Project Title

**Class** : \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Project Participants** : Abolfazl Najar

**Student ID** : \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Date** : 4/26/25

# Introduction

#### General Problem Description

In this project, I want to work on the OCR of Iranian plates. We know that there are lots of different types of license plates all over the world. In Middle east, they usually use license plate with Arabic or Persian numbers. So, OCR of these plates would be an important concept. Since there has been lots of effort, there is still room for improvement in plate OCR.

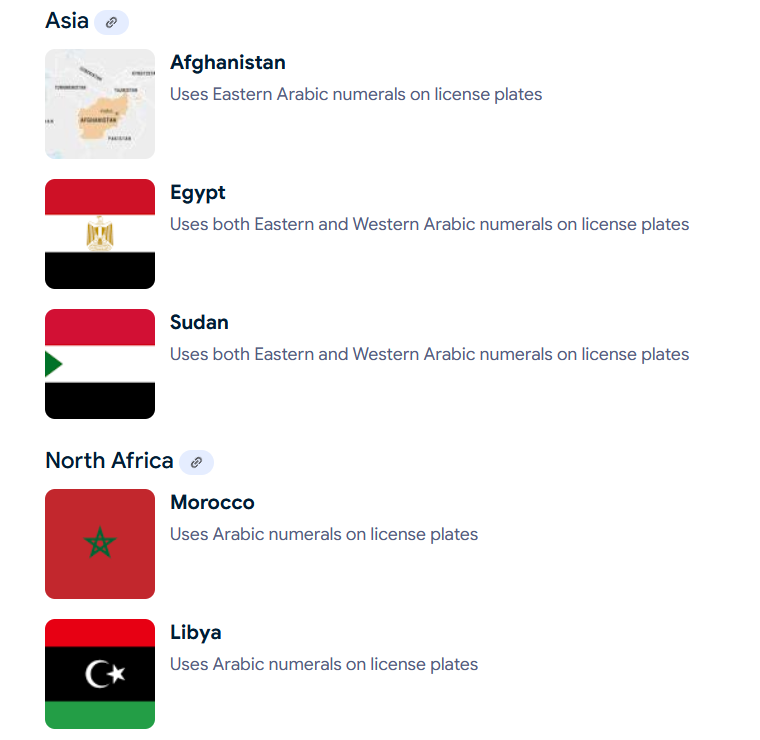
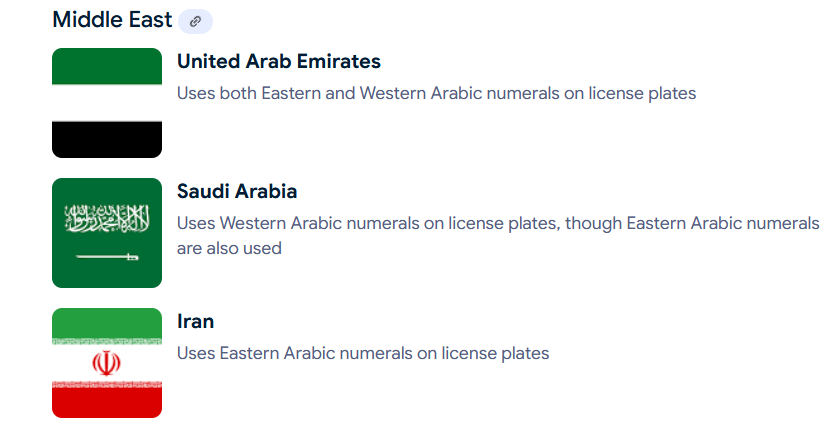
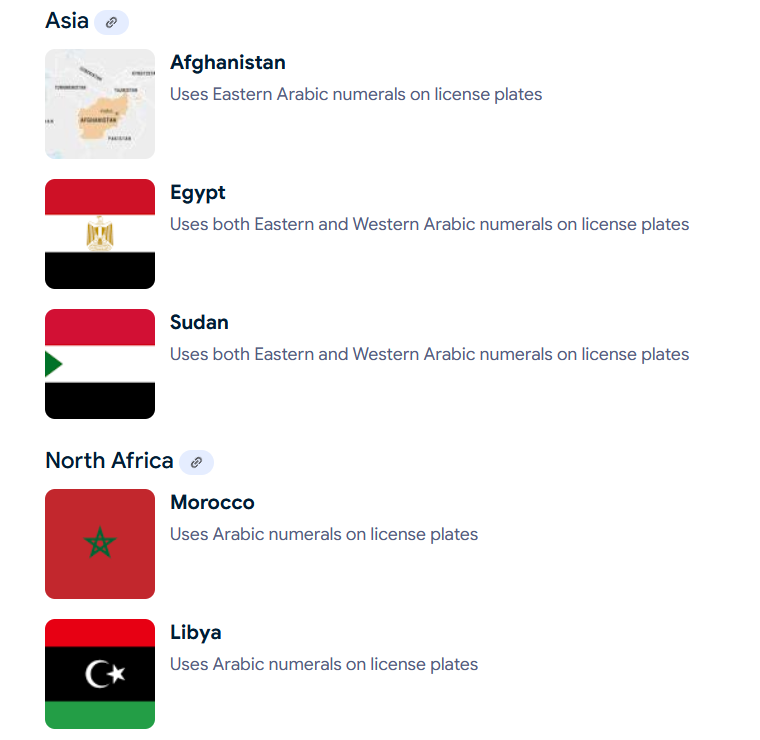
A close-up of a license plate

AI-generated content may be incorrect.

A collage of different numbers and letters

Description automatically generated

Many countries which are listed below use plates with same numbers so the project can be extended to other countries.



Some of the data we need to work on it are shown in the following image.



These projects could have several challenges. The main challenges could be:

* There are limited number of dataset public. It could make this project a little bit hard.
* The public dataset is not usually extracted by normal surveillance cameras which increase the diversity of datasets.
* There are some characters such as 2 and 3 which are hard to distinguish them.
* Some fundamental challenges such as illumination, resolution, and the dirt of plates could affect the results so much.

