


RESEARCH

Open Access



Egyptian car plate recognition based on YOLOv8, Easy-OCR, and CNN

Amany Sarhan^{1*} , Rowyda Abdel-Rahem², Bassel Darwish², Arwa Abou-Attia², Ahmed Sneed², Shahd Hatem², Awatef Badran² and Mohamed Ramadan²

*Correspondence:
amany_sarhan@f-eng.tanta.edu.eg

¹ Department of Computer and Control Engineering, Tanta University, Tanta, Egypt

² Department of Artificial Intelligence, Faculty of Artificial Intelligence, Delta University, Coastal International Road, Dakahlia, Egypt

Abstract

This research presents an innovative approach to Egyptian car plate recognition using YOLOv8 and optical character recognition (OCR) technologies. Leveraging the powerful object detection capabilities of YOLOv8, the system efficiently detects car plates within images, videos, or real-time. Subsequently, OCR algorithms are applied to extract alphanumeric characters from the identified plates, facilitating accurate license plate recognition. The integration of YOLOv8 and OCR enhances the system's robustness in varying conditions, contributing to improved performance in real-world scenarios. This study advances the field of automatic license plate recognition, showcasing the potential for practical applications in traffic management, law enforcement, and security systems. A public dataset of Egyptian car plates is used for training and testing the model. Two OCR approaches are used and tested which proved their performance, while CNN-based approach reaches 99.4% accuracy.

Keywords: YOLOv8, Easy-OCR, CNN, Car plate recognition, Car plate detection, Egyptian car plates

Introduction

Automatic license plate recognition (ALPR) plays a vital role in various applications including traffic surveillance, toll collection, parking systems, and theft prevention [1]. The key goal of ALPR systems is to detect license plates in images and video streams and recognize the characters without extensive human involvement, thereby saving time and resources (Fig. 1). While traditional ALPR techniques relied on image processing and machine learning, recent advances in deep learning have led to highly accurate ALPR systems based on convolutional neural networks (CNNs) [2, 3]. However, ALPR remains an active research problem due to factors such as lighting conditions, angles, distances, backgrounds, and license plate layouts differing across countries [4]. Most existing ALPR research has focused on Latin script plates from Europe, United States, Brazil, and China [5, 6]. Limited work has been done for Arabic script plates prevalent in the Middle East [7, 8].

Car plates' shapes differ from one country to another as shown in Fig. 2, which shows a sample of Egyptian and Saudi car plates [8, 9]. Their design and color also diverge. These variations make it challenging to develop a universal car plate detector and recognizer.



Fig. 1 Car plate detection and recognition



Fig. 2 Sample of Arabic car plates

There have been few previous works for Arabic car plates such as those in [10–12]. Efforts in this direction are not as extensive as those for non-Arabic car plates, such as American or Chinese ones [13, 14]. More work should be directed to this area due to its importance.

Deep learning has been deployed in many applications around us ranging from simple to large problems. Its models are gaining the trust due to the advances made in their construction making its accuracy high enough to rely on their results. Recently, the You Only Look Once (YOLO) family of models have shown immense promise in object detection applications. Tiny-YOLOv3 provides a good balance of speed and accuracy for real-time ALPR [10]. CPR is challenging because it depends on several factors including the plate country's design, colors, environment, language, and texture. Both license plate detection and character recognition could be done using the object detection techniques. There are two approaches of object detection features, hand-engineered features (Haar, HOG, SIFT, SURF, etc.) and deep neural networks features based on convolutional neural networks (YOLO, SSD, Faster-R-CNN, RefineDNet, etc.).

Arabic ALPR using deep learning has been scarcely explored due to lack of publicly available datasets. To this end, the main objective of this paper is to introduce an innovative Egyptian car plate recognition system employing YOLOv8 [15], optical character recognition (OCR) [16], and convolutional neural network (CNN) for an efficient and robust ALPR system tailored to Egyptian license plates. YOLOv8 provides efficient object detection, enabling the localization of car plates, while OCR algorithms extract

alphanumeric characters for accurate license plate recognition. The integration of these technologies enhances system robustness in diverse conditions, demonstrating potential applications in traffic management, law enforcement, and security systems. This research contributes to the advancement of automatic license plate recognition, offering practical solutions for real-world implementation. CNN's ability to automatically learn distinctive visual features of license plate characters enables greater precision and adaptability to complex backgrounds and occlusion. The integrated system combining YOLOv8's efficient object detection capabilities with the CNN's deep learning-based OCR approach results in an ALPR solution that is highly performant in real-world deployment. We demonstrate and evaluate our approach on the publicly available EALPR dataset [9]. Our main contributions can be summarized in solid points as:

- Introducing and implementing a real-time ALPR pipeline by integrating YOLOv8, Easy-OCR, and CNN to detect and recognize Egyptian license plates accurately.
- Evaluation on the EALPR benchmark containing Arabic digits and characters, showing improved performance over prior art.
- Demonstrating the applicability of advanced deep learning techniques for Arabic ALPR, given suitable training data.

Background and related work

Arabic car plates, like any other Arabic content-based area, did not receive as much attention in the literature as non-Arabic car plates. Few works are connected to ours, and we strive to summarize and focus on the most current and relevant to ours.

In 2020, authors proposed an automatic license plate detection system using an improved Faster R-CNN model based on multi-style Egyptian license plate [17]. Faster R-CNN was chosen for its ability to detect small objects like license plates. They enhance it by adding an adaptive attention network to improve detection and segmentation, a deconvolution layer for sensitivity to small objects, and modified anchors suited to license plates. A lightweight Inception V3 model enables real-time processing. To train and evaluate the system, they collected and annotated a new dataset of Arabic country license plates containing Arabic/Indian numbers and Arabic/Latin letters. Experiments demonstrate high detection accuracy and fast 23 FPS processing on this dataset. Key contributions are improving Faster R-CNN for license plates, collecting a new Arabic license plate dataset, and achieving real-time performance with high precision and recall. The proposed system has advantages over existing models and could be applied for auto license plate recognition in Arabic countries.

In 2021, a deep machine learning-based Egyptian vehicle license plate recognition system was introduced [18] that presents four different systems to automatically recognize license plates on Egyptian vehicles, comparing classical machine learning techniques to newer deep learning approaches. For detecting the license plate location in an image, the authors found that a deep learning-based Faster R-CNN model outperformed an edge detection method by 32% in accuracy. However, a connected component labeling technique surpassed a Faster R-CNN model by 6% for the character segmentation task. Character recognition was 8% more accurate with a convolutional neural network instead of using PCA for feature extraction and KNN classification.

Finally, recognizing the entire license plate through a deep learning model proved superior to DCT feature extraction with SVM by around 3% accuracy. Based on these results, the authors recommend an optimal Egyptian license plate recognition system utilizing Faster R-CNN for plate detection, connected component labeling for segmentation, and convolutional neural networks for character and whole plate recognition, integrating traditional computer vision with deep learning for maximum accuracy and robustness.

In 2022, an automatic license plate recognition system for cars in Saudi Arabia was presented [19]. The authors collected 50 images of parked cars in various conditions. After preprocessing the images, they applied Canny edge detection to locate the license plates. To reduce noise, different thresholding techniques were used. For segmentation, they utilized horizontal projection to separate the plate into rows. Then masking isolated the plate region. OCR was used to read the English and Arabic text on the plates separately. The Arabic letters were reshaped before combining them with the English text and overlaying on the original image. The Canny algorithm coupled with projection profiling and suitable preprocessing achieved 96% accuracy on the English plates and 92% on the Arabic. The results demonstrate effective license plate detection and recognition on a locally collected Saudi dataset, outperforming prior published methods.

In 2022, the authors proposed a real-time Automatic License Plate Detection and Recognition (ALPR) system specifically designed for detecting and recognizing Egyptian license plates using Tiny-YOLOV3. The proposed system, comprising two deep convolutional neural networks, underwent testing on the first publicly available Egyptian Automatic License Plate (EALPR) dataset. Experimental results highlight the system's superior robustness in both detecting and recognizing Egyptian license plates, achieving mean average precision values of 97.89% and 92.46% for license plate detection and character recognition, respectively.

In 2023, a reliable deep learning approach was presented for Iranian license plate recognition [12]. The method involves two steps—first detecting the license plate rectangles in the input image using YOLOv4-tiny model, and then cropping these plates and recognizing the characters using a Convolutional Recurrent Neural Network (CRNN) and Connectionist Temporal Classification (CTC). The authors created datasets of 3065 images with license plates for detection and 3364 images of license plate characters for recognition. A key advantage of their approach is that character segmentation is not needed prior to recognition. They achieved fast processing times of 0.0074 s per image for detection and 0.127 s for recognition, with high accuracy. The method is storage-efficient at under 2 MB and well-suited for mobile applications.

In 2023, the work in [20] presents a method using the MVSR normalization algorithm to improve car license plate recognition, especially for low resolution or rotated images. They apply Mean-Variance-SoftMax-Rescale processing to enable high accuracy real-time detection. After reviewing various prior approaches using image processing, deep learning models like CNN and YOLO, and OCR, they demonstrate that their proposed algorithm outperforms other techniques. Using FPGA and GPU platforms, they achieve fast processing times. Their approach does not require character segmentation before recognition.

In 2023, the research in [21] presents an efficient automated vehicle license plate recognition system using image processing techniques [22–25]. It involves four main steps—preprocessing to improve image quality, license plate region extraction using morphological operations, character segmentation with the bounding box method, and finally template matching for character recognition. They reviewed various prior approaches using deep learning and image processing. Their proposed methodology attains high recognition accuracy of 94.17% on MATLAB. Key advantages are its simplicity and efficiency. It can handle images captured under varied conditions and is robust to factors like poor resolution, illumination, reflection, etc. The system has numerous applications in traffic control, parking, toll collection, security, and surveillance. Overall, they demonstrate an accurate and effective license plate recognition methodology using basic image processing techniques. Table 1 summarizes these works mentioning the dataset used, the algorithms implemented, and the highest accuracy reached by each of them.

