in license plate detection and recognition is provided. this is divided into two main parts: license plate recognition utilizing machine learning (ML) techniques and license plate recognition utilizing deep learning (DL) methodologies. The literature review encompasses the approaches in both domains.

2.1 Related works based on machine learning (ML)

Several research works deal with recognizing license plates, the most important of which, based on machine learning, include the following:

- The license plate recognition system proposed in reference [20] is capable of giving results in challenging backgrounds and challenging scenarios. The LPR system initially converts the car image into grayscale and finds all closed shapes in the image that could contain license plate characters. The system then extracts the characters from the grayscale image using OCR technology. The system was tested with more than 1,000 images under different lighting conditions and positions. However, the accuracy obtained through experiments under different conditions did not exceed 72.5%.
- Goel et al. [21] aims to develop and implement an active vehicle identification system that recognizes a vehicle by its

license plate. Using different Open-CV image processing library capabilities, the system can locate the contour with four points inside the list and take that as the vehicle's license plate. The tesseract was used to recognize the characters on the license plate. The paper acknowledges the efficiency of license plate recognition but doesn't specify its accuracy. It suggests enhancing it with deep learning models. The method suits single-line characters but disregards multi-line plates, leading to errors with multiple plates or small images. The research lacks details about test data, plate types, and the countries involved.

- Suneetha and Mounika Raj [22] presented an Automatic Vehicle Number Plate Recognition (ANPR) system that aims to recognize vehicle license plates using image processing techniques and optical character recognition (OCR). The system employs the Open CV Python package to detect car license plates in a generated synthetic image dataset. The process involves several stages, including image preprocessing, plate detection, character segmentation, and character/number recognition using the PyTesseract package applied to the segmented license plate region.
- Mohd Ariffin [23] proposed a system to detect the vehicle's license plate and extract the characters in the detected license plate of the speeding vehicle. Specifically, the preprocessing uses some common sub-processes, which are the gray scaling process and specific filtering processes, namely the bilateral filtering, to detect the license plate from the car using the contour option in Open CV for detecting rectangular objects to find the plate numbers. The license plate recognition reads the plate information from the segmented image using the py-tesseract package to read characters from the image. The study compared the model's accuracy (86.36%) to an SVM model (66.67%) used in its related works using 20 images per character. The system's target country, type of license plate, or region wasn't specified.
- Sawalkar et al. [24] aimed to develop an Automatic Number Plate Recognition (ANPR) system specifically tailored to the unique variety of vehicle plates in Ghana. The system consisted of a four-block image preprocessing, encompassing plate localization and extraction, character segmentation, and character recognition. The implementation was written in C++ using the OpenCV library and relied on edge detection techniques for plate localization. Subsequently, the Tesseract OCR engine was utilized to identify the characters detected on the license plate. The modified ANPR system achieved a successful recognition rate of 60% for most of the distinct number plates, with an average processing time of approximately 0.2 seconds to complete the entire character image capture process.

2.2 Related works based on deep learning (DL)

There are several research works that deal with recognizing license plates, the most important of which, based on deep learning, include the following:

- Abedin et al. [25] developed a license plate recognition (LPR) system designed for the Bangla language to support intelligent vehicle management. The method comprises three main components: license plate detection (LPD), which relies on contour features of the characters in the lower part of the plate; license plate segmentation (LPS), responsible for extracting the essence set from the license plate; and license plate recognition (LPR), which employs a deep learning approach using a convolutional neural network. The proposed algorithm achieved a high success rate with 93% accuracy in license plate detection, 98% accuracy in character segmentation, and 98% accuracy in recognition. the research has some limitations such as using vehicles for all classes, night mode detection, and a real-time application.
- Marzuki et al. [26] put forth a proposal in their study an approach based on an enhanced Convolutional Neural Network (CNN) algorithm specifically designed for license plate recognition systems. The overall system consists of preprocessing, segmentation, and character recognition. the localization of the license plate was achieved through the utilization of the Sobel operator edge detection, morphological operations, and connected component analysis. For the CNN model employed, a four-layered architecture was implemented and was able to reach 94.6% out of 528 testing samples with a Lack of information on image condition variations in real-world scenarios.
- Alam et al. [27] suggested a system for the detection and identification of vehicle license plates, which are written in the Bengali language, in Bangladesh. This system consists of LPD and LPR. The number plate regions were extracted using the template matching method. Then, the segmentation of each character was performed. CNN was employed to extract character features to recognize license plate characters. Through evaluating experimental outcomes using 700 vehicle images, the trained CNN achieved an accuracy of 98.2% on the validation set and 98.1% on the testing set. The system faces real-world limitations, like difficulty capturing plates directly due to complex backgrounds, and challenges recognizing characters in large blurry images.
- Kaur et al. [3] presented an ALPR system that utilizes deep learning, specifically Convolutional Neural Networks (CNN), for character recognition. The proposed system consists of multiple stages, including data acquisition and pre-processing, feature extraction, Region of Interest (ROI) selection, and classification. The CNN model was employed to recognize the license plate data. Experimental results showcased the system's effectiveness, achieving an impressive recognition rate of 98.13% when evaluated on a dataset of 160 images. The system faces real-world limitations, like difficulty capturing plates directly due to complex backgrounds, and challenges recognizing characters in large blurry images.
- Abbass and Marhoon [6] applied a method to recognize the old license plates of Iraqi cars based on DL techniques of (CNN) using LP segmentation to segment numbers, letters, and words from the car license plate images, and then, they converted them into two databases that were used to train the two CNNs; Character and governorate recognition. The dataset comprises 2000 images of numbers and letters extracted through the LP stage's character segmentation .it was divided into a 70-30 ratio for train and test sets. These images form 22 classes, including numbers 0-9 and common Arabic letters found in Iraqi license plates. The testing involved 500 images. The proposed system demonstrated an overall success rate in all stages reached 97%.