#### What Makes This Project Unique?

* **Databases (DBs):** There are some databases online, but each one has different labeling methods. First, I start with the following one and I would be thinking of combining other databases if needed.

https://universe.roboflow.com/shahab-jafari-1vorv/persian-plate-characters-mvinj/dataset/4

* **Algorithms:** I’m planning to use YOLOv11 pretrained model for OCR of plates characters. Using the latest pretrained models to achieve better results could help us to achieve better OCR in comparison with other schemes. After reaching good accuracy we can test different modifications for increasing model performance.
* **Solution/Approach:** Since Image processing and OCR problems are well-known problems and there are numerous papers that have been published in this aspect. So, we are going to follow the same path, and we would not build solutions on scratch.

#### Your Contributions

* **Databases:**
  + I’m trying to use and dataset which is published online, but I will try to add more data to it during the project. There could be different source of dataset, but their labeling is different, so it needs to work on it.
* **Code:**
  + The online resources would be monitored to get inspiration for training and evaluation pipelines.
  + The main plan is to use novel approaches, because building a new model from scratch is not reasonable. One of the latest models in image processing is YOLOv11. It has some advantages which make it suitable for real-time image processing purposes.

# Literature Review

Samadzade et. al [1] proposed method consists of three stages of neural networks connected in a modular fashion. The first stage is the detection of license plates (LP); after that, RILP proceeds to detect text regions and performs character segmentations. Finally, to get to the LP number, optical character recognition (OCR) is done via a neural network previously trained for recognition of Persian characters.

Rahmani et. al [2] addresses the need for high-quality datasets in automatic license plate recognition (ALPR), particularly in Iran. It highlights the limitations of existing datasets and introduces a new, comprehensive dataset containing 20,967 car images annotated for license plate detection and 27,745 images for character recognition. This dataset is designed to support deep learning-based object detection methods and facilitate advancements in both security applications and business opportunities related to ALPR.

Alborzi et. al [3] presents an Automatic License Plate Recognition (ALPR) system designed for unsupervised parking lot applications, optimized for embedded devices like the Raspberry Pi 3. The system consists of two main stages: license plate detection using the Single Shot Detection (SSD) architecture with MobileNet and Optical Character Recognition (OCR) using LPRNet. The method is computationally efficient, robust, and performs in real-time, achieving 79.86% end-to-end accuracy. To train the OCR model, the authors generated 130k synthetic license plate images and introduced a dataset of 1,500 real images under various conditions

Tourani et. al [4] The paper introduces IRANIS, a large-scale dataset specifically designed for Farsi character recognition in Iranian car license plates. It addresses the lack of publicly available datasets for this purpose, which is crucial for developing deep learning-based systems for law enforcement and surveillance. IRANIS contains over 83,000 images of Farsi numbers and letters, captured under varying conditions of angle, illumination, resolution, and contrast. The dataset is manually annotated for object detection and classification. Additionally, the authors provide a baseline performance analysis using YOLO v3 for Farsi character recognition.

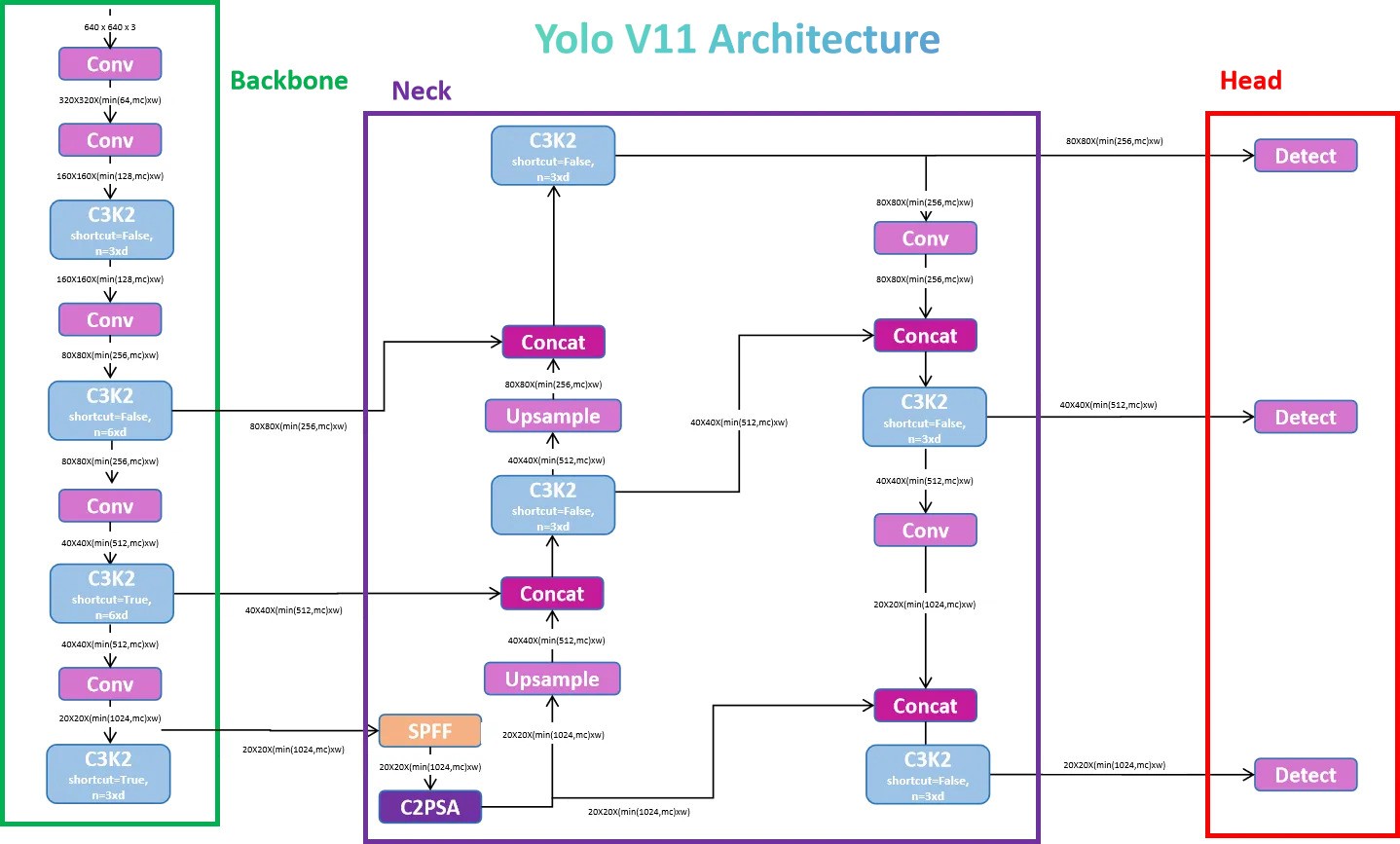
Rashtehroudi et. al [5] paper presents an Automated License Plate Recognition (ALPR) system designed for Iranian license plates using deep learning techniques. The system follows three main stages: (1) License Plate Localization, (2) Character Segmentation, and (3) Optical Character Recognition (OCR). Each stage is optimized to handle real-world challenges such as varying lighting conditions and viewing angles. The proposed method enhances accuracy and robustness, making it suitable for applications in law enforcement, surveillance, and automated parking systems.