Proposed methodology

In this work, we propose a two-stage automatic license plate recognition (ALPR) system for detecting and recognizing Egyptian car license plates. The proposed ALPR framework is illustrated in Fig. 3. The system has two main stages: car plate detection and car plate recognition which perform the following tasks:

1. The first phase is the detection of the car plate relies on an image captured from camera to generate a segmented image. This is achieved using YOLOv8 deep learning model, which is currently witnesses a widespread use of the YOLO series models due to their efficiency for different object detection. We have implemented version 8 of this algorithm to ensure a more reliable and precise detection of car plates which

Table 1 Previous related work for car plate detection

References	Dataset	Algorithm	Highest accuracy
Shehata et al. [18]	VLP dataset	Deep convolutional neural networks (DCNN), KNN, Connected Component Labeling (CCL)	CCL achieved 96% detection accuracy DCNN achieved 95% recognition accuracy
Alghyaline [11]	JALPR dataset	Convolutional Neural Networks (CNNs), YOLO3 network architecture	Jordanian license plates achieved 87% recognition accuracy Commercial systems have recognition accuracies that are less than 81%
Youssef et al. [10]	EALPR dataset	Convolutional Neural Networks (CNNs), YOLO3 network architecture	97.89% and 92.46% for LP detection and character recognition
Aljelawy [26]	Not mentioned	YOLO (You Only Look Once) OCR (optical character recognition) and cascade classifier	YOLO: efficiency is high and near to 100% until 6 m Cascade Classifier: The accuracy near 75%
Moussaoui et al. [27]	Arabic and Latin Plates	YOLOv7, Thresholding, Kernel, Arabic OCR, Easy-OCR	Detection: 98% Segmentation: 99%

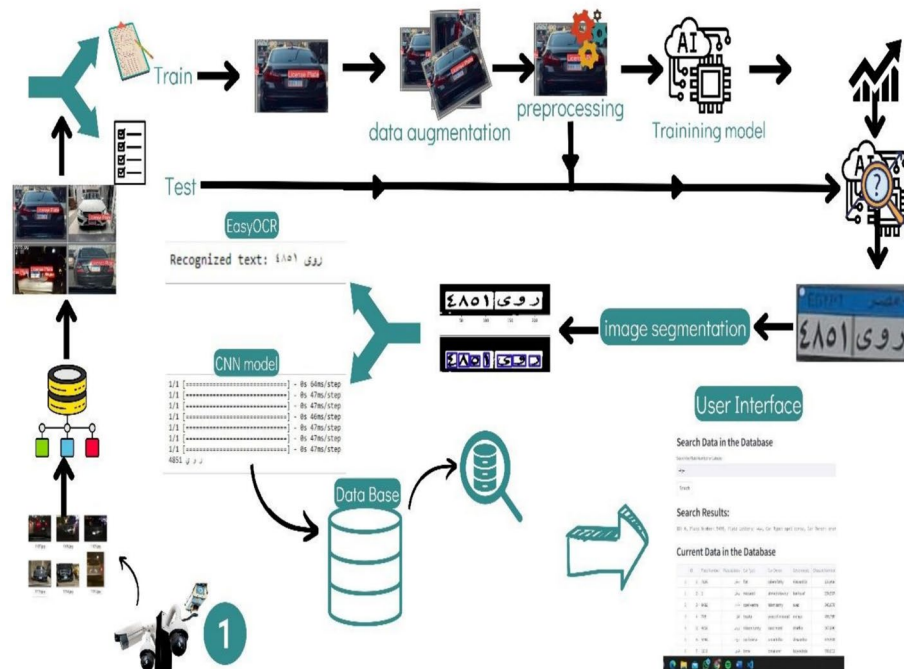


Fig. 3 The proposed ALPR framework

will affect the second phase of the system. The training and testing of the YOLOv8 model are performed using the EALPR dataset.

2. The second phase involves the recognition of the text and number within the detected car plate. For this task, we employed two different approaches: Easy-OCR tool and a custom CNN deep learning model, which we trained with letters and images from plates dataset with Arabic characters to match the problem at hand. Each phase will be discussed in details along with any training and dataset involved in its building.

In the training of the models, we perform image preprocessing and data augmentation. The system is then applied to real images captured by a camera to generate the details of the car with that plate.

Car plate detection using YOLOv8

Accurate localization of license plates within complex scenes is a key first step. We leverage YOLOv8 [28], the latest version of the popular YOLO model family, as our detector due to its improved speed, accuracy, generalization ability, and robustness to occlusion. YOLOv8 utilizes a transformer-based backbone architecture instead of the prior Darknet models. The YOLO (You Only Look Once) algorithm has emerged as a leading deep learning-based object detection framework, distinguished by its real-time efficiency and straightforward design [29, 30]. In the realm of computer vision, YOLOv8 stands out as a notable iteration, further refining the delicate equilibrium between accuracy and speed. With its core principles rooted in a unique detection pipeline, YOLOv8 excels in directly predicting bounding boxes and class probabilities in a single pass-through input

image. This enhances the contextual reasoning capability leading to better detection performance, especially for small objects like license plates.

In the context of car plate recognition, YOLOv8’s capabilities become particularly significant. Leveraging a grid-based approach, the algorithm divides the image into cells, each responsible for predicting bounding boxes and class probabilities for objects within its domain. The incorporation of anchor boxes, predefined shapes with varying scales and aspect ratios, enhances YOLOv8’s versatility in handling car plates of diverse sizes and proportions.

The strength of YOLOv8 lies in its adept utilization of a deep convolutional neural network (CNN), adopting a variant of the DarkNet architecture. This backbone network seamlessly integrates multiple convolutional and fully connected layers, facilitating effective object classification. The algorithm’s prowess in real-time processing is especially valuable in applications like car plate recognition for video surveillance and autonomous driving, where swift and accurate responses are imperative.

Despite its remarkable attributes, YOLOv8 encounters trade-offs between accuracy and speed inherent to its single-shot nature. Challenges may arise in detecting smaller or low-contrast car plates compared to multi-stage methods. Additionally, accurately localizing car plates within crowded or overlapping scenarios presents certain complexities. Nevertheless, YOLOv8 strikes a commendable balance between accuracy and speed, rendering it well-suited for real-time applications, including car plate recognition.

The influence of YOLOv8 and its variants goes beyond theoretical frameworks, drastically altering the landscape of object identification in real-world circumstances. Its simplicity, along with real-time performance, has established its reputation among researchers and practitioners alike, making it an invaluable tool in the field of automobile plate identification and beyond. YOLOv8 is the most recent version of the YOLO object detection model (Fig. 4). This latest version has the same architecture as the prior one 6, but it includes numerous improvements over previous versions of YOLO, such as a new neural network architecture that uses both Feature Pyramid Network (FPN) and Path Aggregation Network (PAN), as well as a new labeling tool that simplifies the annotation

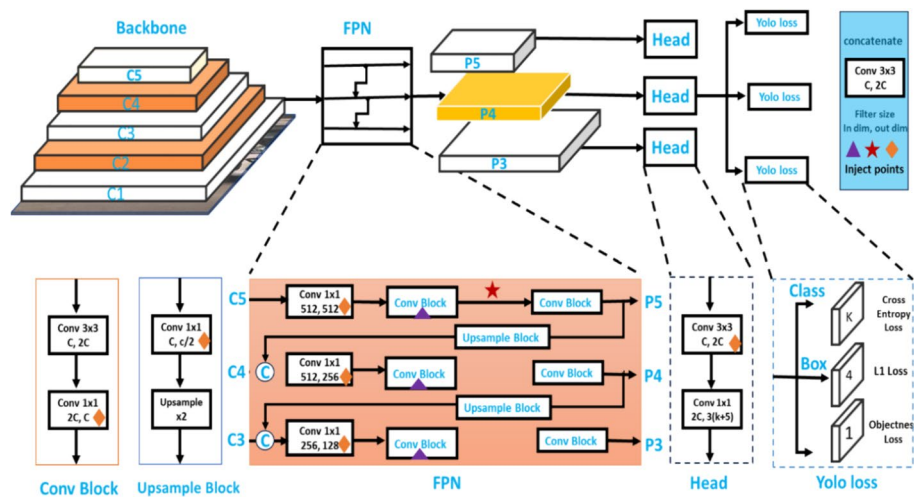


Fig. 4 YOLOv8 architecture

process. This labeling tool has various handy features, including auto-labeling, labeling shortcuts, and customized hotkeys. The combination of these attributes facilitates picture annotation for model training. The FPN operates by gradually decreasing the spatial resolution of the input picture while increasing the number of feature channels. This leads in the generation of feature maps that can detect objects at various sizes and resolutions. In contrast, the PAN design uses skip connections to collect characteristics from various layers of the network. This allows the network to better collect information at many scales and resolutions, which is critical for reliably recognizing objects of varying sizes and forms.

The model is first pre-trained on the massive COCO dataset before being fine-tuned for our job using EALPR. We utilize a single-scale version with an input resolution of 640×640 . The model is trained to predict a single class label (the license plate). The bounding box coordinates returned by YOLOv8 enable us to clip the observed plate sections. For each image, create an annotation text file. Annotation text files should have the same names as image files and the ".txt" extensions. In the annotation files, you should add records about each object that exist on the appropriate image in the following format in Fig. 5.