3. COLLECTING THE DATASET

To test the two models, work has been done on a dataset of 180 real images of vehicles with new Iraqi plates for the year 2022, which were collected were gathered from various locales in Iraq, to guarantee a rich assortment of lighting conditions and environmental variables, it was collected randomly at diverse angles and distances to tackle these challenges in license plates recognition because this dataset is not available and they were recently issued. In particular, their issuance was currently limited to the northern governorates of Erbil, Dohuk, and Sulaymaniyah. Figure 2 shows a sample of the plates.

Figure 2. Sample of the new Iraqi license plate

To train Model 2, we used a dataset downloaded from the Kaggle dataset [28], which is commonly used for license plate character recognition. This dataset contains 1080 images with (28x28) pixels with 36 classes (A-Z and 0-9) that were extracted. These images are clear without any added noise. The width of the characters is almost the same. However, in a real-life scenario, the images may be thicker and sometimes very thin or with open or closed edges. A sample of the original dataset is shown in Figure 3 (a). In the training dataset of Figure 3 (b), more images were manually added, and this dataset contains 1100 images with the same 36 classes. 20 more images were added along with the existing images after resizing them. These newly added images were manually cropped from a segmented character in the segmentation stage, and the pixels for some images were reduced in all 36 classes. As a result, Model 2 was trained on dataset (b).

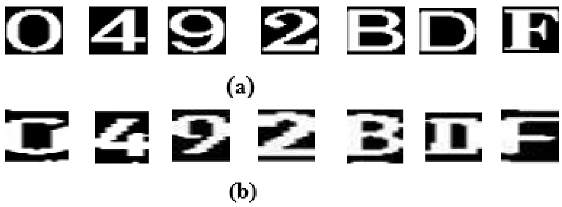


Figure 3. The sample dataset (a) original dataset (b) training dataset

4. METHODOLOGY OF LICENSE PLATE RECOGNITION (LPR) OF MODEL 1

The vehicle's license plate is recognized using various features of the image processing library Open-CV while recognizing the text on the license plate is done using a Python tool called Py-Tesseract. We can detect the license plate under consideration because most vehicles have a rectangular shape. Therefore, after all the image processing stages, we will discover four edges and contours. Points within the list and treat it as the vehicle's license plate. The stages of the methodology include the following.

4.1 License plate detection

Image preprocessing is crucial in many image processing applications because color information does not help identify the critical edges or contours. Hence, the image is converted to grayscale to remove noise from the image. Following this, we will use a canny detector to detect edges. It is an essential part of computer vision, especially with contours. Edges are defined as sudden changes in an image. Canny Edge Detection is a powerful edge detection method, which has been used to extract the edges from the picture because of its optimal result, well-defined edges, and accurate detection. The line connecting all the similarly intense places along an image's edge is called a contour. They are used in the research of object recognition and form. The contour may be found by combining the advantages of edge detection. The LP contours in the image created result from the approximate contour detection process, the process of detection contours involves connecting edges derived from edge detection. In generating license plate (LP) contours in an image, contours are first sorted based on their area, with a focus on the 30 largest contours determined through experimental evaluation. The perimeter of each contour is then calculated, followed by an approximation of contour perimeters into shapes with fewer vertices. This approximation ensures closure and is guided by a specified precision using an algorithm, Polygonal Approximation of Plane Curves. finally, contours with four corners were selected as license plate contours, encompassing both the plate itself and its borders. as can be observed in Figure 4.