Baharlou et. al [6] introduces an advanced approach to license plate recognition. The authors propose a novel feature extraction method combined with deep learning techniques to enhance the accuracy and robustness of the recognition process. This method addresses common challenges such as varying lighting conditions, occlusions, and diverse license plate formats. The proposed model demonstrates significant improvements in recognition performance compared to traditional methods, making it suitable for applications in traffic monitoring and automated vehicle identification systems.

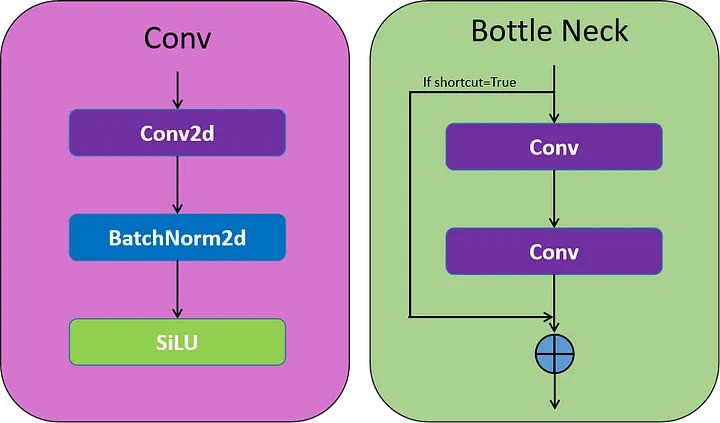
# Methodology

#### Tools

* **Programming Languages and Tools:**
  + For programming tools, we are using python environment to use different packages for Image processing.
  + Yolo (You Only Look Once) as a powerful tool for image processing can extract features from images and do several tasks such as detection, classification, and segmentation. In the past years, several versions of Yolo have been introduced, and we are going to use the latest version named Yolov11 for our project. Our plan is to fine-tune the yolo on our database and if we need more data, we will search for it online and improve our solution.
  + OpenCV toolbox is a powerful tool for image handling purposes. We are going to use it on this project. Other tools also would be used if we needed them. For example, API can be used to make a service provider over a port to send process requests on data.
* **Algorithms:**
  + The YOLO algorithm has been improved over the last years to improve feature extraction ability to be adaptive for different usage.

****

* + Yolo is constructed from 3 main parts which are backbone, neck, and head. One of the most interesting parts in Yolo structure is its residual connections which are used in different parts.
  + This is a sequence of convolutional block with a shortcut parameter, this would decide if you wanted to get the residual part or not. It is like the ResNet Block, if shortcut is set to False then no residual would be considered.



I have used yolo11s to fine-tune and test my algorithm on it.

* **Overall Approach:**

The most important part of our approach is fine-tuning the model to be familiar with real world application. Since we may have a limited dataset in this aspect, our model may perform differently in case of facing new data. Overall, the steps of doing our project would be like:

1. Dataset cleaning: in this step, we should be sure about the data and clean any irrelevant data which may affect our model performance.
2. Model fine-tuning: fine-tune the Yolov11 for detecting characters and classifying them. This step would be tricky since we are facing with imbalance dataset which alphabets in plates are much less than numbers and their detection may be hard.
3. Model Evaluation: to determine the performance of fine-tuned models on our real-world, we use test datasets and other online resources to do the evaluation.
4. End-to-end API: we are planning to add the model to an API for processing real-time plate recognition tasks

* **Significance of the Approach:**
  + Our scheme is effective since it can perform character detection and OCR on the same time would be suitable for real world applications.
  + Using Yolo make our approach implementation easier since it is fast and designed for fast inference along with high accuracy.

