

# Fastest Moroccan license plate recognition using a lightweight modified YOLOv5 model

Abdelhak Fadili<sup>1</sup>, Mohammed El Aroussi<sup>2</sup>, Youssef Fakhri<sup>1</sup>

<sup>1</sup>Laboratory of Research in Informatics, Department of Informatics, Faculty of Science, University Ibn Tofail, Kenitra, Morocco

<sup>2</sup>Laboratory of Intelligent Systems and Sensor Networks, Hassania School of Public Works, Casablanca, Morocco

## Article Info

### Article history:

Received Jan 7, 2024

Revised Jul 6, 2024

Accepted Jul 26, 2024

### Keywords:

Deep learning

Intelligent transportation system

License plate detection

Optical character recognition

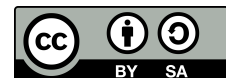
ShuffleNet

YOLOv5

## ABSTRACT

Morocco is witnessing an alarming surge in road accidents. Automatic license plate recognition (ALPR) technology is vital in enhancing road safety. It enables applications like traffic management, law enforcement, and toll collection by automatically identifying vehicles on the roads. This paper integrated the ShuffleNet V2 architecture into the end-to-end YOLOv5 object detection system. The goal was to develop a model capable of accurately detecting Moroccan license plates with an 87% accuracy rate. The proposed model was able to achieve high processing speeds of 60 frames per second (FPS) while maintaining a compact size of 1.3 megabytes and a limited computational requirement of 0.44 million floating-point operations. Compared to other models used in similar contexts, this model demonstrates superior performance and high compatibility with embedded systems, making it a promising solution for addressing road safety challenges in Morocco.

*This is an open access article under the [CC BY-SA](#) license.*



## Corresponding Author:

Abdelhak Fadili

Laboratory of Research in Informatics, Department of Informatics, Faculty of Science, University Ibn Tofail

BP 242, Kenitra, Morocco

Email: fadili101@gmail.com

## 1. INTRODUCTION

As reported by the World Health Organization (WHO), Morocco has a significant number of road traffic-related fatalities. Approximately 3,500 deaths per year are attributed to road traffic accidents, according to estimates in [1]. Additionally, a study conducted by the Moroccan Ministry of Transport in 2019 revealed that road accidents in Morocco injured around 30,000 people annually. This is explained by the significant increase in the number of vehicles.

It can be observed that this rise in numbers presents various challenges in effectively monitoring instances of car abuse or theft, administering parking fees, enforcing highway speed limits, implementing red light enforcement measures, and collecting highway tolls, among other related matters. These concerns are of utmost importance in ensuring traffic safety and urban security, preventing traffic congestion, and facilitating automated traffic management. The most effective approach to accomplish this task is through the implementation of an automated license plate recognition system (ALPR). ALPR systems employ cameras and software to detect and recognize license plate numbers from vehicles automatically. Complex environments such as lighting conditions, weather conditions, obstructions, angle of view, and plate variation can present a challenge for ALPR systems, making it difficult for them to detect and recognize license plate numbers accurately.

To address these challenges, ALPR systems frequently employ sophisticated image processing techniques to enhance the quality of captured images. Additionally, machine learning algorithms are utilized

to enhance the precision of license plate recognition. Convolutional neural networks (CNNs) are deep learning algorithm that enhances image object detection.

Previous studies [2], [3] used for license plate detection propose methods that typically capture some morphological, color, or texture features. These techniques are either computationally demanding, making them unsuitable for real-time systems, or highly susceptible to changes in plate color. Hough transformation approaches [4] assume that license plates are delineated by lines that surround them. With the advent of deep learning, researchers turned to the CNN-based object detection model [5] and its derivatives such as regions with convolutional neural network features (R-CNN) [6], single shot multibox detector (SSD) [7], and you only look once (YOLO) [8] which have been widely used for license plate recognition (LPR) systems.

Laroca *et al.* [9] utilized the YOLO object detector in two stages, employing simple data augmentation strategies on inverted characters and license plates, with fine-tuning and training at each stage, achieving a recognition accuracy of 93.53% at 47 frames per second (FPS), outperforming commercial systems from OpenALPR and Sighthound. Moreover, according to Tung *et al.* [10], multitask learning was employed for LPR, utilizing single-stage detection CNNs like RetinaFace and MobileNet, achieving high precision (97.70%) and a speed surpassing 62 FPS. Furthermore, according to Selmi *et al.* [11], an LPR method was proposed, employing Gaussian filtering, morphological operators, edge detection, and geometry analysis in the pre-processing stage to enhance license plate features. However, the method has limitations, such as handling only one license plate per image and sensitivity to distortions and illumination. Lastly, according to Kundrotas *et al.* [12], the internal structure of the Hourglass block has been improved using ResNet blocks to enhance the speed of license plate identification, particularly in complex conditions, providing a fast and efficient solution, achieving an average accuracy of 96.19% and a speed of up to 405 FPS.