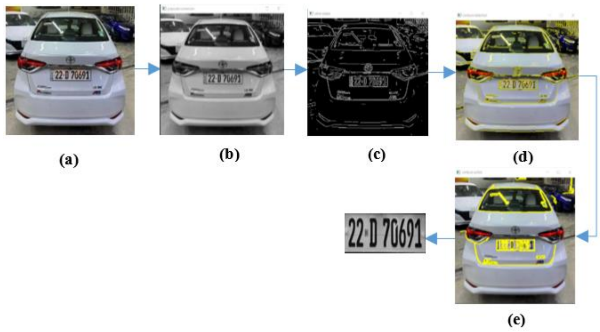


Figure 4. The proposed system in Model 1 (a) image acquisition (b) grayscale conversion (c) canny edge detection (d) contours detection (e) approximate contours detection

4.2 License plate recognition

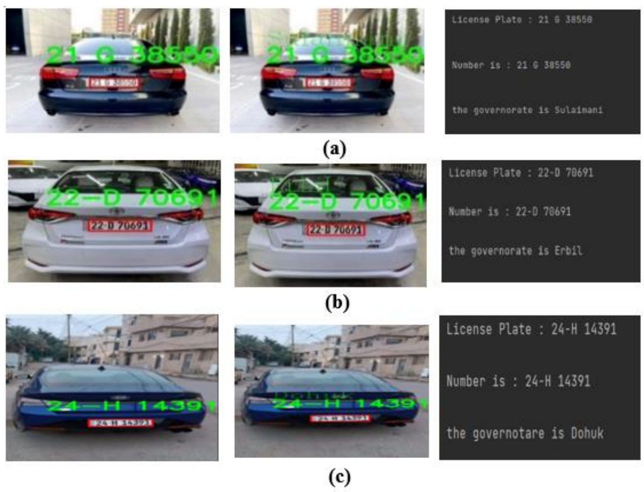


Figure 5. Recognition among the three governorates and their number plates (a) Sulaimani (b) Dohuk (c) Erbil

Reading the number plate data from the segmented image is the last stage in this license plate recognition process. To read characters from the image, we utilize the Py-tesseract package. An optical character recognition (OCR) tool for Python is called Py-tesseract, which can "read" the text embedded in photos.

Tesseract-OCR Engine from Google is wrapped in Python. Optical character recognition recovers the characters from the plate using localized license plate photos as input. based on the first two symbols of the Iraqi license plate previously described it recognizes the governorate as in Figure 1.

A limitation in employing the Tesseract Engine for Iraqi license plate recognition lies in the ambiguity when recognizing license plate letters. This results in an inability to correctly identify characters, ultimately in the failure of license plate number recognition.as shown in Figure 5.

5. METHODOLOGY OF LICENSE PLATE RECOGNITION (LPR) BASED ON DL OF MODEL 2

The process begins by capturing the image, which is then forwarded to the detection stage for license plate detection. Subsequently, the license plate undergoes the segmentation process, where individual characters are isolated. These segmented images are collected and stored in a list, which is then fed into the CNN model during the recognition stage for character recognition. The architecture of the proposed model is visually depicted in Figure 6.

The work's architecture consists of capturing the vehicle's image, which serves as input for the model. Initially, the license plate detection stage identifies the license plate in the input image by detecting its edges and contours. Subsequently, the detected license plate is passed as input to the segmentation stage, which is responsible for segmenting the characters from the license plate and preserving their order. Lastly, the character recognition CNN stage recognizes the characters on the license plate and produces the output.

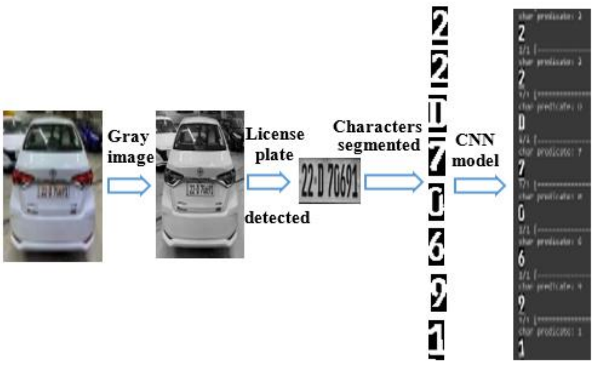


Figure 6. The architecture of the proposed model

5.1 LP detection

This method's initial stage of extracting license plates is similar to that of Model 1. The image is converted to grayscale, and the edges and contours detection is used to find the license plate, as shown in Figure 7.