These values are normalized to the image dimensions, enabling consistent representation across diverse images. This YOLO label format facilitates effective training and localization of objects, such as license plates, within the dataset, aligning with the YOLOv8 architecture used in our project for accurate and efficient object detection. Table 2 gives the details of YOLOv8 model architecture.

Plate recognition using Easy-OCR

The cropped license plate images are passed to an OCR model for character recognition. We opted for Easy-OCR [16] due to its ease-of-use, high accuracy, and built-in support for Arabic script recognition. Easy-OCR encapsulates multiple OCR techniques like feature extraction and sequence modeling and provides an end-to-end pipeline. The images



Fig. 5 Yolo label format [31]

Table 2 Yolov8 model architecture

Type	In/out size	Params	Module	Arguments
Conv	[3, 16, 3, 2]	464	Conv	[3, 16, 3, 2]
Conv	[16, 32, 3, 2]	4672	Conv	[16, 32, 3, 2]
C2f	[32, 32, 1, True]	7360	C2f	[32, 32, 1, True]
Conv	[32, 64, 3, 2]	18,560	Conv	[32, 64, 3, 2]
C2f	[64, 64, 2, True]	49,664	C2f	[64, 64, 2, True]
Conv	[64, 128, 3, 2]	73,984	Conv	[64, 128, 3, 2]
C2f	[128, 128, 2, True]	197,632	C2f	[128, 128, 2, True]
Conv	[128, 256, 3, 2]	295,424	Conv	[128, 256, 3, 2]
C2f	[256, 256, 1, True]	460,288	C2f	[256, 256, 1, True]
SPPF	[256, 256, 5]	164,608	SPPF	[256, 256, 5]
Upsample	[None, 2, 'nearest']	0	Upsample	[None, 2, 'nearest']
Concat	[1]	0	Concat	[1]
C2f	[384, 128, 1]	148,224	C2f	[384, 128, 1]
Upsample	[None, 2, 'nearest']	0	Upsample	[None, 2, 'nearest']
Concat	[1]	0	Concat	[1]
C2f	[192, 64, 1]	37,248	C2f	[192, 64, 1]
Conv	[64, 64, 3, 2]	36,992	Conv	[64, 64, 3, 2]
Concat	[1]	0	Concat	[1]
C2f	[192, 128, 1]	123,648	C2f	[192, 128, 1]
Conv	[128, 128, 3, 2]	147,712	Conv	[128, 128, 3, 2]
Concat	[1]	0	Concat	[1]
C2f	[384, 256, 1]	493,056	C2f	[384, 256, 1]
Detect	[1, [64, 128, 256]]	751,507	Detect	[1, [64, 128, 256]]

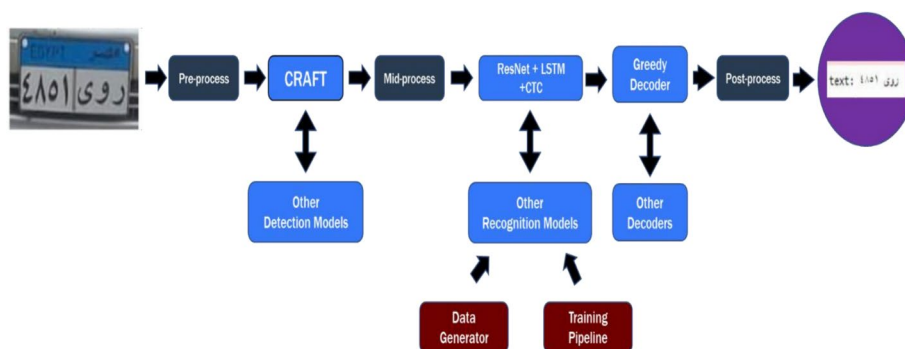


Fig. 6 Steps of Easy-OCR in segmenting and recognition of car plate

undergo cleaning and enhancement pre-processing before being fed to the OCR model, as shown in Fig. 6. Easy-OCR outputs the recognized text string corresponding to the license plate characters. The text strings from individual plates are collated to build the final ALPR output (Fig. 7).

Segmentation and character recognition Easy-OCR

Segmentation is a method of breaking down an image into smaller portions to detect the borders of text, characters, words, and lines. Horizontal and vertical projections

$$BF[I]_p = \frac{1}{W_p} \sum_{q \in S} G_{\sigma_s}(\|p - q\|) G_{\sigma_r}(|I_p - I_q|) I_q \quad (1)$$

↓

**Normalization
Factor**

↓

Space Weight

↓

Range Weight

Fig. 7 Bilateral filter components

are used on the image to define the limits of the text, characters, words, and lines. The picture is then broken into smaller sections, each comprising one character, word, or sentence. To minimize the noise in the plate image displayed in Fig. 8, we used a bilateral filter as indicated in Fig. 7.

Parameters σ_s and σ_r will specify the amount of filtering for the image I. Equation (1) is a normalized weighted average where G_{σ_s} is a spatial Gaussian weighting that decreases the influence of distant pixels, and G_{σ_r} is a range Gaussian that decreases the influence of pixels q when their intensity values differ from I_p [32, 33].

Thresholding is a basic image processing method that transforms a grayscale image into a binary image (as shown in Fig. 9), which contains only black and white pixels. The algorithm selects a threshold value and then classifies each pixel in the image depending on whether its intensity exceeds or falls below the set threshold. The dual threshold strategy uses two thresholds on the gradient values: a high threshold and a low threshold. If an edge pixel's gradient value exceeds the high threshold, it is considered a strong edge. If it falls between the two criteria, it is considered a weak edge.

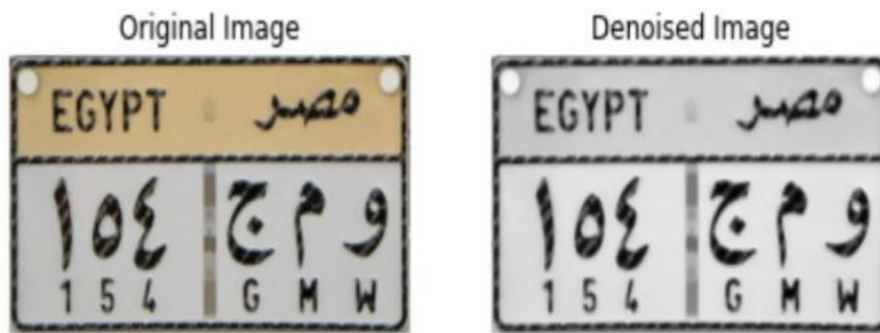


Fig. 8 Applying bilateral filter to Egyptian car plate

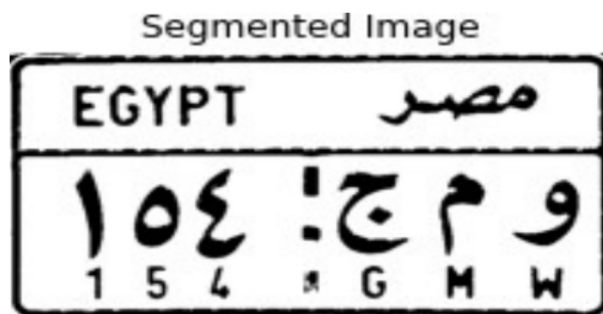


Fig. 9 Converting a grayscale car plate image into a binary image

However, if it moves below the low level, it will be repressed. The purpose for keeping both strong and weak edges at this point is as follows: If a weak edge is coupled to a strong edge, it is most likely a true edge; otherwise, it is most likely noise.

Character recognition Easy-OCR

OCR is used after dividing the characters, and each character is detected by matching binary formats to existing templates. The OCR process begins with scanning a document into a digital picture. Following digitalization, OCR software analyzes the picture to identify letters, symbols, and punctuation marks using machine learning techniques and pattern recognition technologies. The detected characters are then subjected to a variety of algorithms to turn the picture into text. This involves knowing text structure and how it interacts with other page components like lines, paragraphs, and columns.

The next steps rely heavily on the CRAFT Algorithm. The Region Proposal Network (RPN) uses geometric clues and color variations to identify acceptable text areas. The Character Proposal Network (CPN) improves these ideas by predicting boundary boundaries for particular characters.

Non-max suppression removes overlapping or redundant character suggestions in mid-processing, while perspective correction corrects any skewed or distorted license plates, assuring proper character extraction. ResNet (Convolutional Neural Network) and LSTM (Long Short-Term Memory Network) work together to extract deep features from characters, focusing on spatial relationships and patterns, while LSTM analyzes the sequence of features, taking context and dependencies into account for improved character recognition. Greedy decoding comes next, in which character probability mapping takes place and the decoder iterates across locations, picking characters with the highest probabilities. Alternative decoding approaches, such as beam search or random sampling, may be considered for improved accuracy, although these incur a larger processing cost.

Post-processing determines the confidence score of the decoded license plate based on individual character probabilities and conformity with the predicted license plate format. In addition, error correction logic is used to identify and repair any problems, utilizing context information or a database of valid license plate forms. The system's final output is the recognized license plate number in textual format, which might be accompanied with a confidence score indicating the accuracy of the recognition [16].

Plate Recognition using CNN

Car plate identification with a CNN model is a two-step procedure that involves segmenting the plate picture into different characters before applying a CNN deep learning model to recognize the individual characters. Figure 10 illustrates the method. The proposed approach yields promising results for properly extracting and segmenting characters from license plate photos. Contour extraction successfully limits possible characters, while segmentation creates well-defined character pictures that can be recognized.