According to the information provided in [10]–[12], it is observed that to achieve precise LPR, the suggested approaches consist of two primary components:

- License plate detection: in this phase, the objective of the researchers is the detection of the license plate, which, when captured, may be under unfavorable conditions or far away (a small object). Computer vision has recently recorded great success in various fields, such as the detection and classification of objects. Therefore, computer vision currently offers the possibility to counteract these problems. For example, in the literature, proposed methods that are based on CNN can achieve reasonable accuracy under different conditions.
- Character recognition: once the plate is detected, the researchers proceed to the phase of recognizing its components. In this part, the proposed methods must be able to detect several objects simultaneously. Currently, various object detection algorithms have excellent performance in real-time.

However, the majority of images and video sequences are captured in real-world environments where objects are subject to external factors or are of limited scale. The available methods for addressing these challenges are limited in number. The methods derived from the YOLO algorithm continue to exhibit superior performance inside this particular setting. This article presents a novel and robust method for real-time detection and recognition of Moroccan license plates. The primary contributions of this work can be briefly summarized as follows:

- We provide a lightweight and enhanced Moroccan license plate detector based on YOLOv5, which can achieve an excellent balance between accuracy and speed. Firstly, the new ShuffleCANet, which combines the ShuffleNetV2 network and the attention coordination mechanism, is proposed as backbone. However, adjustments in the algorithm would be required to work on different models of Moroccan number plates.
- We have created a collection of over 7,000 high quality digital photos showing license plates on cars from Morocco.
- According to our tests on the VisDrone DET 2021 dataset [13], application-oriented license plate (AOLP) dataset [14], and our Moroccan license plate dataset, the approach we propose performs more effectively than previous tries on embedded systems.

The structure of this paper is as follows: in the section 2 we describe the ALPR method we suggest. Section 3 displays and analyzes the outcomes of our tests. The concluding section presents a conclusion of our study with projections of future perspectives in section 4.

## 2. METHOD

In this section, we dissect our ALPR network architecture. We used an improved version of YOLO. As a family of object detectors, YOLO has proven to be the best in this field. Figure 1 illustrates our end-to-end ALPR network architecture.

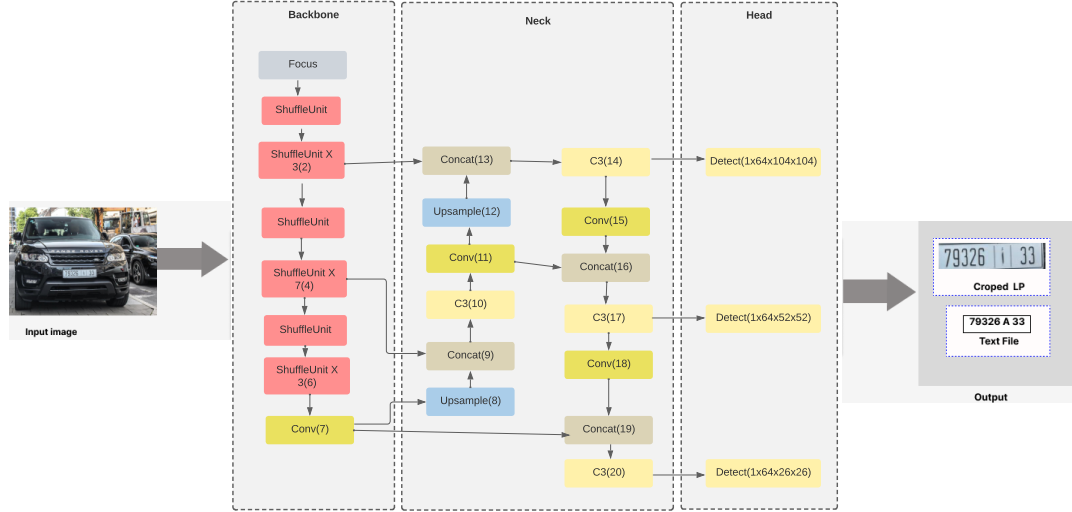


Figure 1. The diagram flowchart of our system

### 2.1. YOLOv5

The YOLOv1 architecture, as outlined in [8], employs a single neural network to directly predict bounding boxes and class probabilities from entire images in one evaluation, facilitating the development of real-time models. This is achieved by generating an  $S \times S$  grid from the input image and determining whether the target object is situated at the center of any grid cell. Each grid cell then predicts  $B$  bounding boxes along with their corresponding confidence scores, reflecting the model's certainty regarding an object's presence and its predictions' accuracy. Formally, confidence is defined as (1):

$$Pr(Object) * IOU \quad (1)$$

The bounding box consists of  $(x, y)$ , which defines the center point relative to the grid cell boundaries. The width ( $W$ ) and height ( $H$ ) are predicted relative to the full picture size, and a confidence score that intersection over union (IoU) between the predicted box and any ground truth box.