5.2. Characters segmentation

A bounding box around the license plate is applied during the detection step to identify and locate the license plate. Subsequently, in the segmentation stage, bounding boxes are employed to encompass each character on the license plate. This segmentation step holds crucial importance as it ensures proper isolation of the characters, enabling the model to accurately recognize them. In particular, character segmentation is performed using the following primary steps, as shown in Figure 8.

- Image reading and grayscale conversion to remove some unimportant details from the image.
- Binary conversion of the grayscale image. This technique is crucial for separating characters or specific image components from their background, as most recognition algorithms work only with binary images.
- Splitting the characters. The criterion is that the width of a character must be greater than 5 pixels. If the width of a detected character is below this threshold, it will not be considered a valid character. The value of 5 pixels has been chosen based on observations of the average width of characters in the given Dataset and through trial and error to achieve satisfactory segmentation results.
- Loop through each column's total number of black and white pixels.
- Given the character's initial place, divide the image.

Figure 7. License plate detection

Figure 8. License plate characters' segmentation

5.3 LP recognition

The last stage involves the recognition of the segmented characters extracted from the license plate. To achieve this, a Convolutional Neural Network (CNN) with three convolutional layers is trained specifically for character recognition. Our emphasis was on developing a lightweight model, which led us to propose a CNN architecture with minimal layers and parameters. This streamlined CNN model effectively learns the low-level features of the input data and performs accurate classification. As a result, it is utilized in the recognition module of this study.

5.4 CNN structure of Model 2

The CNN architecture proposed comprises a combination of 2D convolutional layers and pooling layers, with flattened and dense layers utilized at the end. The input to the CNN model is derived from the output of the segmentation process. Given that individual segmented characters are fed into the model, an input size of 28×28 is adequate for accurate character recognition. The model consists of three 2D convolutional layers, with max-pooling layers of size 2×2 inserted between the convolutional layers to extract low-level features from the images, Max pooling is important in the architecture for feature extraction because it helps pick out important patterns by shrinking the data. It keeps the biggest value in a specific area, helping to recognize features even if they're a little bit shifted. This makes calculations faster and more efficient computation and better generalization. as shown in Table 1.

Table 1. Structure of the convolutional and max pooling layers

Convolution, Max Pooling Layers	Kernel Size	Strides	Filters	Activation Function
Conv2D-1	(3,3)		16	ReLu
Conv2D-2	(3,3)		32	ReLu
Max-pooling-1	(2,2)	(2,2)		
Conv2D-3	(3,3)		64	ReLu
Max-pooling-2	(2,2)	(2,2)		

At the end of the model, a Dropout layer with a rate of (0.4) was used to take care of overfitting, and a Flatten layer was used to convert the input dimensions into a single dimension. Finally, two Dense layers were added, including one with the dimensionality of the output space as 128 and the other with the dimensionality of the output space as 36, using the (Relu) activation function. The final layer has 36 outputs for categorizing the 26 alphabets (A-Z) along with the 10 digits (0–9) using the (Softmax) activation function. Figure 9 illustrates a graph showing the proposed CNN architecture.

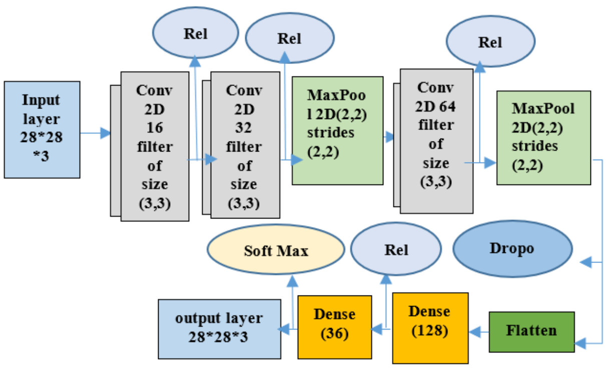


Figure 9. CNN architecture of Model 2

6. EXPERIMENTAL RESULTS

The results of implementing the two models can be illustrated below.

6.1 Model 2 training results

After building the CNN, a learning rate of 0.001 was chosen as a critical factor in the training model for successful neural network training as it governs weight updates per optimization step. it directly affects the convergence speed and the generalization performance of the network. Experiments tested diverse rates, extremes included, aiming for an optimum. Among these, 0.001 outperformed others in loss reduction and validation. Values below 0.001 led to sluggish convergence, while higher ones induced loss fluctuations. Backed by empirical findings, 0.001 was chosen for its balanced training impact, ensuring meaningful results. To complete the training task, the parameters shown in Table 2 were used. After 100 epochs of training, the results for the training and testing data are listed in Table 3.