Figure 11 shows the contour extraction technique, which finds probable characters inside a binary picture. The procedure includes:

1. Locating all contours in a binary picture.

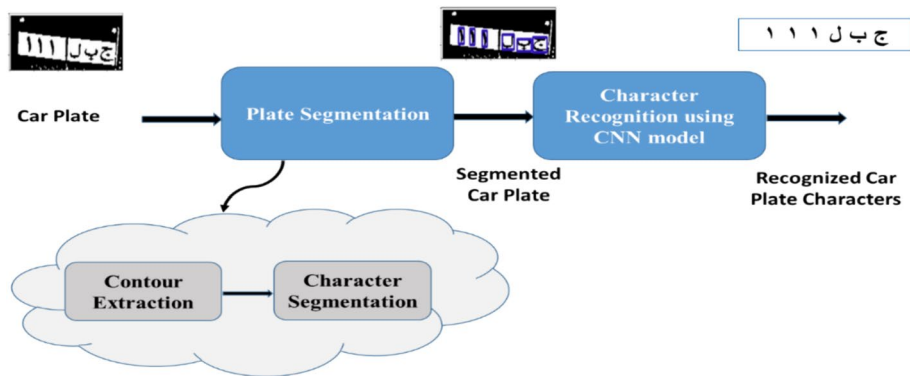


Fig. 10 Segmenting and recognition of car plate image

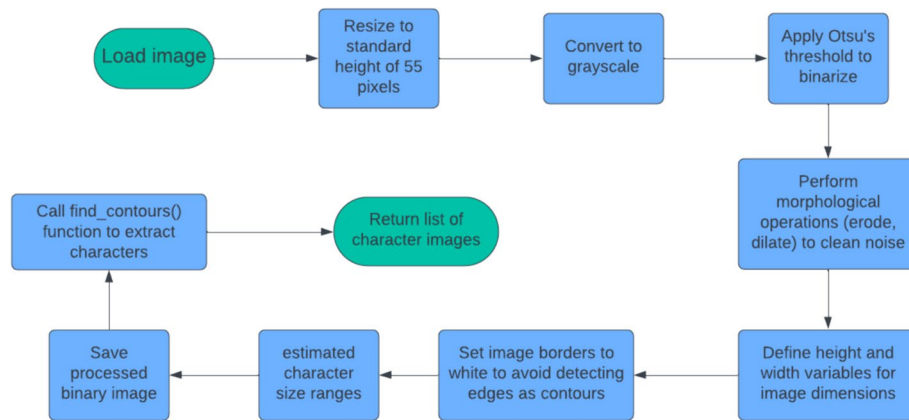


Fig. 11 Contour extraction algorithm

2. Filtering outlines according to given dimensions.
3. Selecting top contours based on area.
4. Extracting characters with bounding rectangles.
5. Preprocess each character picture for categorization.

The character segmentation method improves the segmentation process on a scaled license plate image:

1. Resized license plate image for consistency.
2. Converting image to greyscale.
3. Using thresholding to generate binary images.
4. Use erosion and dilation to remove unnecessary pixels.
5. Defining boundaries and maintaining color consistency.
6. Estimating license plate character contour dimensions.
7. Display the binary image.

An illustration of the two consecutive algorithms is given in Fig. 12. The presented approach showcases a robust method for license plate character recognition. The

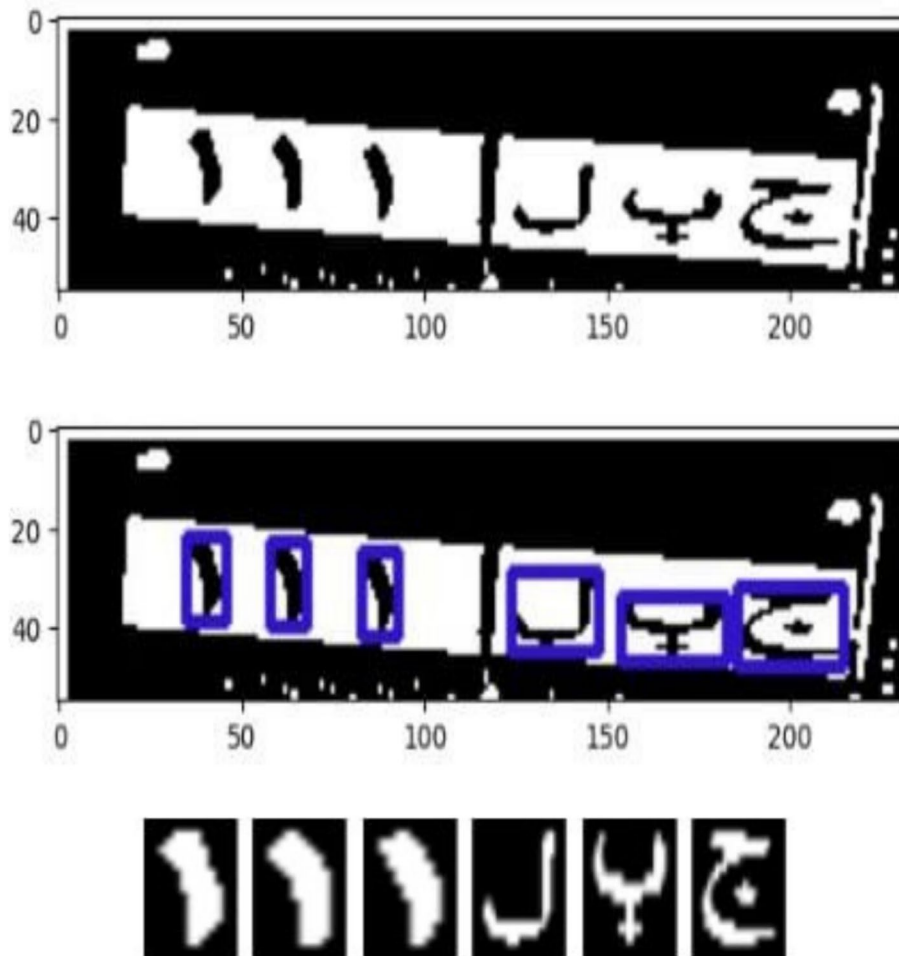


Fig. 12 Example of segmentation results

combination of contour extraction and character segmentation contributes to the accurate identification of characters within license plates. While the method exhibits success, further refinements and optimizations may enhance its performance in diverse scenarios.

The OCR component uses a convolutional neural network (CNN) model to extract text from recognized license plates. The CNN-based OCR methodology outperforms classic OCR approaches in terms of accuracy and resilience. The CNN model is especially intended to recognize Egyptian license plate characters, including Arabic digits and alphabets, as shown in Fig. 10. It is trained on a huge, representative dataset of around 96 K pictures, which includes various fonts, sizes, orientations, and styles of characters on Egyptian plates.

The convolutional neural network (CNN) model is the optical character recognition (OCR) workhorse in our system pipeline, detecting alphanumeric characters on license plates identified by YOLOv8. Our CNN technique provides considerable increases in accuracy, flexibility, and robustness compared to traditional OCR techniques for license plate recognition. It offers the following major advantages:

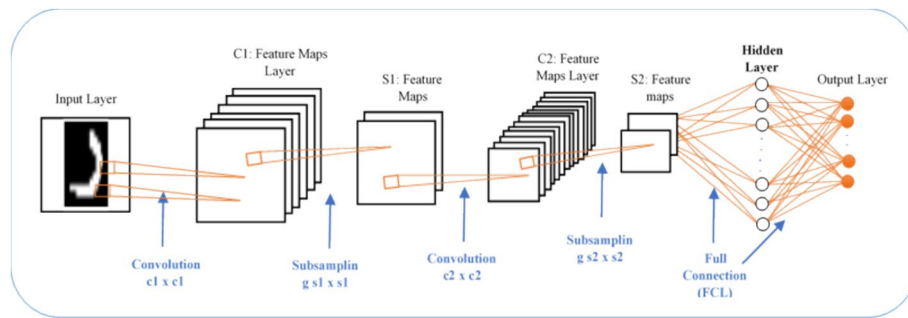


Fig. 13 CNN architecture for Arabic characters classification

- The CNN's convolutional layers automatically extract essential visual features for distinguishing Egyptian license plate characters, eliminating the need for human feature engineering.
- The CNN model learns representations of character styles, typefaces, and orientations from a varied dataset of plate pictures, allowing for accurate interpretation of a wide range of input patterns.
- The CNN's 2D structure maintains surrounding pixel correlations, allowing for greater recognition of forms and context in images compared to other OCR methods.
- Deep CNNs' hierarchical learning process improves generalization, allowing for accurate inference of previously unknown samples in real-world applications.
- Our training methodology and architecture are customized to recognize Egyptian license plate characters with peculiarities in Arabic script.

In essence, our CNN technique allows the system to 'understand' license plate images on a deeper level, resulting in increased accuracy, precision, and general robustness to real-world situations. Convolutional neural networks are hierarchical, multi-layer neural networks with a deep supervised learning architecture taught using the back-propagation method [34]. They include an automated feature extractor and a trainable classifier.

CNNs are used to learn complicated, high-dimensional data, and they differ in how convolutional and sub-sampling layers are examined. Several CNN designs are proposed for various challenges, including object identification [35] and handwritten digit/character recognition [36, 37]. The best result in the pattern recognition challenge was attained. In addition, to provide some degree of invariance to scale, shift, and distortion, CNN combines three major hierarchical aspects: local receptive fields, weight sharing, and spatial sub-sampling [38].