To improve YOLOv1, YOLOv2 [15], [16] uses Darknet-19 as a backbone, batch normalization to enhance neural network stability, a high-resolution classifier increased from  $224 \times 224$  to  $448 \times 448$  for more accuracy, and employ anchor boxes which are accountable for the prediction of the bounding container and which are designed for a given dataset using clustering (k-means clustering). YOLOv3 [17], [18], also called Darknet-53, is a model that has been developed based on ResNet [19] and feature-pyramid network (FPN). It uses skip connections like ResNet and three prediction heads like FPN topology, which allows YOLOv3 to detect objects of different sizes.

YoloV4 [20] is an important improvement of YoloV3, whose authors have developed a new architecture in the backbone, and they have made modifications in the Neck in order to improve the mean average precision ( $mAP$ ). The YOLOv4 backbone is a deep neural network composed mainly of convolution layers. The key objective of the backbone is to extract relevant features. Backbone selection is a crucial step in improving object detection performance. The system consists of three components: a bag of gifts, which serves to enhance the training cost or modify the training method while maintaining a low inference cost; a bag of specials, which slightly increases the inference cost but can greatly enhance the accuracy of object detection; and cross stage partial (CSP) Darknet53, which employs a CSPNet approach to divide the backbone feature map into two segments and subsequently joins them through a multi-step hierarchy. The implementation of a split and merge approach facilitates a more gradual and attenuated propagation of information within the network.



trade-off, including Xception [25], MobileNet [26], MobileNet V2 [27], ShuffleNet [28], and CondenseNet [29]. To measure computational complexity, there are three widely used metrics: memory access cost (MAC), the number of floating point operations or FLOPs, and the degree of parallelism. Ma *et al.* [30] showed that Shufflenetv2 is the best in comparison with the others.

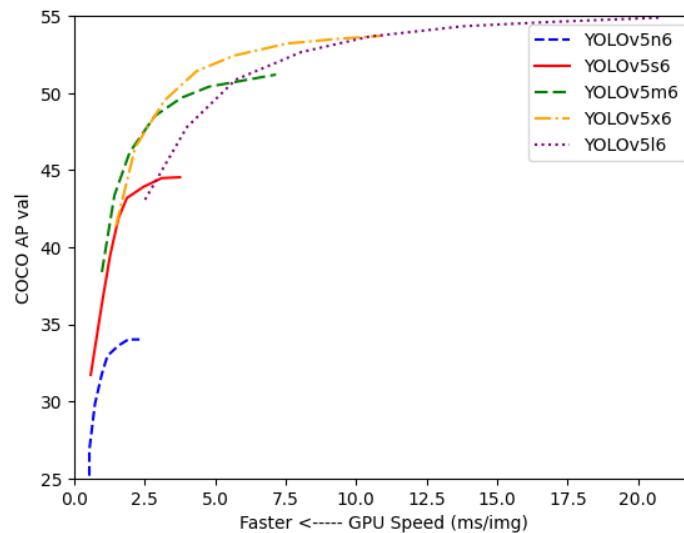


Figure 4. Comparison of performance between the different versions of YOLOv5

### 2.2.1. ShuffleNet V2

ShuffleNet V2 block is an image model block used in the ShuffleNet V2 architecture, where speed is the metric optimized for. It utilizes a simple operator called channel split, which divides the characteristic channel input into two branches with channels  $c-c'$  and  $c'$  channels individually at the start of each unit. Following G3, one branch remains as identity. The other branch consists of three convolutions with the same input and output channels to satisfy G1. The two  $1 \times 1$  convolutions are no longer group-wise, unlike the original ShuffleNet. This is partially to follow G2 and partially due to the fact that the split operation already produces two other groups. After convolution, the two branches are merged. So, the number of channels remains the same (G1). The same "channel shuffle" operation as in ShuffleNet is then used to enable information communication between the two branches. The ShuffleNet V2 basic unit appears in Figure 5.