The training and validation accuracy of the CNN model is plotted in Figure 10, and the training and validation loss is plotted in Figure 11.

Table 2. The results of training the CNN

	Function	Value
Training	accuracy	0.9600
	loss	0.1865
Testing	accuracy	1.0000
	loss	0.0011

Table 3. The parameters of training the CNN

Function	Value
optimizer	Adam
loss	categorical cross-entropy
metrics	accuracy
Batch size	1
epochs	100

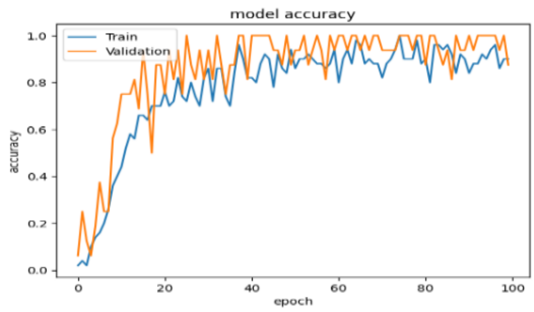


Figure 10. The training and validation accuracy of Model 2

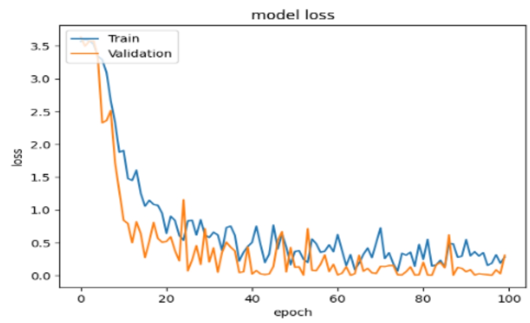
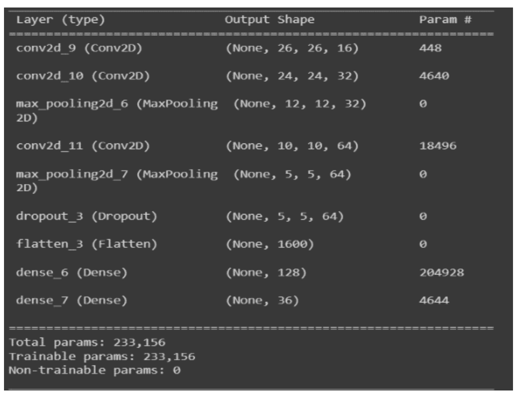


Figure 11. The training and validation loss of Model 2



Layer (type)	Output Shape	Param #
conv2d_9 (Conv2D)	(None, 26, 26, 16)	448
conv2d_10 (Conv2D)	(None, 24, 24, 32)	4640
max_pooling2d_6 (MaxPooling2D)	(None, 12, 12, 32)	0
conv2d_11 (Conv2D)	(None, 10, 10, 64)	18496
max_pooling2d_7 (MaxPooling2D)	(None, 5, 5, 64)	0
dropout_3 (Dropout)	(None, 5, 5, 64)	0
flatten_3 (Flatten)	(None, 1600)	0
dense_6 (Dense)	(None, 128)	204928
dense_7 (Dense)	(None, 36)	4644

Total params: 233,156
Trainable params: 233,156
Non-trainable params: 0

Figure 12. Model summary of applying the 2D CNN of Model 2

In this work, 2D CNN structures were used, consisting of three successive CNN layers followed by a dropout and a max-pooling layer, which are demonstrated as a model summary in Figure 12.

The CNN was trained using a batch size of 1, for a total of 11,000 iterations over 100 epochs on a dataset of 1100 images. The training data was augmented using random rescale=1. /255, width shift range=0.1, and height shift range=0.1.

6.2 Evaluation of the models

To evaluate the effectiveness of the two proposed models, they were tested using (180) new Iraqi car images from several scenes under different conditions.

For the rating criteria, the proposed models are evaluated by calculating the accuracy of the recognition, which is defined as the number of correctly recognized license plates, which was (63) images in the first form and (172) images in the second form, divided by the number of correctly recognized images plus the number of incorrectly recognized images, based on Eq. (1).

$$Accuracy = \frac{true\ positive}{true\ positive + false\ positive} \quad (1)$$

The results show that Model 2 gives high accuracy, where the recognition accuracy reaches up to (95.5%) confirming that it can effectively recognize license plate characters in different situations, while Model 1 gives low accuracy of (35%).

6.3 Experimental setup

The proposed system (Model 1) was implemented using Python 3.10 language, and the experiments were performed on a platform having Intel (R) Core (TM) i7-5600U CPU @ 2.60GHz 2.59 GHz, with RAM 8.00 GB on System type 64-bit operating system, x64-based. On the other hand, the other proposed system (Model 2) was carried out using Google Colab, a cloud server. The training tools used in this case were Python, Keras, TensorFlow, etc.