As shown in Fig. 13, the network represents a typical convolutional neural network architecture for handwritten character recognition. It includes a set of several layers. Initially, the input is convoluted with a set of filters (C hidden layers) in order to obtain the values of the feature map. Next, in order to diminish the dimensionality (S hidden layers) of the spatial resolution of the feature map, each convolution layer is pursued by a sub-sampling layer. Convolutional layers alternate sub-sampling layers constitute the feature extractor to retrieves discriminating features from the raw

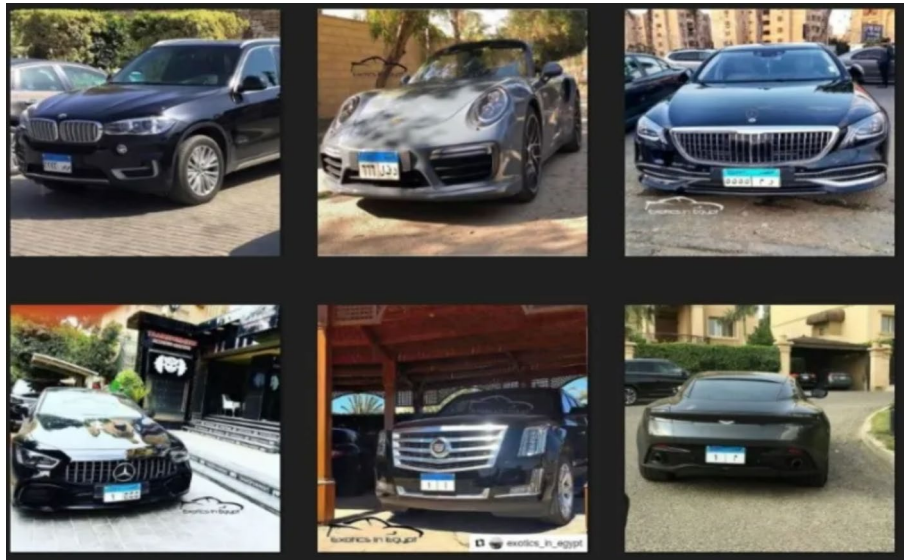


Fig. 14 Data sample of EALPR cars [9]

images. Ultimately, these layers were pursued by two fully connected layers (FCL) and the output layer. The output of the previous layer is taken by each layer as the input.

Experiments and results

In this section, we present the detailed experiments, datasets, and training process of the two phases of the proposed Egyptian car plate detection and recognition system. We separate each stage evaluation to indicate the effect of each stage on the overall system performance.

Experimental results on first phase: plate detection using YOLOv8

Detection dataset

To train and test the YOLOv8 model for detecting the Egyptian car plates through images, the EALPR dataset is utilized [9, 39]. It comprises 2450 images of Egyptian license plates taken in diverse conditions. Samples of the dataset images is given in Fig. 14. These images are annotated in the YOLO format, with a split of 80% for training and 20% for validation. The dataset aims to reflect real-world scenarios by including variations in lighting, perspectives, and plate sizes. The YOLO annotation format used defines bounding boxes around license plates. To enhance the model's adaptability, images are distributed across different scenarios, such as low-light conditions and reflections.

A careful quality control process was implemented, including manual annotation verification and removal of duplicate images. Furthermore, we carefully chose our dataset to include a variety of tough settings, such as low light, fog, and low-contrast situations. This large dataset enables our model to perform well in a wide range of real-world settings. This dataset's distinctiveness lies in its focus on Egyptian license plates, contributing to the model's efficacy in practical applications.

Table 3 Dataset class balance

Class	No. of images	No. of objects	Count on image average	Area on image average
License plate	2084	2140	1.03	2.22%

Table 3 reports the objects distribution in EALPR cars dataset [9], which indicate how many images are in the dataset with a certain number of objects of a specific class. For additional details on the EALPR dataset, including class sizes, object distribution, and licensing information, please refer to the dataset's official page [31]. This comprehensive resource provides researchers with a deeper understanding of the dataset's characteristics and ensures proper citation and acknowledgment in academic work.

Evaluation metrics and model results

We use the mean average accuracy (mAP) to evaluate the detection phase's performance which is based on YOLOv8. Mean average precision (mAP) is the weighted average of AP values from all sample categories, and it is used to quantify the model's detection accuracy in all categories, as indicated in Eq. (2):

$$\text{mAP} = \frac{1}{N} \sum_{i=1}^N \text{AP}_i \quad (2)$$

The AP_i signifies the AP value with category index value i , and N is the number of categories of the samples in the training dataset. In this study, N is 10. The average precision (AP) is equal to the area under the precision–recall curve, which is determined as:

$$\text{AP} = \int_0^1 \text{Precision (Recall)} d(\text{Recall}) \quad (3)$$

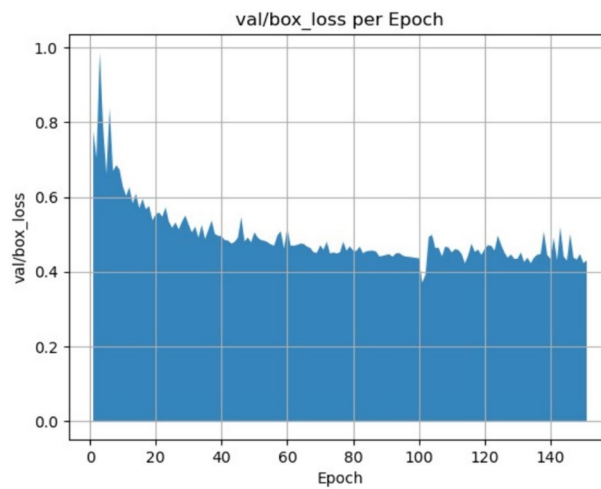
$\text{mAP}=0.5$ represents the average accuracy when the detection model's IoU is set to 0.5, and $\text{mAP}=0.5:0.95$ denotes the average accuracy when the detection model's IoU is set between 0.5 and 0.95. Table 4 presents the YOLOv8 model results, with a detection mAP of 94.265%.

The bounding box loss "box_loss" statistic measures the accuracy of bounding box predictions in a model. A smaller box_loss value point to more exact localization [40], demonstrating the model's ability to precisely identify regions of interest. It is an important metric for jobs involving object detection since it ensures proper placement of bounding boxes. The classification loss "cls_loss" assesses the accuracy of a model's classification. Low values of cls_loss indicate an improved classifications accuracy which is fundamental in evaluating the model's ability to recognize and classify distinct features within the input data.

The mean average accuracy for bounding box predictions "mAP50-95(B)" was obtained using intersection over union (IoU) criteria ranging from 0.5 to 0.95. It provides a detailed assessment of a model's accuracy in localizing objects inside pictures. A greater mAP50-95(B) value demonstrates consistent precision in object localization over a range of IoU thresholds, exhibiting the model's ability to properly anticipate object

Table 4 Yolov8 car plate detection model results

No. of epochs	train/box_loss	val/box_loss	train/cls_loss	val/cls_loss	metrics/recall(B)	metrics/precision(B)	metrics/mAP50-95(B)
10	0.6066	0.62901	0.42537	0.76719	0.92176	0.98724	0.82184
20	0.54877	0.55294	0.37001	0.32482	0.98044	0.99549	0.86741
30	0.53393	0.5267	0.34376	0.30638	0.98778	0.9967	0.88842
40	0.50157	0.49483	0.30499	0.27649	0.98778	0.9975	0.89512
50	0.47439	0.50549	0.28352	0.26983	0.99263	0.99754	0.90656
60	0.46432	0.51686	0.27219	0.25644	0.98778	0.99399	0.88897
70	0.44287	0.47053	0.25612	0.23451	0.99267	0.99687	0.9085
80	0.41716	0.45653	0.23923	0.21876	0.99267	0.99739	0.91257
90	0.39829	0.44486	0.22496	0.20898	0.99264	0.9951	0.91707
100	0.34673	0.43622	0.17728	0.19621	0.99261	0.99754	0.91626
110	0.5117	0.45134	0.32697	0.28441	0.98955	0.99794	0.90313
120	0.47706	0.46102	0.29044	0.24423	0.99274	0.99667	0.90239
130	0.45763	0.43443	0.27431	0.2356	0.98547	0.99754	0.91025
140	0.44504	0.43434	0.26595	0.22913	0.99031	0.99502	0.91669
150	0.4298	0.42294	0.25802	0.22231	0.98547	0.99701	0.91842

**Fig. 15** Validation box_loss per epoch

borders. This statistic is especially important for assessing the model's generalization and ability to handle changes in item size and location within the dataset.

In addition to numerical results, graphical representations offer a more comprehensive view of the model's performance during training. Figures 15 and 16 depict the validation losses for bounding box (val/box_loss) and class predictions (val/cls_loss) across different numbers of epochs. These visualizations provide insights into the convergence and stability of the model during the training process. A decreasing trend in both loss curves suggests effective learning, while fluctuations or sudden spikes may indicate potential challenges or overfitting. The following graphs illustrate

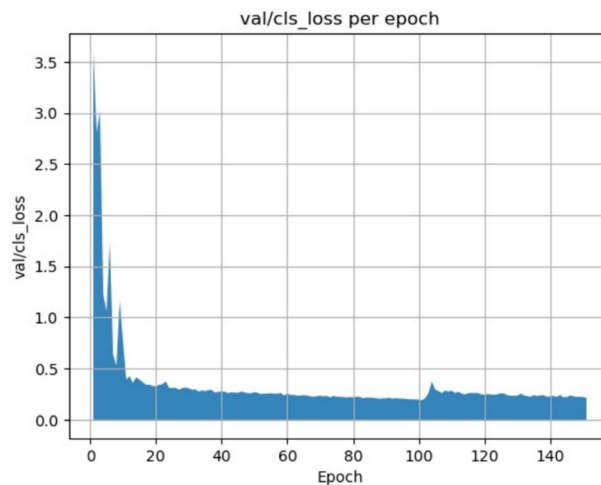


Fig. 16 Validation class_loss per epoch

the evolution of the validation losses over epochs, shedding light on the training dynamics and guiding our understanding of the model's optimization process.