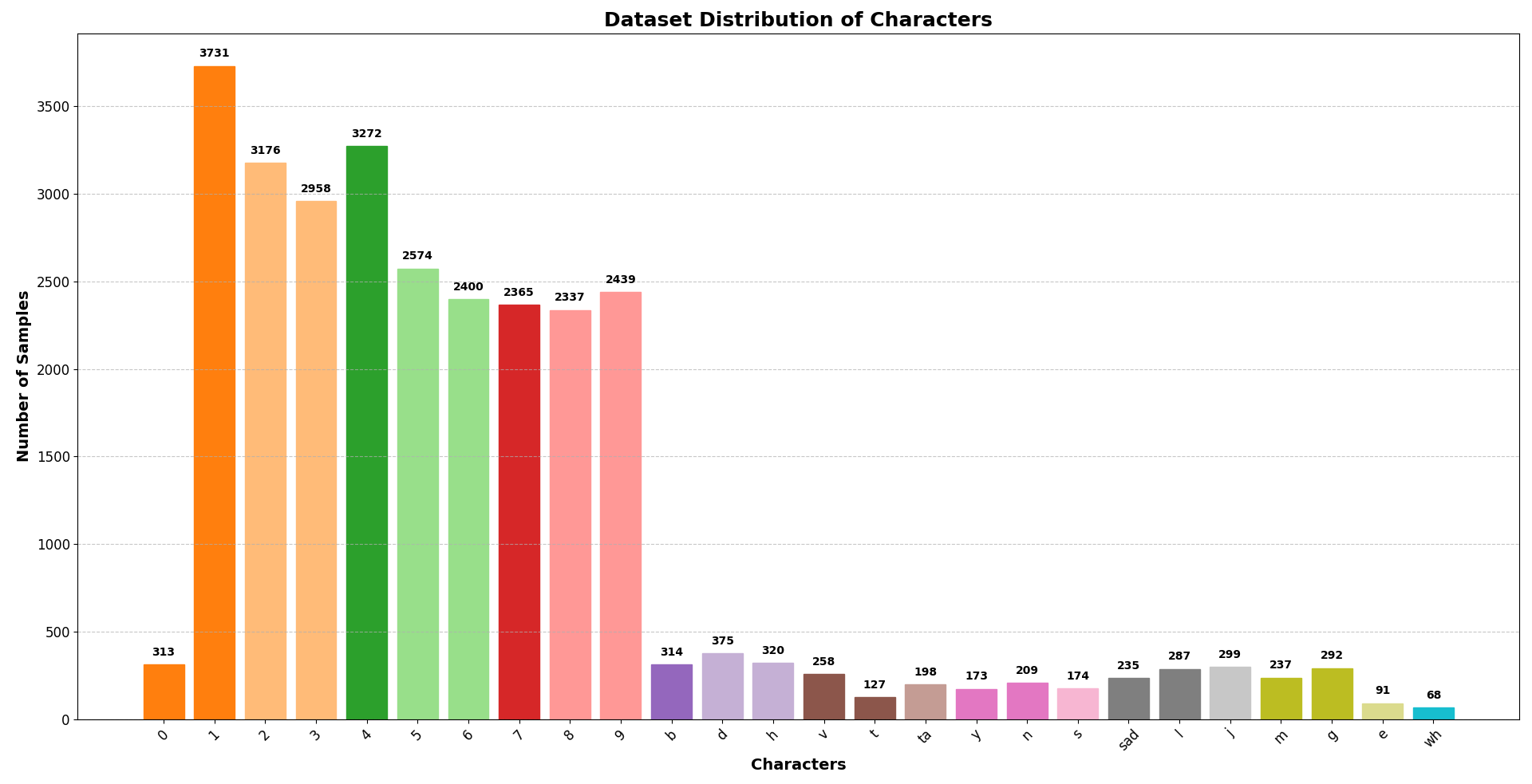
# Experiments and Results

* **Database Description:**

**The database that I’m using is called Persian Plate Characters. It is available online** <https://universe.roboflow.com/shahab-jafari-1vorv/persian-plate-characters-mvinj>

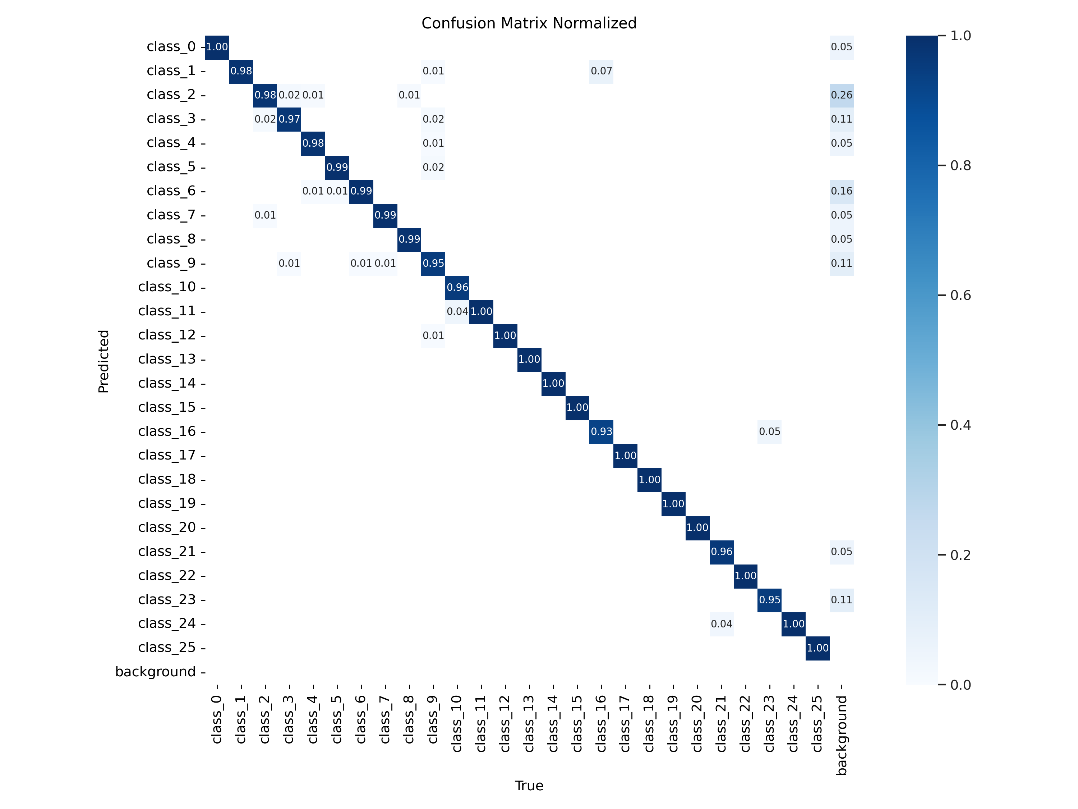
* 1. This database contains 3300 trains, 153 tests, and 231 validation images.
  2. Data has a good diversity from different plates in terms of taking pictures from different angles and variation in resolution and characters.
  3. It has different characters for classification. Also, we would be able to add other databases in case we have any limitations.
  4. It seems that an augmentation is applied to each image already and it tripled the amount of data in database.