7. DISCUSSION OF THE RESULTS

This section discusses the advantages and limitations of the two proposed models. Based on the experimental analysis and the performance evaluation for Model 1 and after detecting all the possible best rectangular images of a license plate, the license plate was extracted from the image. The last step is to extract the text in the form of a string from the image using Py-Tesseract, which is an optical character recognition (OCR) library in Python. The percentage of recognition of license plate numbers was 35%, while it was over 95.5% in Model 2, according to the evaluation of the two models when they were tested with 180 images of vehicles using the newest Iraqi license plates. The method applied in Model 1 is faster in terms of execution time and requires fewer calculations than those of Model 2 because it does not need to be trained by the dataset. However, its accuracy is low, where the shortcomings of Model 1 include the ambiguous recognition of the following letters a: (0) is recognized as (6) for his being open top end as per the new Iraqi license; (G) was recognized as (6), and (J) was recognized as ")", which leads to a failure in recognizing the characters correctly. Thus, the license plate number recognition fails.

This shows the limitations of the first model in recognition, stemming not solely from the inherent limitations of OCR functionality but also from the unique design attributes of license plate characters. These characters often deviate from standard designs, such as the numeral '0' on plates that feature an open top, introducing recognition ambiguities. as shown in Figure 13.

Therefore, it was necessary to recognize them more precisely using one of the deep learning techniques, which is the convolutional neural network that was used in Model 2.

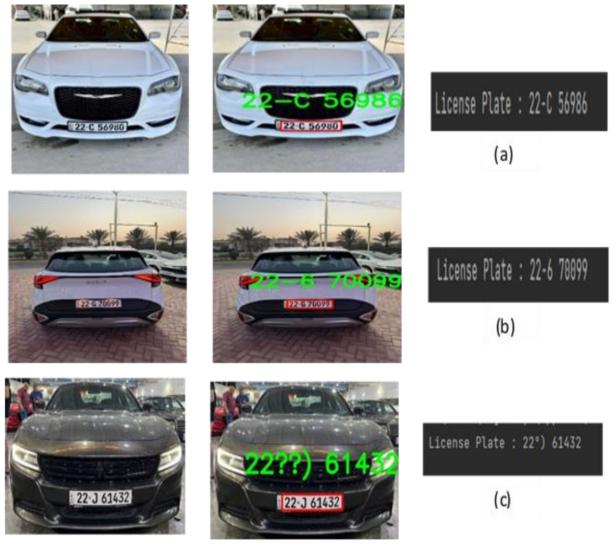


Figure 13. The ambiguous recognition of characters (a) recognizing (0 as 6) (b) recognizing (G as 6) (c) recognizing (J as))

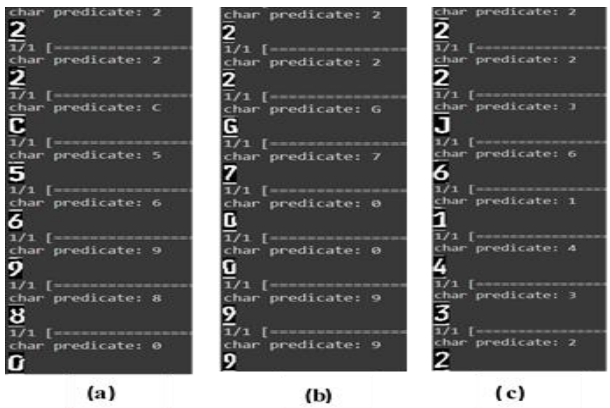


Figure 14. Resolving the ambiguous recognition of characters (a) resolve (0 as 6) (b) resolve (G as 6) (c) resolve (J as)) The way that section titles and other headings

Model 1 recognized only a limited number of license plate characters without the need for a segmentation stage and without training using the external dataset. In contrast, Model 2 used the segmentation stage, which is followed by training the CNN model with the dataset (b) that contains (1100) images of different characters modified from the Kaggle dataset (a), as shown in Figure 9. Thus, Model 2 managed to recognize all license plate characters efficiently, and it resolved the ambiguous recognition of characters, such as the characters (0), (G), and (J), and hence, it succeeded in correctly identifying the vehicle number of the license plate.

The efficacy of the second model in accurately identifying distinct license plate characters can be attributed to its adeptness at overcoming the challenge of ambiguity. This achievement is by leveraging a Convolutional Neural Network (CNN) that undergoes training on character patterns within the training dataset. By meticulously extracting and learning these distinctive features, the second model gains the capacity to discriminate license plate characters with a higher degree of precision. as shown in Figure 14.