Experimental results on second phase: plate recognition using CNN OCR

Recognition dataset

We incorporated the "Arabic Letters & Numbers OCR" dataset from Kaggle [41] to train the CNN model. The dataset consists of 39 classes, including 10 Arabic numbers and 29 other classes for the Arabic alphabet (as shown in Fig. 17). In total, it contains 96,759 images, divided into 2481 images in each category. We divided the data into 0.8 for training and 0.2 for test. Then, we resized the images to 100×100 pixels for each image. By leveraging this extensive dataset, we aimed to enhance the training of our convolutional neural network (CNN) for optical character recognition (OCR). The inclusion of Arabic characters is particularly valuable for our Egyptian car plate recognition project, as it ensures our model is adept at recognizing the specific linguistic nuances present on Egyptian license plates. While we initially experimented with Easy-OCR, we later endeavored to create our own version of OCR, integrating insights gained from training on this comprehensive Arabic dataset to tailor our OCR system for optimal performance in the context of Egyptian car plates [42]. We used patch size 64 and epochs 20.

The convolutional neural network (CNN) model architecture used for optical character recognition (OCR) is detailed in Table 5. The model consists of a series of convolutional layers interspersed with max pooling layers for downsampling, followed by fully connected dense layers culminating in a final softmax output layer for classification. Rectified linear unit (ReLU) activation is used for all hidden layers. As shown, we utilized a range of filter sizes, with earlier convolutional layers extracting low-level features and later layers building up higher-level abstractions. Max pooling was included to reduce spatial dimensions and introduce translational invariance.



Fig. 17 Sample of the dataset used for CNN OCR model [41]

Implementation details

The preprocessing phase meticulously standardizes image dimensions, executes grayscale conversion, applies a bilateral filter for noise mitigation, and employs edge detection algorithms, notably Canny, to precisely delineate license plate edges. YOLOv8, featured prominently, significantly contributes to plate localization through a comprehensive connected component analysis, accurately defining the rectangular region encapsulating the license plate. Subsequent skew correction, facilitated by YOLOv8, rectifies any potential tilt, ensuring optimal input for focused analysis.

Concurrently, the system enhances character differentiation by applying binarization to improve image contrast. Connected component analysis efficiently segregates individual characters on the plate, and feature extraction captures crucial characteristics, including bounding box size and aspect ratio, establishing a foundation for the

Table 5 CNN model architecture

Type	In/out size	Params	Module	Arguments
Conv	[100, 100, 1]	200	Conv	[3, 200, (3, 3), 'relu']
Conv	[100, 100, 200]	300	Conv	[200, 150, (3, 3), 'relu']
MaxPool2D	[50, 50, 150]	0	MaxPool2D	[4]
Conv	[50, 50, 150]	27,150	Conv	[150, 120, (3, 3), 'relu']
Conv	[50, 50, 120]	108,120	Conv	[120, 80, (3, 3), 'relu']
Conv	[50, 50, 80]	36,080	Conv	[80, 50, (3, 3), 'relu']
MaxPool2D	[12, 12, 50]	0	MaxPool2D	[4]
Flatten	[12, 12, 50]	0	Flatten	[]
Dense	[4800]	60,520	Dense	[4800, 120, 'relu']
Dense	[120]	12,120	Dense	[120, 100, 'relu']
Dense	[100]	10,100	Dense	[100, 50, 'relu']
Dense	[50]	4950	Dense	[50, 39, 'softmax']

subsequent integration with Easy-OCR. Tailored specifically for Egyptian license plate characters, Easy-OCR assumes a pivotal role in character recognition, leveraging the refined input from YOLOv8 to accurately decipher the alphanumeric sequence.

A noteworthy aspect of the system lies in its systematic storage of each detection in a structured database, this strategic approach ensures the archival of recognized license plate information, enabling seamless retrieval for subsequent reference and comprehensive analytical endeavors. The subsequent plate reconstruction phase adeptly amalgamates recognized characters, and a rigorous validation step safeguards the integrity of the recognized license plate number. YOLOv8's dual role in localization and image optimization, complemented by systematic database integration, significantly amplifies the overall efficacy of the system. The output manifests as a confidently recognized plate number, securely cataloged in a database, poised for seamless integration into applications such as traffic monitoring and security systems. This sophisticated amalgamation represents a robust implementation within the framework of license plate detection.

Model results

We used two metrics to quantify the results of the CNN OCR model: Precision and Recall. Precision is the ratio of the number of positive samples (TP) predicted by the model to the number of all detected samples, which include the true positive and false positive (FP) samples:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (4)$$

Recall is the ratio of the number of positive samples (correctly predicted by the model) to the number of positive samples that actually appeared. Recall is calculated as:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (5)$$

where FN is the false negative samples.

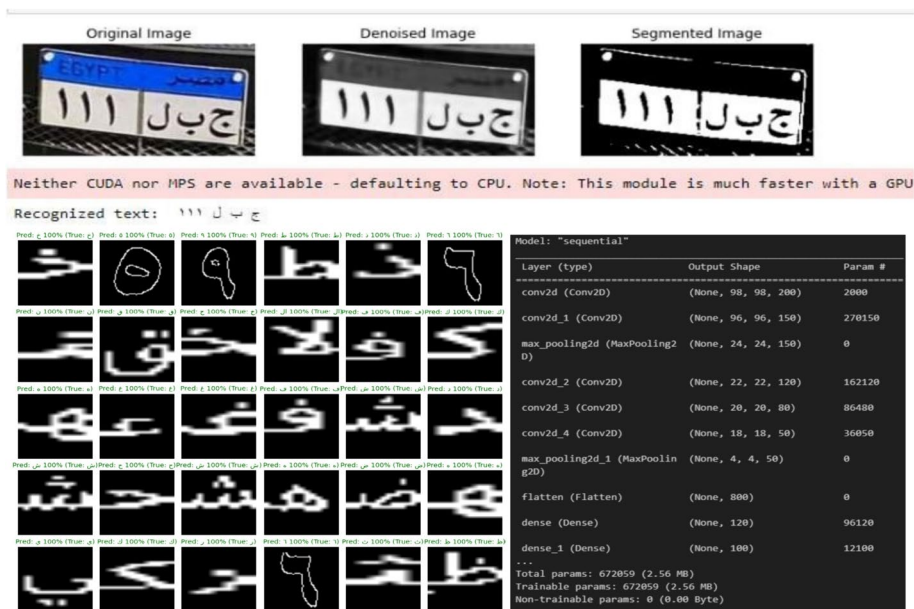


Fig. 18 Model results

Table 6 CNN plate character recognition model best results

No. of epochs	Training accuracy	Training loss	Validation accuracy	Validation loss
20	0.9942	0.0241	0.9944	0.0300

We evaluate the performance of LP-Net and Char-Net using the mean average precision. After training both networks with 150 epochs, the Val-Accuracy eventually reached 0.9944, and the Val-loss percentage was 0.0300. This percentage is the maximum we have reached after trying many times and many failures in training this huge amount of images. The image recognition rate finally reached 100 percent as shown in Fig. 18. The model’s results in terms of Accuracy = 94%.

Hyperparameter optimization experiments were conducted to improve performance of our CNN model. The number of training epochs, dropout regularization rate, and mini-batch size were systematically varied, and multiple evaluation metrics were monitored to assess impact on model skill. According to Table 6, results showed that training for 10 epochs with a dropout rate of 0.2 and batch sizes of 8 or 16 achieved strong performance across metrics, indicating excellent generalization with accuracy over 99%, AUC over 0.9998, and F1-score over 0.99. Precision and recall were balanced at 96–98%, and loss was acceptably low. The choice of a moderate dropout regularization likely prevented overfitting to the training data. Lower batch sizes appeared to positively impact model robustness to unseen data. In contrast, shorter 5 epochs training struggled to converge, while higher 0.5 dropout and batch size 32 conditions degraded performance due to underfitting and over smoothing respectively.

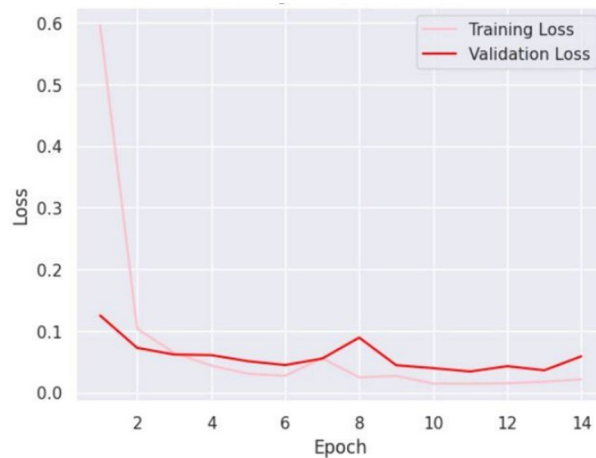


Fig. 19 Training and validation loss

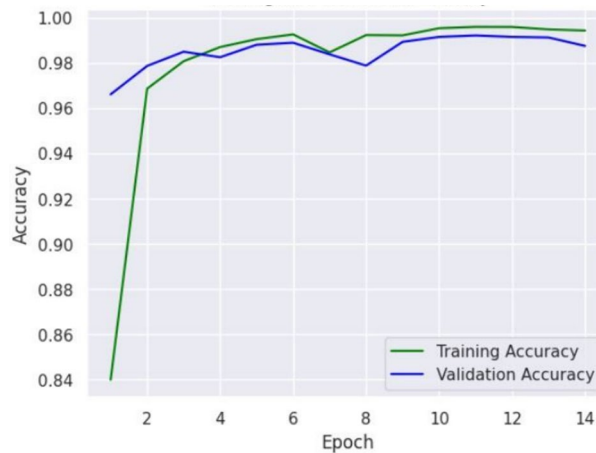


Fig. 20 Training and validation accuracy

The performance of the model was tested over 20 training epochs. According to Figs. 19 and 20 reporting key evaluation metrics on both the training data as well as a held-out validation set. Strong model performance, high accuracy, and low loss scores that generalize well from training to validation data, indicative of an effective and well-generalized model suitable for deployment. The results are given in Table 7.