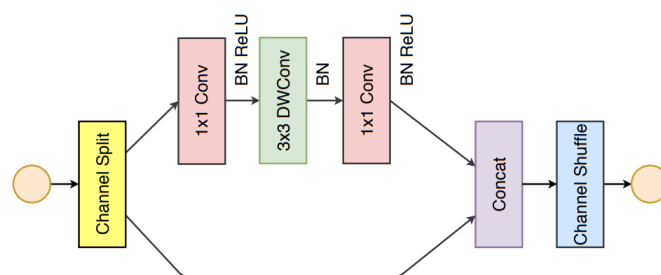


Figure 5. ShuffleNet V2 basic unit

The inspiration for channel split is that elective structures, where pointwise group convolutions and bottleneck structures are utilized, prompt expanded memory access expenses. Additionally, more network fragmentation with group convolutions diminishes parallelism (less congenial for GPU), and the element-wise addition activity, while they have low FLOPs, has a high memory access cost. Channel split is an elective where we can support a huge number of similarly wide channels (equally wide minimizes memory access cost) without having dense convolutions or too many groups. According to the study carried out, it seems that the

integration of shufflenetv2 in the backbone part of YOLOv5s will give a precise and especially fast model applicable to the embedded computing boards.

### 2.2.2. Focus layer

The focus module in YOLOv5 partitions the image prior to its input into the backbone. The particular procedure involves obtaining a value for alternate picture pixels, similar to nearby subsampling. By employing this method, a total of four photos are acquired, each of which serves as a complement to the others. Furthermore, it is noteworthy that these images possess nearly identical lengths, hence ensuring the absence of any loss of information. This approach consolidates the W and H data within the channel space, resulting in a fourfold expansion of the input channel. In this context, the pasted images are being compared to the original three-channel RGB mode. The image is initially processed using 12 channels, followed by a convolution operation. Finally, a double subsampling feature map is generated, ensuring that no information is lost in the process. Figure 6 illustrates the slice operation performed with yolov5s on a  $640 * 640 * 3$  image that becomes after the slice operation a  $320 * 320 * 12$  feature map, then after a convolution operation, it finally becomes a  $320 * 320 * 32$  feature map.

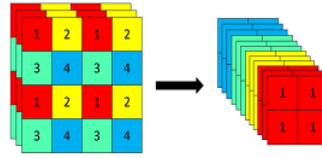


Figure 6. Focus structure slice operation

## 3. EXPERIMENTS AND RESULTS

### 3.1. Evaluation metrics

Object detection is both a regression and a classification issue. The  $mAP$  metric is utilized as a part of the examination. Firstly, the inclusion of the predicted detection box  $P$  to the ground-truth bounding box  $G$  is calculated as (2):

$$IoU(P, G) = \frac{P \cap G}{P \cup G} \quad (2)$$

where  $IoU$  represents the degree of overlap between predicted bounding box and ground-truth bounding box.

The threshold of the experiment datasets is 0.5; if the  $IoU$  is more prominent than 0.5, the detection is deemed a success. Afterward, the recall (3) and precision (4) of each class are figured independently. Where  $TP$  is the number of correctly predicted samples,  $FP$  is the number of incorrectly predicted samples, and  $N_c$  is the actual number of samples in this class.

$$Recall = \frac{TP}{N_c} \quad (3)$$

$$precision = \frac{TP}{TP + FP} = \frac{Recall \cdot N_c}{Recall \cdot N_c + FP} \quad (4)$$

The average precision for each class is then computed independently as pursues. Taking 11 positions on the interval  $[0, 1]$  of the recall curve at intervals of 0.1, the precision for that class is expressed as a piecewise function of the recall rate. The area under the function curve is computed as the average precision of the class. Finally, the  $mAP$  of the whole test set is acquired by averaging the mean accuracy of all classes. The detection speed is used to evaluate the promptness of object detection in application scenarios, and the FPS metric is used to assess the detection speed, which is the number of images that can be processed per second.

### 3.2. Performance evaluation on VisDrone dataset

In the context of YOLOv5, it is preferable to initiate the test with a comparative evaluation between its different versions. We first compared our proposed method with YOLOv5n, YOLOv5s, and YOLOv5m on the VisDrone2021-DET dataset to test their performance in terms of precision, recall and  $mAP$ . The YOLOv5n,

YOLOv5s, YOLOv5m, and our model were completed training across 100 epochs using the Visdrone dataset, which included training images, test images, and validation images. The relative outcomes of the object detection models are displayed in Table 1. In addition, this table shows the precision, recall,  $mAP_{0.5}$ , and  $mAP$  of all models.