8. CONCLUSION

The summary of the results focuses on two license plate recognition models, Model 1, using the Tesseract OCR library, achieved an accuracy rate of 35% in recognizing license plate numbers. It was faster and required fewer calculations than Model 2, as it did not depend on external dataset training. However, Model 1 had limitations, including ambiguous recognition of certain characters such as '0' being misidentified as '6'. This affected its overall accuracy in identifying license plate numbers.

Model 2 utilized deep learning techniques, specifically Convolutional Neural Networks (CNNs), to achieve higher accuracy reached 95.5%. It employed a segmentation stage and trained the CNN model using a dataset of 1100 diverse license plate characters. Unlike Model 1, Model 2 achieved accurate recognition of all license plate characters, resolving the ambiguities surrounding characters like '0' and 'G'. The success of Model 2 can be attributed to its ability to extract and learn intricate character patterns within the training dataset, providing precise identification of license plate characters.

This study represents the first of its kind by introducing the inaugural model for recognizing the latest Iraqi license plates, a task that was previously unaddressed due to the absence of a standardized dataset for this newly introduced version.

The significance of this contribution is underscored by its role in addressing security challenges within Iraq. By effectively recognition the new license plates, it enhances the data available with vehicle numbers to both public and private entities. This, in turn, contributes to the safeguarding of vehicle garages and overall security measures.

The limitations in the initial model can be faced in the future by conducting experiments that involve applying a segmentation process to the license plate and then to initiating segmented character recognition through Tesseract. Within the second model, although its implementation demands substantial time and an expansive dataset, its strengths stem from its adeptness at extracting the features of license plate characters. This precision significantly influences the accurate recognition of vehicle numbers and their governorates, consequently facilitating the detection of stolen vehicles. Additionally, the model aids traffic management systems and bolsters garage security by effectively discerning unauthorized vehicles, thereby elevating overall security measures in Iraq.

REFERENCES

[1] Wang, W., Yang, J., Chen, M., Wang, P. (2019). A light CNN for end-to-end car license plates detection and recognition. IEEE Access, 7: 173875-173883. https://doi.org/10.1109/ACCESS.2019.2956357

[2] Chen, H., Lin, Y., Zhao, T. (2023). Chinese license plate recognition system based on convolutional neural network. Highlights in Science, Engineering and Technology, 34: 95-102. https://doi.org/10.54097/hset.v34i.5386

[3] Kaur, P., Kumar, Y., Ahmed, S., Alhumam, A., Singla, R., Ijaz, M.F. (2022). Automatic license plate recognition system for vehicles using a CNN. Computers, Materials & Continua, 71(1): 35-50. https://doi.org/10.32604/cmc.2022.017681

[4] Hassan, N.F., Ali, A.E., Aldeen, T.W., Al-Adhami, A. (2021). Video mosaic watermarking using plasma key. Indonesian Journal of Electrical Engineering and Computer Science, 22(2): 619-628. https://doi.org/10.11591/ijeecs.v22.i2.pp619-628

[5] Falih, E., Mazhar, A.N. (2020). Using FREAK descriptor to classify plasma influence in Mice sperm. Karbala International Journal of Modern Science, 6(1): 36-43. https://doi.org/10.33640/2405-609X.1352

[6] Abbass, G.Y., Marhoon, A.F. (2022). Car license plate segmentation and recognition system based on deep learning. Bulletin of Electrical Engineering and Informatics, 11(4): 1983-1989. https://doi.org/10.11591/eei.v11i4.3434

[7] https://ninanews.com/Website/News/Details?key=977300.

[8] Géron, A. (2022). Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow. O'Reilly Media, Inc.

[9] Chollet, F. (2021). Deep learning with python. Simon and Schuster, 10(2): 119. https://doi.org/10.1016/s1474-4422(11)70007-2

[10] Mubeen, S., Brahmani, J., Kalyan, D.P., Jagirdar, A., Kumar, A.P. (2022). Optical character recognition using tesseract. International Journal for Research in Applied Science & Engineering Technology (IJRASET), 10(1): 672-675. https://doi.org/10.22214/ijraset.2022.47414

[11] San, H., Patilano, L., Botin, M., Espinosa, A.A., Barco, J.V., Gamboa, G.A.L. (2022). Plate number real-time recognition using Tesseract OCR. Abstract, 5(3): 381-385. http://www.ijsred.com/volume5/issue3/IJSRED-V5I3P54.pdf.