FYI: After reviewing database I deleted one of classes and overall classes has been 26. The number of characters in modified dataset is illustrated in following image.

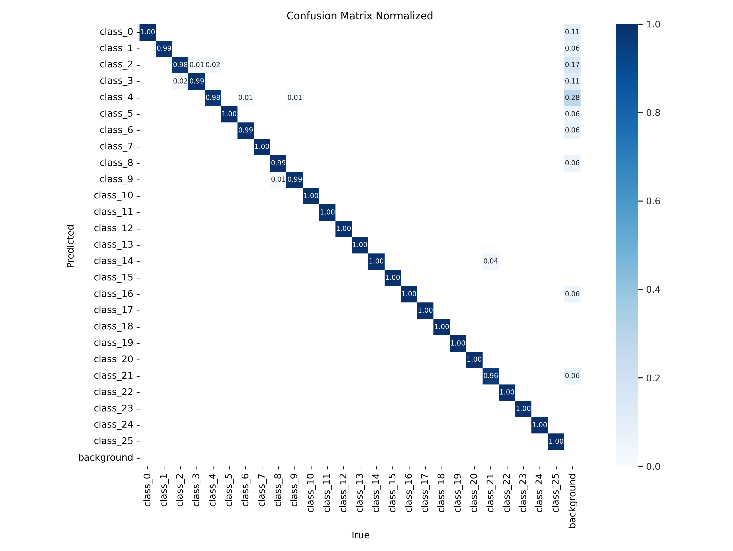


* **Experiment Details:**
  1. The experiment has two main objectives. First detecting characters in the image and classify them. These two main objectives are done simultaneously. Our label for each plate character consists of 5 items. 4 coordination and 1 class label.
  2. The parameters of Network would be investigated in training like depth\_multiple and width\_multiple.
  3. After training and evaluation of the model, I found that database labels have a problem and I put time to solve it and a unused label has been deleted. The modified database has 26 classes in OCR.

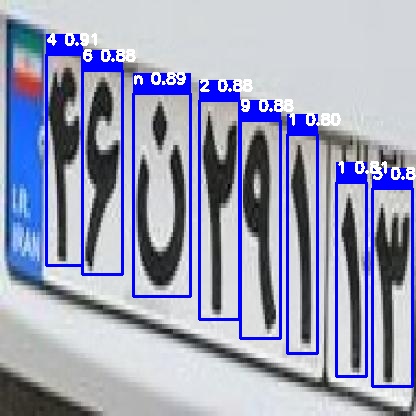
The confusion matrix by best model before modification of dataset is illustrated in following image.



After modifying the dataset and resolving incorrect labels, we see a big improvement in predictions of best model.



* **Error Analysis:**
  1. The performance of my model on the data is illustrated in following image.



* 1. To investigate where our model has misclassification, we do a review on the test data and wrong predicted labels would be plotted. There are three main groups among faults. These groups are
     + 1. wrong classified characters
       2. bonding box fault
       3. label revise

Each of these classes would be investigated in the following sections.

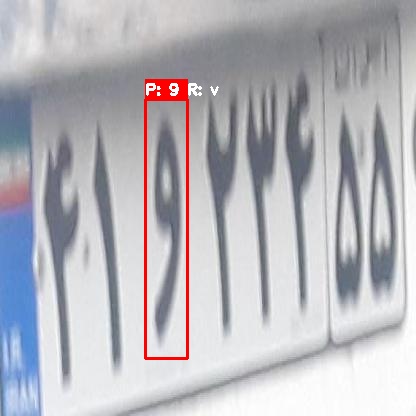
* wrong classified characters



A close up of a license plate

AI-generated content may be incorrect.

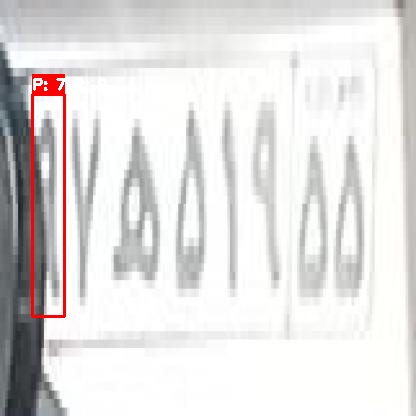




Some of these misclassified characters are result of low resolution, orientation, similarity between and other parameters. This problem can be solved by using more similar data which helps the model to generalize in different working conditions and perform a better OCR with higher reliability.

* bonding box fault



* label revise



#### Additional Aspects for Focus:

1. **Comparative Analysis:**

**To compare our algorithm with other algorithms we search for results of approaches online and we discuss each of them in following.**

* + The persian plate recognition system which is presented in the following link has accuracy about 75 percent in character recognition. It also performs a plate control for accessing to building or town

https://github.com/truthofmatthew/persian-license-plate-recognition?tab=readme-ov-file.

* + In [2], the performance of LPR on proposed dataset is 0.954 in precision metric and 0.956 in term of recall. While our scheme performance is more than 98 percent in both metrics.
  + In [3], the proposed model can achieve accuracy of 90.33% on their dataset.
  + In [7], the performance of a deep learning based approach on the validation data is 87.81% in precision metric.
  + In [8], a LPR has been proposed using multi-class Adaboost. The overall accuracy of the proposed method is examined 90.45%.