Table 1. Comparison results between the suggested architecture and other methods on VisDrone-DET dataset

| Method  | $P(\%)$ | $R(\%)$ | $mAP_{0.5}(\%)$ | $mAP(\%)$ |
|---------|---------|---------|-----------------|-----------|
| YOLOv5n | 36.3    | 26.0    | 24.1            | 12.2      |
| YOLOv5s | 44.2    | 31.1    | 30.9            | 16.7      |
| YOLOv5m | 48.4    | 34.6    | 35.3            | 20.1      |
| Our     | 33.4    | 26.8    | 26.1            | 13.5      |

We compared based on the best result values of the 100 epochs. The  $mAP$  value of the original YOLOv5n, YOLOv5s, and YOLOv5m models are 12.2%, 16.7%, and 20.1%, respectively, while ours is 13.5%. Overall, We can see that our method has still seen an improvement (1.3%) in  $mAP$  compared to YOLOv5n, which is the best model for mobile and embedded systems. But, it is less accurate compared to YOLOv5s and YOLOv5m. This is justified by the latter methods requiring more resources Table 2.

Table 2. Performance comparison of memory size and the number of parameters on VisDrone

| Method  | Params (Million) | Memory (MB) |
|---------|------------------|-------------|
| YOLOv5n | 1.77             | 3.80        |
| YOLOv5s | 7.03             | 14.4        |
| YOLOv5m | 20.88            | 42.2        |
| Our     | 0.44             | 1.30        |

The deployment of computationally intensive and memory-demanding CNN models on resource-constrained mobile and embedded devices poses significant challenges. A primary limitation arises from the restricted memory capacities inherent to these platforms. Consequently, minimizing the memory footprint of CNNs becomes imperative for enabling efficient inference on mobile and embedded systems. The memory requirements of a CNN architecture are contingent upon the memory allocation necessary for storing its learnable parameters, encompassing weights and biases, as well as the intermediate activation data exchanged between the computational operators constituting the network.

In this context, we compared several parameters and memory required between YOLOv5s, YOLOv5m, and our model. As exhibited in Table 2, the total parameters of our modified YOLOv5 are 6.60 M less than the YOLOv5s model and 20.4 M less than the YOLOv5m model. The memory requirement is also reduced for our model to 1.3 MB while it is 14.4 MB and 42.2 MB for YOLOv5s and YOLOv5m, respectively. This comparison shows that our model is less demanding regarding memory and computation than the original models.

### 3.3. Performance evaluation on application-oriented license plate dataset

We moreover compared our approach with others on the AOLP database, specifically designed to evaluate performance in LPR. This database includes images of  $352 \times 240$  pixels, totaling 2049 records, with 1268 utilized for testing. It is subdivided into three subsets identified by categories (automatic cars (AC), license enforcement (LE), and regional plates (RP)). Table 3 shows the comparison between the methods of previous researchers and ours.

The evaluation of the performance of different LPR methods on the AOLP database reveals distinct results, as shown in Table 3. The method [9] stands out with the highest  $mAP$  of 99.85%, although it is accompanied by a relatively longer processing time, requiring 8.54 ms for detection and 3.09 ms for recognition. Similarly, method [10], displaying a slightly lower  $mAP$  of 95.92%, is also slower with a processing time of 16.13 ms. Method [11] shows similar performance with an  $mAP$  of 95.50%, but details on its processing time are not specified. In contrast, method [12] manages to balance precision and speed, with an  $mAP$  of 96.19% and processing times of 2.2468 ms for detection and 0.4686 ms for recognition. Finally, our method achieves an  $mAP$  of 97.6%, placing it between the results of [9] and [12], and standing out with its particularly short processing time of only 0.384 ms per image. This demonstrates that our approach effectively balances precision and speed, thus providing an efficient solution for LPR in various scenarios and in real-time.

Table 3. Comparison results between the suggested architecture and other methods on AOLP datasets

| Method | <i>mAP</i> (%) | Speed (ms)      |
|--------|----------------|-----------------|
| [9]    | 99.85          | 8.54 + 3.09     |
| [10]   | 95.92          | 16.13           |
| [11]   | 95.50          | - + -           |
| [12]   | 96.19          | 2.2468 + 0.4686 |
| Our    | 97.6           | 0,384           |

### 3.4. Performance evaluation on our dataset

As far as we know, there are no publicly accessible databases containing images of Moroccan license plates. That's why we decided to create our own. Most of the existing systems do not handle plates with Arabic letters. In addition, the Moroccan plates have a special shape. It is split into three parts: the first part (on the right) includes the code associated with the plate's emission region identifier.

This identification goes from 1 to 89. Then, a letter of the Arabic dialect is incremented in the middle of the control plate. This last one takes into account the registration number of the car. The last part is composed of a series of five digits ranging from 1 to 99.999, corresponding to the car's registration number. Figure 7 shows an example of a Moroccan license plate.