After we finished training the model to recognize the characters, and before that we performed segmentation, we divide the car plate into parts, each part containing one character, then we search for each character using a loop for each of them, and as shown in Fig. 21, there are 3 letters and 3 numbers, so the model will have 6 loops. Each loop will search and identify each letter, and after completing them all, we will reach the result of the above as shown in Fig. 22. We have also compared the performance of the proposed Easy-OCR & CNN models and found out that Easy-OCR performance is only 94% accuracy, while CNN achieved 99.42% accuracy on the same dataset.

Table 7 CNN model results

epochs	dropout_ rate	batch_ size	Loss	Accuracy	AUC	F1	Precision	Recall	Specificity
5	0	4	0.071979411	0.957692308	0.999405	0.956466	0.93299	0.978378	0.998293
		8	0.051171385	0.975897436	0.999817	0.975737	0.972678	0.962162	0.999343
		16	0.031769846	0.988589744	0.999878	0.988608	0.973262	0.983784	0.999343
		32	0.057965472	0.981666667	0.999854	0.981496	0.989011	0.972973	0.999737
	0.2	4	0.139418855	0.976538462	0.999727	0.976657	0.983425	0.962162	0.999606
		8	0.119022809	0.982948718	0.999849	0.982922	0.983696	0.978378	0.999606
		16	0.111460537	0.978589744	0.999801	0.978709	0.989071	0.978378	0.999737
		32	0.123060733	0.983461538	0.999875	0.983462	0.983871	0.989189	0.999606
	0.5	4	0.827011406	0.902820513	0.997631	0.894894	0.76652	0.940541	0.99304
		8	0.685262978	0.923974359	0.997964	0.916371	0.719828	0.902703	0.991464
		16	0.848628819	0.908461538	0.998222	0.903997	0.770563	0.962162	0.99304
		32	0.980712116	0.886410256	0.997367	0.864714	0.950276	0.92973	0.998818
10	0	4	0.055446621	0.986794872	0.999871	0.986773	0.973118	0.978378	0.999343
		8	0.035086103	0.976666667	0.999679	0.975534	0.978723	0.994595	0.999475
		16	0.018043647	0.987307692	0.99986	0.987277	0.983957	0.994595	0.999606
		32	0.027643941	0.988589744	0.999905	0.988581	0.994536	0.983784	0.999869
	0.2	4	0.088619314	0.983333333	0.999825	0.983376	0.942105	0.967568	0.998555
		8	0.062872604	0.991025641	0.999869	0.991032	1	0.989189	1
		16	0.059801929	0.989615385	0.999936	0.989617	0.994536	0.983784	0.999869
		32	0.060563173	0.984102564	0.999902	0.984172	0.962567	0.972973	0.999081
	0.5	4	0.611292303	0.926025641	0.998263	0.917574	0.943503	0.902703	0.998687
		8	0.461384475	0.939487179	0.998757	0.930742	0.900524	0.92973	0.997505
		16	0.530284286	0.939615385	0.998692	0.931268	0.976879	0.913514	0.999475
		32	0.543869913	0.948461538	0.998861	0.942719	0.952381	0.972973	0.998818
15	0	4	0.089574672	0.982179487	0.999864	0.98225	0.983871	0.989189	0.999606
		8	0.071143515	0.988717949	0.999911	0.988722	0.994444	0.967568	0.999869
		16	0.020087067	0.988333333	0.999882	0.988302	0.978495	0.983784	0.999475
		32	0.029663144	0.982179487	0.999622	0.982193	0.918782	0.978378	0.997899
	0.2	4	0.19821614	0.98474359	0.999837	0.98473	1	0.967568	1
		8	0.06231042	0.988205128	0.99988	0.988171	0.967914	0.978378	0.999212
		16	0.048448678	0.989615385	0.9999	0.989617	0.989247	0.994595	0.999737
		32	0.036850434	0.991410256	0.99994	0.991431	0.989189	0.989189	0.999737
	0.5	4	0.662570536	0.885769231	0.996991	0.869531	0.87574	0.8	0.997242
		8	0.454496354	0.939871795	0.99865	0.932206	0.956044	0.940541	0.998949
		16	0.5698722	0.902948718	0.997144	0.889991	0.957895	0.983784	0.998949
		32	0.510036588	0.931538462	0.998719	0.92192	0.912371	0.956757	0.997768

Comparison between deep learning models

We benchmarked several deep convolutional neural network architectures on our task, with key training statistics reported in Table 8. DenseNet121 achieved the lowest training loss of 0.6395, indicating excellent convergence, while VGG-16 suffered from high overfitting with a sizable gap between train accuracy (86%) and validation (52%). ResNet-50 was competitively accurate on the train set (90%) but generalizability remained poor per validation metrics. In contrast, the CNN model demonstrated state-of-the-art performance, attaining 99.4% validation accuracy with negligible overfitting and just 0.03 validation loss. This result can be attributed to sensible hyperparameter selection of 20 training epochs and 64 batch size, allowing sufficient



Fig. 21 CNN model segmentation result

model complexity without information loss from excessive coalescing batch statistics. As concluded from the table, While DenseNet surpassed CNN on one metric.

The overall results suggest the compact CNN architecture generalized the best for this problem, reaching accuracy and loss parity between train and validation partitions unseen during optimization. Further lifestyle gains may be attainable for the CNN via additional hyperparameter tuning or through ensemble techniques.

We also compared our work against state-of-the-art approaches [10, 18] on the same dataset as reported in Table 9. The results indicate the superiority of our approach which reached 99% accuracy. Finally, Table 10 gives a visual sample of a comparison held between the two proposed models for two randomly selected car plates. The results are shown through our interface to the system we built for our purposes. The two approaches steps are different in their extraction to the characters and results as clearly shown in Table 10.

Conclusion

This paper has developed an Egyptian car plate recognition system leveraging YOLOv8, Easy-OCR, and CNN technologies. Through customization and rigorous experimentation, we have crafted an intelligent system capable of efficiently detecting and recognizing license plates in diverse real-world conditions. Our implementation of YOLOv8 optimized for plate detection with accuracy 94.2%, enabled by a tailored dataset of localized Egyptian plates. Easy-OCR and CNN integration further enriches recognition capabilities. The additional refinement of segmentation isolates characters, elevating accuracy. Extensive comparative analysis revealed optimal configurations, culminating in the unified architecture presented. The system exhibits remarkable precision on test datasets, correctly extracting over 99% with CNN and 94% with Easy-OCR of text from detected plates. Augmentation and hyperparameter tuning continue to drive enhancements.

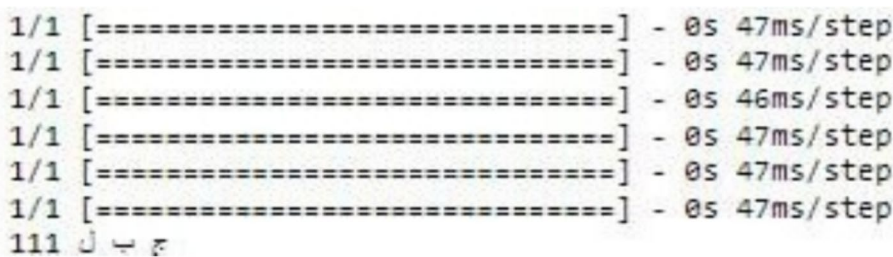


Fig. 22 CNN model prediction result

Table 8 CNN models results

Model	No. of epochs	Batch size	Loss	Accuracy	Val-loss	Val-accuracy
CNN	20	64	0.0241	0.9942	0.0300	0.9944
VGG-16	30	32	0.3688	0.8607	1.5971	0.5241
ResNet-50	100	32	0.3199	0.9016	3.3224	0.6979
DenseNet121	20	32	0.6395	0.8014	0.4345	0.8879

Table 9 Comparison to state-of-the-art on same dataset

Paper	Detection algorithm	Detection accuracy (%)	Character recognition algorithm	Character recognition accuracy (%)
Proposed YOLOv8, Easy-OCR and CNN	YOLOv8	94.265	Convolutional neural network (CNN)	99.42
Shehata et al. [18]	Faster RDNN	90	Connected components labeling (CCL)	96
Youssef et al. [10]	YOLOv3	97.89	Convolutional Neural Networks (CNNs)	92.46

The paper’s contributions span multiple domains. For transportation, our system facilitates automation in critical applications like electronic tolling, parking access

Table 10 Sample of Comparison of the two proposed models

Easy OCR	CNN
<p>Original Image Denoised Image Segmented Image</p> <p>Neither CUDA nor MPS are available - defaulting to CPU. Note: This module is much faster with a GPU</p> <p>Recognized text: ج ب ل ١١١</p>	<pre>In [28]: 1 for i in range(len(char)): 2 plt.subplot(1, 10, i+1) 3 plt.imshow(char[i], cmap='gray') 4 plt.axis('off')</pre> <p>1/1 [=====] - 0s 64ms/step 1/1 [=====] - 0s 47ms/step 1/1 [=====] - 0s 47ms/step 1/1 [=====] - 0s 46ms/step 1/1 [=====] - 0s 47ms/step 1/1 [=====] - 0s 47ms/step 1/1 [=====] - 0s 47ms/step 111 ج ب ل</p>
<p>Original Image Denoised Image Segmented Image</p> <p>Neither CUDA nor MPS are available - defaulting to CPU. Note: This module is much faster with a GPU.</p> <p>Recognized text: ر و ي ٤٨٥١</p>	<pre>In [12]: 1 for i in range(len(char)): 2 plt.subplot(1, 10, i+1) 3 plt.imshow(char[i], cmap='gray') 4 plt.axis('off')</pre> <p>1/1 [=====] - 0s 64ms/step 1/1 [=====] - 0s 47ms/step 1/1 [=====] - 0s 47ms/step 1/1 [=====] - 0s 46ms/step 1/1 [=====] - 0s 47ms/step 1/1 [=====] - 0s 47ms/step 1/1 [=====] - 0s 47ms/step ٤٨٥١ ر و ي</p>

control, and traffic analysis. In law enforcement, it aids investigations through indexes of vehicle sightings and identification of suspects. The security sector stands to gain from heightened monitoring abilities.