1. **Implementation Challenges:**
   * One of challenges in this problem is the effect of validation data on accuracy calculation. There could be several approaches while each of them is reporting their accuracy, precision and recall based on their validation data.
   * Proposing an LPR algorithm which can be used in real environment and deal with data coming from real-time streams could be tricky.
   * Environmental effect on the data is rally high. Different sun lights, shadows, streetlights at night, resolution of pictures, occlusion and etc.
2. **Scalability and Real-World Application:**
   * One of facts about this problem is that we may never get to 100 percent accuracy due to different environmental conditions such as rain. To reach acceptable accuracy, we need to keep gathering data from the real world, especially where our model has weaknesses. In this case, we can test our model in different ways and gather useful data to append our training dataset.
   * These approaches could be widely used in Middle-East countries which have the same plates numbers.
3. **Future Work:**
   * Adding plates from various condition is a time-consuming process since labeling and evaluation of them is tricky.

# FINAL REPORT - Conclusions

* **There are three different losses in training procedures to increase object detection experience for user. Box loss is responsible for detecting objects in images. Class loss determines object classified in bonding box match with true label. DFL loss tries to fine tune bonding box places to be places on the edges of object.**

|  |  |  |  |
| --- | --- | --- | --- |
| Loss type | Role? | Affects... | Visual symptom |
| Box Loss | Measures the difference between **predicted box** and **ground truth box**. | Box position, size, shape | Box misses the object, or too big/small |
| Class Loss | Measures how well YOLO **predicts the object class** inside a box. | Which label is predicted | Box says "dog" but it's actually a "cat" |
| DFL Loss | Makes the box boundary prediction **more fine-grained** | How tightly box hugs the object | Box is slightly "loose" or "offset" |

* The **total loss** is:

Total Loss = Box Loss + Class Loss + DFL Loss

* **Starting with charts to see what is happening in the training process.**

**A graph of a graph

AI-generated content may be incorrect.**

**There seems to be a little overfitting in bonding box classification, which may be due to limited number of training samples. I have use**

**A graph of a graph

AI-generated content may be incorrect.**

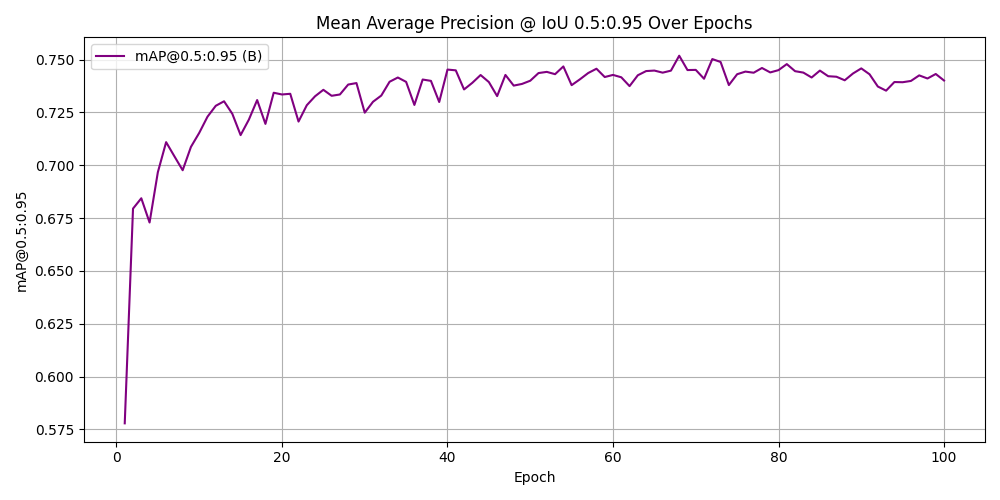
**A graph of a graph

AI-generated content may be incorrect.**

* **Precision, Recall and MAP during training**

**A graph with lines and numbers

AI-generated content may be incorrect.**

****

* **Comparing our method with other schemes:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| approach | dataset | Year | Precision | Recall | mAP50 | mAP50-95 |
| My Approach | [PPD](https://universe.roboflow.com/shahab-jafari-1vorv/persian-plate-characters-mvinj) | 2025 | 0.98286 | 0.99634 | 0.99464 | 0.75185 |
| YOLOv5 [2] | IR-LPR | 2022 | 0.954 | 0.956 | 0.982 | 0.817 |
| LPRNet [3] | Own Collected data | 2019 | 0.9033 | - | - | 0.732 |
| CTC [7] | Own Collected data | 2024 | 0.8722 | - | - | - |
| Multiclass Adaboost [8] | - | 2012 | 0.9045 | - | - | - |

* **Cross Validation training**

**In order to perform cross-validation, we first split our dataset to 5 folds and perform training on each fold.**

A screen shot of a computer program

AI-generated content may be incorrect.

**The training would be done using following code on the modified dataset and results would be saved to further usage.**

****

**In following images, the results of cross validation in the dataset has been illustrated. As it is obvious, the results of cross validation on each chunk is higher than 90 percent for Precision, Recall, and mAP50 index. For mAP50-95 this value is between 73 and 74 percent for all chunks.**