Figure 7. Moroccan license plate structure

The initial step in developing a dataset involves acquiring visual data in the form of photos. The acquisition of a dataset constitutes a crucial component in developing a precise object detection system. This procedure's initial step involves acquiring photos and video clips, followed by their subsequent classification and annotation. We used a Garmin Dashcam 56 camera, a Huawei Y9 phone, and a Samsung Galaxy M53 5G phone to capture 7312 images. These choices will offer us a diversified resolution, brightness, and starlight mention dataset. Figure 8 shows some Moroccan license plate images. After selecting the suitable photographs, we will proceed to annotate them using a labeling tool. This tool, known as a graphical image annotation tool, enables us to delineate visual boxes around the objects present in each image. Additionally, the software can store the XML files corresponding to the annotated photographs automatically.



Figure 8. Sample of database images

In the comparison section using the Visdrone dataset, we highlight the efficiency of our model in terms of reduced parameter count, compact size, and fast processing speed. Subsequently, we assess the performance of our model on our dataset. Table 4 presents the *mAP* and detection speed results for various methods, including YOLOv5n, YOLOv5s, YOLOv5m, YOLOv5l, YOLOv5x, and our approach. While YOLOv5x and YOLOv5l achieve the highest *mAP* (90.9% and 90.54%, respectively), they exhibit slower processing. In contrast, YOLOv5m achieves an *mAP* of 88.96% at 54 FPS, and YOLOv5s balances precision and speed with an *mAP* of 85.70% at 68 FPS. Our method achieves a balance with an *mAP* of 86.9% at 60 FPS, indicating a commendable compromise. Figure 9 visually displays the performance of our model.



Table 4.  $mAP$  and detection speed results on our dataset

| Method  | $mAP$ (%) | FPS (ms) |
|---------|-----------|----------|
| YOLOv5n | 77.21     | 72       |
| YOLOv5s | 85.70     | 68       |
| YOLOv5m | 88.96     | 54       |
| YOLOv5l | 90.54     | 37       |
| YOLOv5x | 90.9      | 24       |
| Our     | 86.9      | 60       |

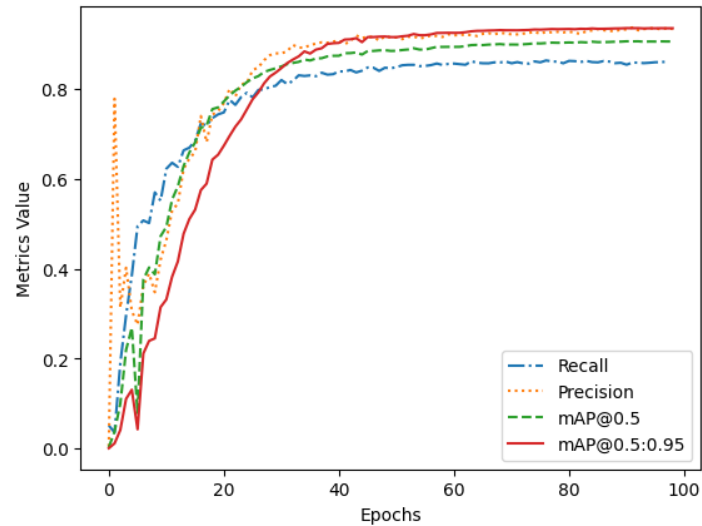


Figure 9. Performance metrics curve of our model on our dataset

To validate the capability and viability of our proposed technique, we have collected a set of videos of real road scenes, and we have selected different images. The outcomes of the detection are appeared in Figure 10. We can see that the proposed network can detect small objects such as far license plates. This demonstrates that our proposed improvement works well even for complex real-time road environments.



Figure 10. ALPRS results

Our approach outperforms YOLOv5n on the VisDrone dataset with a 13.5%  $mAP$ , showcasing efficiency for mobile systems. On the AOLP database, our method achieves 97.6% accuracy at a remarkable speed of 0.384 ms per image, demonstrating robustness in diverse scenarios. We created a Moroccan license plate dataset, validating our model’s versatility in real-world conditions. Efficient parameter usage and reduced memory footprint make our model suitable for resource-constrained devices, providing a balance between accuracy and efficiency.