This initiative also holds deeper societal implications. By harnessing AI to extract insights from visual data, it accelerates Egypt's technological evolution. The economic benefits cannot be ignored either, saving substantial costs in manual administrative tasks. Environmentally, optimizing traffic flows reduces congestion and emissions. The surface has only been scratched. The system can be expanded upon by incorporating additional databases, new algorithms as they emerge, and bigger datasets. Transfer learning presents opportunities to adapt the models for new languages and regions.

At the heart of this project lies the firm belief that artificial intelligence, when ethically employed, holds immense potential in uplifting societies. We stay committed to this vision. The present work contributes a building block, demonstrating Egyptian innovation that brings positive transformation.

Author contributions

AS, RA, and AA contributed to conceptualization of this study and added the basic ideas writing and editing the manuscript. BD and AS contributed to formal analysis, methodology, software, validation, and writing original draft preparation. SH, AB, MR, RA, AA, AS, and AS contributed to Investigation, methodology, and validation. All authors read and approved the final manuscript.

Funding

No funding is available.

Availability of data and materials

The dataset used in this paper is publicly available online <https://www.kaggle.com/datasets/mahmoudebase/egyptian-cars-plates>, <https://www.kaggle.com/datasets/mahmoureda55/arabic-letters-numbers-ocr/data>.

Declarations

Competing interests

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Received: 24 June 2024 Accepted: 4 August 2024

Published online: 12 August 2024

References

1. Laroca R, Severo E, Zanlorensi LA, Oliveira LS, Gonçalves GR, Schwartz WR, Menotti D (2018) A robust real-time automatic license plate recognition based on the YOLO detector. In: 2018 International joint conference on neural networks (IJCNN). IEEE, pp 1–10
2. Silva SM, Jung CR (2017) Real-time Brazilian license plate detection and recognition using deep convolutional neural networks. In: SIBGRAP
3. Adewopo VA, Elsayed N, ElSayed Z, Ozer M, Abdelgawad A, Bayoumi M (2023) A review on action recognition for accident detection in smart city transportation systems. *J Electr Syst Inf Technol* 10(57):66
4. Kim HH, Park JK, Oh JH, Kang DJ (2017) Multi-task convolutional neural network system for license plate recognition. *Int J Control Autom Syst* 15:2942–2949
5. Jung CR, Montazzoli S (2017) Real-time Brazilian license plate detection and recognition using deep convolutional neural networks. In: 30th SIBGRAP conference on graphics, patterns and images (SIBGRAP), IEEE, pp 55–62
6. Gong Y, Deng L, Tao S, Lu X, Wu P, Xie Z, Ma Z, Xie M (2022) Unified Chinese license plate detection and recognition with high efficiency. *J Vis Commun Image Represent* 86:66
7. Du S, Ibrahim M, Shehata M, Badawy W (2012) Automatic license plate recognition (ALPR): a state-of-the-art review. *IEEE Trans Circuits Syst Video Technol* 23(2):311–325
8. Basalamah SM (2013) Saudi license plate recognition. *Int J Comput Electr Eng* 66:1–4
9. Bahaa M Egyptian Cars Plates. <https://www.kaggle.com/datasets/mahmoudebase/egyptian-cars-plates>
10. Youssef AR, Ali AA, Sayed FR (2022) Real-time Egyptian license plate detection and recognition using YOLO. *Int J Adv Comput Sci Appl* 13(7):66
11. Alghyaline S (2022) Real-time Jordanian license plate recognition using deep learning. *J King Saud Univ Comput Inf Sci* 34:66

12. Hatami S, Sadedel M, Jamali F (2023) Iranian license plate recognition using a reliable deep learning approach, arXiv preprint [arXiv:2305.02292](https://arxiv.org/abs/2305.02292)
13. Sun Z, Bebis G, Miller R (2006) On-road vehicle detection: a review. *IEEE Trans Pattern Anal Mach Intell* 7(5):694–711
14. Youssef SM (2008) A smart access control using an efficient license plate location and recognition approach. *Expert Syst Appl* 34(1):256–265
15. Jocher G et al (2022) Ultralytics YOLOv8: scaling object detection beyond 10K Classes
16. JaidedAI (2023) Easy-OCR [Software]. GitHub. <https://github.com/JaidedAI/EasyOCR>
17. Elnashar M, Hemayed EE, Fayek MB (2020) Automatic multi-style Egyptian license plate detection and classification using deep learning. In: 16th International computer engineering conference (ICENCO), Cairo, Egypt, pp 1–6. <https://doi.org/10.1109/ICENCO49778.2020.9357371>
18. M Shehata, MT Abou-Kreisha, H Elnashar (2021) Deep machine learning based Egyptian vehicle license plate recognition systems. In: Alazhar engineering 15th international conference, Egypt
19. Antar R, Alghamdi S, Alotaibi J, Alghamdi M (2022) Automatic number plate recognition of Saudi. *Eng Technol Appl Sci Res* 12(2):8266–8272
20. Yaman S, Erol Y (2023) MVSR normalization algorithm method for improving vehicle license plate recognition. *Turk J Sci Technol* 18(2):543–552
21. Islam D, Mahmud T, Chowdhury T (2023) An efficient automated vehicle license plate recognition system under image processing. *Indones J Electr Eng Comput Sci* 29(2):1055–1062
22. Silva SE, Jung CR (2018) License plate detection and recognition in unconstrained scenarios. In: Proceedings of the European conference on computer vision (ECCV), pp 580–596
23. Kim S, Daechul K, Ryu Y, Kim G (2002) A robust license-plate extraction method under complex image conditions. In: International conference on pattern recognition, vol 3. IEEE, pp 216–219
24. Zhu Y et al (2018) Vehicle license plate recognition based on convolutional neural networks. *IEEE Photonics J* 10(2):1–11
25. Sighthound-Automatic License Plate Recognition System. Sighthound, Inc., www.sighthound.com/solutions/automatic-license-plate-recognition-system
26. Mb Q, Salman TM (2023) License plate recognition in slow motion vehicles. *Bull Electr Eng Inform* 12(4):2236–2244
27. Moussaoui H, El Akkad N, Benslimane M (2023) Arabic and Latin license plate detection and recognition based on YOLOv7 and image processing methods. In: Researchsquare
28. Smith R (2007) An overview of the tesseract OCR engine. In: Ninth international conference on document analysis and recognition (ICDAR 2007), vol 2. IEEE
29. R Joseph, S Divvala, R Girshick, A Farhadi (2016) You only look once: Unified, real-time object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 779–788
30. Li H, Liu H (2018) License plate detection using yolo v2
31. Terven JR, Cordova-Esparza DM (2023) A comprehensive review of YOLO: from YOLOv1 and beyond, arXiv preprint [arXiv:2304.00501](https://arxiv.org/abs/2304.00501)
32. LeCun Y, Huang FJ, Bottou L (2004) Learning methods for generic object recognition with invariance to pose and lighting. In: Computer vision and pattern recognition conference (CVPR). IEEE
33. Ranzato M, Huang F, Boureau Y, LeCun Y (2007) Unsupervised learning of invariant feature hierarchies with applications to object recognition. In: Computer vision and pattern recognition conference (CVPR). IEEE
34. Sayed SA, Abdel-Hamid Y, Hefny HA (2023) Artificial intelligence-based traffic flow prediction: a comprehensive review. *J Electr Syst Inf Technol* 10(13):62
35. Jocher G, Chaurasia A, Qiu J (2023) YOLO by Ultralytics. <https://github.com/ultralytics/ultralytics>
36. Nguyen D et al (2020) A real-time and integrated automatic license plate recognition system using deep learning based on YOLOv3 and CNN. *Appl Syst Innov* 3(3):66
37. Sochor J et al (2018) Automatic license plate recognition using convolutional neural networks. In: IEEE international conference on intelligent transportation systems (ITSC). IEEE
38. Reis D, Kupec J, Hong J, Daoudi A (202) Real-time flying object detection with YOLOv8, arXiv preprint [arXiv:2305.09972](https://arxiv.org/abs/2305.09972)
39. Youssef AR, Sayed FR, Ali AA (2022) A new benchmark dataset for Egyptian license plate detection and recognition. In: 7th Asia-Pacific conference on intelligent robot systems (ACIRS), pp 106–111. <https://doi.org/10.1109/ACIRS55390.2022.9845514>
40. Zhanga YF, Renc W, Zhanga Z, Jiaa Z, Wang L, Tana T (2022) Focal and efficient IOU loss for accurate bounding box regression. In: CRIPAC & NLPR, CASIA, Beijing, China University of Chinese Academy of Sciences, Beijing, China Horizon Robotics
41. Bahaa M Arabic Letters & Numbers OCR. <https://www.kaggle.com/datasets/mahmoudreda55/arabic-letters-numbers-ocr/data>
42. Soo CP (2017) Segmentation-free license plate recognition using deep neural networks. Ph.D. Thesis

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.