[12] Shashirangana, J., Padmasiri, H., Meedeniya, D., Perera, C. (2020). Automated license plate recognition: a survey on methods and techniques. IEEE Access, 9, 11203-11225. https://doi.org/10.1109/ACCESS.2020.3047929

[13] Ma, L., Zhang, Y. (2021). Research on vehicle license plate recognition technology based on deep convolutional neural networks. Microprocessors and Microsystems, 82: 103932. https://doi.org/10.1016/j.micpro.2021.103932

[14] Lateef, R.A., Abbas, A.R. (2022). Tuning the hyperparameters of the 1D CNN model to improve the performance of human activity recognition. Engineering and Technology Journal, 40(4): 547-554. https://doi.org/10.30684/etj.v40i4.2054

[15] AL-Musawi, R. S., Ali, A. H., Al-Majdi, K., Al Gayar, S. (2022). Development of a convolutional neural network joint detector for non-orthogonal multiple access uplink receivers. EUREKA: Physics and Engineering, 4: 170-178, 2022, https://doi.org/10.21303/2461-4262.2022.002496

[16] Ibitoye, O., Ejidokun, T., Dada, O., Omitola, O. (2020, December). Convolutional neural network-based license plate recognition techniques: A short overview. In 2020

International Conference on Computational Science and Computational Intelligence (CSCI), Las Vegas, NV, USA, pp. 1529-1532. https://doi.org/10.1109/CSCI51800.2020.00283
[17] Abdullah, M., Al-Nawah, S.M., Osman, H., Jaffar, J. (2021). License plate recognition techniques: Comparative study. Malaysian Journal of Computer Science, 94-105. https://doi.org/10.22452/mjcs.sp2021no1.9
[18] Jawale, M.A., William, P., Pawar, A.B., Marriwala, N. (2023). Implementation of number plate detection system for vehicle registration using IOT and recognition using CNN. Measurement: Sensors, 27: 100761. https://doi.org/10.1016/j.measen.2023.100761
[19] Silva, S.M., Jung, C.R. (2020). Real-time license plate detection and recognition using deep convolutional neural networks. Journal of Visual Communication and Image Representation, 71: 102773. https://doi.org/10.1016/j.jvcir.2020.102773
[20] Chadha, A., Kashyap, S., Gupta, M., Kumar, V. (2020). License plate recognition system using OpenCV & PyTesseract. CSI Journal of Computing, 3(3): 31-35.
[21] Goel, T., Tripathi, K.C., Sharma, M.L. (2020). Single line license plate detection using OpenCV and tesseract. International Research Journal of Engineering and Technology (IRJET), 7(5): 5884-5887.
[22] Suneetha, K., Mounika Raj, K. (2021). Automatic vehicle number plate recognition system (AVNPR) using OpenCV python. In Proceedings of the 2nd International Conference on Computational and Bio Engineering: CBE 2020, Tirupati, India, 215, pp. 487-496. https://doi.org/10.1007/978-981-16-1941-0_49
[23] Mohd Ariffin, M.A. (2015). Automatic license plate detection and recognition. International Advanced Research Journal in Science, Engineering and Technology, 8(8): 22-26. https://doi.org/10.17148/iarjset.2021.8805
[24] Sawalkar, A., Pathan, A., Kakade, A., Telvekar, P., Chandne, B. (2022). Number plate recognition system using pytesseract & Open Cv. International Research Journal of Modernization in Engineering Technology and Science, 6: 2582-5208. https://www.irjmets.com/uploadedfiles/paper/issue_6_june_2022/25342/final/fin_irjmets1659638501.pdf.
[25] Abedin, M.Z., Nath, A.C., Dhar, P., Deb, K., Hossain, M.S. (2017). License plate recognition system based on contour properties and deep learning model. In 2017 IEEE Region 10 Humanitarian Technology Conference (R10-HTC), Dhaka, Bangladesh, pp. 590-593. https://doi.org/10.1109/R10-HTC.2017.8289029
[26] Marzuki, P., Syafeeza, A.R., Wong, Y.C., Hamid, N.A., Alisa, A.N., Ibrahim, M.M. (2019). A design of license plate recognition system using convolutional neural network. International Journal of Electrical and Computer Engineering (IJECE), 9(3): 2196-2204. https://doi.org/10.11591/ijece.v9i3.pp2196-2204
[27] Alam, N.A., Ahsan, M., Based, M.A., Haider, J. (2021). Intelligent system for vehicles number plate detection and recognition using convolutional neural networks. Technologies, 9(1): 9. https://doi.org/10.3390/technologies9010009
[28] https://www.kaggle.com/datasets/sarthakvajpayee/ai-indian-license-plate-recognition-data?select=data.