#### 4. CONCLUSION

In this work, our study presents a new lightweight and powerful ALPR system based on YOLOv5 and incorporating ShuffleNetV2 for a good balance between accuracy and speed. Our results on our dataset show 86.9% accuracy, with fast 60 FPS detection. The benefits of our method include fewer parameters and a small 1.3 MB model size, suitable for limited resource embedded systems. This design is effective for real-time license plate detection, with implications in traffic control and law enforcement. In the future, we suggest conducting detailed studies to further enhance optimizations, focusing on tailoring the system to specific situations and broadening the dataset for robustness in various environments. Analyzing energy consumption is crucial for optimizing energy efficiency. It is also important to continually improve real-time processing and memory efficiency to enhance the practical use of our ALPR system.

#### ACKNOWLEDGEMENTS

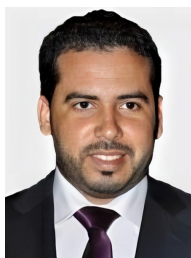
We feel obliged to extend our sincere appreciation to Ibn Tofail University in the Kingdom of Morocco for their dedicated support and steady motivation that have served as guiding indications throughout this work.




#### REFERENCES

- [1] A. Zerka and F. Jawab, "Contribution to the economic analysis of numerical data of road accidents in Morocco," in *Digital Technologies and Applications*, Springer, 2022, pp. 133–144, doi: 10.1007/978-3-031-02447-4\_14.
- [2] C. N. E. Anagnostopoulos, I. E. Anagnostopoulos, I. D. Psoroulas, V. Loumos, and E. Kayafas, "License plate recognition from still images and video sequences: A survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 9, no. 3, pp. 377–391, 2008, doi: 10.1109/TITS.2008.922938.
- [3] S. Du, M. Ibrahim, M. Shehata, and W. Badawy, "Automatic license plate recognition (ALPR): A state-of-the-art review," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 23, pp. 311–325, 2012.
- [4] W. Le and S. Li, "A hybrid license plate extraction method for complex scenes," *18th International Conference on Pattern Recognition (ICPR'06)*, vol. 2, no. 1, pp. 324–327, 2006, doi: 10.1109/ICPR.2006.83.
- [5] Z. Huang, J. Wu, and F. Xie, "Automatic recognition of surface defects for hot-rolled steel strip based on deep attention residual convolutional neural network," *Materials Letters*, vol. 293, 2021, doi: 10.1016/j.matlet.2021.129707.
- [6] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, 2017, doi: 10.1109/TPAMI.2016.2577031.
- [7] W. Liu et al., "SSD: single shot multibox detector," in *Computer Vision – ECCV 2016*, Cham: Springer International Publishing, 2016, pp. 21–37, doi: 10.1007/978-3-319-46448-0.
- [8] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: unified, real-time object detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 779–788, doi: 10.1109/CVPR.2016.91.
- [9] R. Laroca, L. A. Zanlorensi, G. R. Gonçalves, E. Todt, W. R. Schwartz, and D. Menotti, "An efficient and layout-independent automatic license plate recognition system based on the YOLO detector," *IET Intelligent Transport Systems*, vol. 15, no. 4, pp. 483–503, 2021, doi: 10.1049/itr2.12030.
- [10] C.-L. Tung, C.-H. Wang, and B.-S. Peng, "A deep learning model of dual-stage license plate recognition applicable to the data processing industry," *Mathematical Problems in Engineering*, vol. 2021, pp. 1–13, 2021, doi: 10.1155/2021/3723715.
- [11] Z. Selmi, M. Ben Halima, and A. M. Alimi, "Deep learning system for automatic license plate detection and recognition," in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, 2017, pp. 1132–1138, doi: 10.1109/ICDAR.2017.187.
- [12] M. Kundrotas, J. Janutėnaitė-Bogdanienė, and D. Šešok, "Two-step algorithm for license plate identification using deep neural networks," *Applied Sciences*, vol. 13, no. 8, 2023, doi: 10.3390/app13084902.
- [13] P. Zhu et al., "Detection and tracking meet drones challenge," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 11, pp. 7380–7399, 2021, doi: 10.1109/TPAMI.2021.3119563.
- [14] G. S. Hsu, J. C. Chen, and Y. Z. Chung, "Application-oriented license plate recognition," *IEEE Transactions on Vehicular Technology*, vol. 62, pp. 552–561, 2013.
- [15] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 6517–6525, doi: 10.1109/CVPR.2017.690.
- [16] S. M. Abas, A. M. Abdulazeez, and D. Q. Zeebaree, "A YOLO and convolutional neural network for the detection and classification of leukocytes in leukemia," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 25, no. 1, pp. 200–213, 2022, doi: 10.11591/ijeecs.v25.i1.pp200-213.
- [17] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv-Computer Science*, pp. 1–6, 2018, doi: 10.48550/arXiv.1804.02767.
- [18] M. Razzok, A. Badri, I. E. L. Mourabit, Y. Ruichek, and A. Sahel, "Pedestrian detection under weather conditions using conditional generative adversarial network," *IAES International Journal of Artificial Intelligence*, vol. 12, no. 4, pp. 1557–1568, 2023, doi: 10.11591/ijai.v12.i4.pp1557-1568.
- [19] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778, doi: 10.1109/CVPR.2016.90.
- [20] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," *arXiv-Computer Science*, pp. 1–17, 2020.