**A white sheet of paper with black text

AI-generated content may be incorrect.**

**A close-up of a graph

AI-generated content may be incorrect.**

**A white sheet of paper with black text

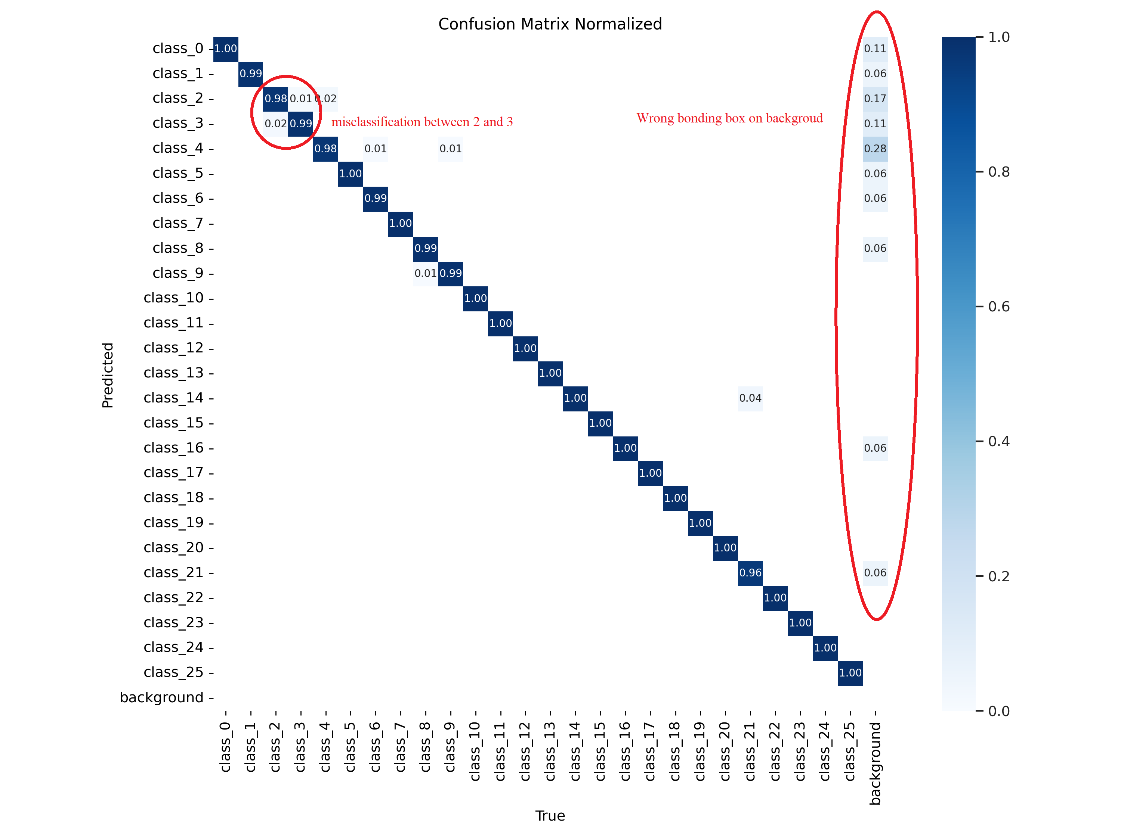
AI-generated content may be incorrect.**

**A close-up of a graph

AI-generated content may be incorrect.**

* **Insights from Experiments:**
  + Confusion matrix interpretation

The algorithm performs a good accuracy in classification of each character. The most problem right now is misclassification between 2 and 3 which is so common since these two characters are so similar.



Background wrong bonding box is a problem which can be easily addressed by adding new training data. In this case the network will learn more about background objects and errors like this will be reduced.

* + Best confidence threshold to reach optimum F1-score

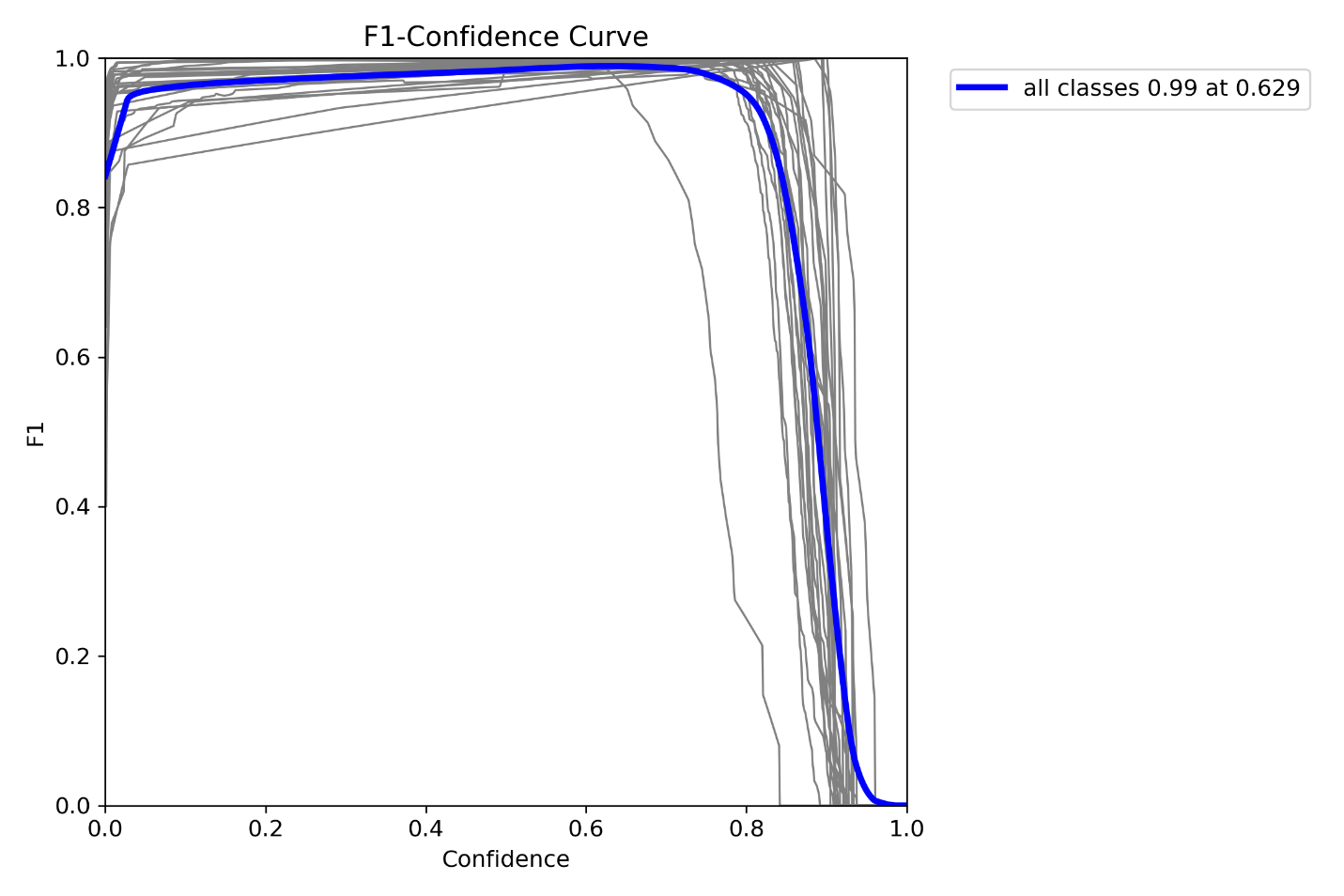
To have the highest performance on data, we need to pick up a threshold which directly impacts Precision and recall. To keep precision and recall trade-off at the highest value, F1 score would be defined and the effect of changing threshold on this parameter has been plotted.

**Precision** = TP / (TP + FP)

**Recall** = TP / (TP + FN)

**F1 Score** = 2 \* (Precision \* Recall) / (Precision + Recall)

As it is obvious when the threshold is selected 0.629 the F1-score is the highest and in maximum value of itself. In this case we have minimum number of FP and FN.



* **Conclusion:**

1. **Impact of the Work:**
   * Although it may be easy to develop a OCR algorithm using pre-trained model and available datasets, all datasets are far from real-world applications.
   * In this project we tried to work with a dataset which is close to real-world applications to increase the generalization of our model to be able to be used in different conditions.
2. **Lessons Learned:**
   * Data collection plays a significant role in deep learning approaches since any problems in data collection could be cascaded in wrong classification and detection in plate recognition task
   * Although we were dealing with data which backgrounds have similar objects, some backgrounds are predicted as characters which makes the problem harder to solve and requires more training data to prevent similar problems.
3. **Limitations:**
   * Data collection has so many limitations since it should be collected in a standard way which is not possible because different environmental effects on output images
   * Data from different cameras, direction, sun light, and weather can directly affect the performance of models.
   * To ensure that model has its best performance, data collection for test should be done by same standard
4. **Future Directions:**
   * Collecting a comprehensive dataset from different parts of the country can increase the challenges of this problem due to different conditions of them.
   * Work more on threshold detection since it seems to play a significant role in optimizing Precision and Recall Metrics
5. **Closing Statement:**
   * In this project we tried to propose a license plate recognition algorithm for plates with Persian and Arabic numbers. In this method, we performed detection and classification at the same time on an image and we returned labels for each image as output. The proposed method has good accuracy on test and validation dataset. It is obvious that the performance of our method would be directly affected by testing it over a wider dataset and in real-world application and it requires fine-tuning the model to perform better in those conditions.
   * This project gave me a good insight into how to use the prepared tools or pre-trained model developed by others and try to modify it in a way to get my desired results. This can provide you with an infinite resource which you can use in the right place to reach your goals.

# References

[1] A. Samadzadeh, A. M. Shayan, B. Rouhani, A. Nickabadi, and M. Rahmati, "RILP: Robust Iranian License Plate Recognition Designed for Complex Conditions," in *2020 International Conference on Machine Vision and Image Processing (MVIP)*, 18-20 Feb. 2020 2020, pp. 1-7, doi: 10.1109/MVIP49855.2020.9116910.

[2] M. Rahmani, M. Sabaghian, S. M. Moghadami, M. M. Talaie, M. Naghibi, and M. A. Keyvanrad, "IR-LPR: Large Scale of Iranian License Plate Recognition Dataset," *arXiv [cs.CV],* 2022 2022. [Online]. Available: <http://arxiv.org/abs/2209.04680>.

[3] Y. Alborzi, T. S. Mehraban, J. Khoramdel, and A. N. Ardekany, "Robust Real time Lightweight Automatic License plate Recognition System for Iranian License Plates," in *2019 7th International Conference on Robotics and Mechatronics (ICRoM)*, 20-21 Nov. 2019 2019, pp. 352-356, doi: 10.1109/ICRoM48714.2019.9071863.

[4] A. Tourani, S. Soroori, A. Shahbahrami, and A. Akoushideh, "Iranis: A Large-scale Dataset of Iranian Vehicles License Plate Characters," in *2021 5th International Conference on Pattern Recognition and Image Analysis (IPRIA)*, 28-29 April 2021 2021, pp. 1-5, doi: 10.1109/IPRIA53572.2021.9483461.

[5] A. R. Rashtehroudi, A. Shahbahrami, and A. Akoushideh, "Iranian License Plate Recognition using Deep Learning," in *2020 International Conference on Machine Vision and Image Processing (MVIP)*, 18-20 Feb. 2020 2020, pp. 1-5, doi: 10.1109/MVIP49855.2020.9116897.

[6] S. M. Baharlou, S. Hemayat, A. Saberkari, and S. Yaghoobi, "Fast and adaptive license plate recognition algorithm for Persian plates," in *2015 2nd International Conference on Pattern Recognition and Image Analysis (IPRIA)*, 11-12 March 2015 2015, pp. 1-6, doi: 10.1109/PRIA.2015.7161638.

[7] S. Hatami, F. S. Jamali, and M. Sadedel, "Iranian license plate recognition using a reliable deep learning approach," *Scientia Iranica,* vol. 31, no. 14, pp. 1105-1121, 2024, doi: 10.24200/sci.2024.61312.7245.

[8] M. M. Dehshibi and R. Allahverdi, "Persian Vehicle License Plate Recognition Using Multiclass Adaboost," *International Journal of Computer and Electrical Engineering,* vol. 4, pp. 355-358, 05/16 2012, doi: 10.7763/IJCEE.2012.V4.511.