- [21] G. Jocher *et al.*, “Ultralytics/yolov5: v7.0 - YOLOv5 SOTA realtime instance segmentation,” *Zenodo*, version 7.0, 2022, doi: 10.5281/zenodo.7347926.
- [22] M. E. Ghmary, Y. Ouassine, and A. Ouacha, “License plate character recognition system using YOLOv5,” in *Artificial Intelligence and Smart Environment*, Springer, 2023, pp. 588–594, doi: 10.1007/978-3-031-26254-8\_85.
- [23] C. Y. Wang, H. Y. M. Liao, Y. H. Wu, P. Y. Chen, J. W. Hsieh, and I. H. Yeh, “CSPNet: A new backbone that can enhance learning capability of CNN,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2020, pp. 1571–1580, doi: 10.1109/CVPRW50498.2020.00203.
- [24] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, “Path aggregation network for instance segmentation,” in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8759–8768, doi: 10.1109/CVPR.2018.00913.
- [25] F. Chollet, “Xception: Deep learning with depthwise separable convolutions,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 1251–1258, doi: 10.1109/CVPR.2017.195.
- [26] A. G. Howard *et al.*, “Mobilenets: Efficient convolutional neural networks for mobile vision applications,” *arXiv-Computer Science*, pp. 1–9, 2017, doi: 10.48550/arXiv.1704.04861.
- [27] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, “Mobilenetv2: Inverted residuals and linear bottlenecks,” in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4510–4520, doi: 10.1109/CVPR.2018.00474.
- [28] X. Zhang, X. Zhou, M. Lin, and J. Sun, “Shufflenet: An extremely efficient convolutional neural network for mobile devices,” in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 6848–6856, doi: 10.1109/CVPR.2018.00716.
- [29] G. Huang, S. Liu, L. V. D. Maaten, and K. Q. Weinberger, “CondenseNet: An efficient DenseNet using learned group convolutions,” in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2752–2761, doi: 10.1109/CVPR.2018.00291.
- [30] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, “ShuffleNet V2: Practical guidelines for efficient CNN architecture design,” in *Computer Vision – ECCV 2018*, Cham, Switzerland: Springer, 2018, pp. 122–138, doi: 10.1007/978-3-030-01264-9\_8.

## BIOGRAPHIES OF AUTHORS






**Abdelhak Fadili**    obtained his Bachelor's degree in JAVA & C++ Development from the Faculty of Sciences at Ibn Tofail University, Kenitra, Morocco, in 2009. He also earned his Master's degree in Networks and Systems from the same institution in 2012. Currently, he is pursuing his doctoral studies at the LaRIT Laboratory, Artificial Intelligence team, Faculty of Sciences, Ibn Tofail University Kenitra. His research interests primarily lie in image processing and artificial intelligence. He can be contacted at email: abdelhak.fadili@uit.ac.ma or fadili101@gmail.com.



**Mohammed El Aroussi**    received his Bachelor's degree (Licence ès Sciences) in Science, with a specialization in Computing, Automatics, Electronics, and Electrotechnics. He obtained his Master's degree (DESA) in Computer Science and Telecommunications from the Faculty of Science, University Mohammed V-Agdal, Rabat, Morocco, in 2004, followed by his Ph.D. degree from the same institution in 2009. Currently, he serves as a professor and the head of the Department of Electrical Engineering at EHTP Casablanca. His extensive academic contributions span various domains, including information theory, signal processing, wireless telecommunications, IoT, and AI. He can be contacted at email: moha387@gmail.com.



**Youssef Fakhri**    earned his Ph.D. on November 17, 2007, from Mohammed V University - Agdal, Rabat, Morocco, in collaboration with the Polytechnic University of Catalonia (UPC), Spain. He holds a Bachelor's Degree in Electronic Physics and a Diploma of Advanced Studies (DESA) in Computer and Telecommunications from the Faculty of Sciences, University Mohammed V, Rabat, Morocco, obtained in 2001 and 2003, respectively. Currently, he is a Professor of Higher Education at the Faculty of Sciences in Kénitra. His research focuses on information theory, signal processing, and wireless telecommunications. He can be contacted at email: fakhri@uit.ac.ma.