Fastest Moroccan License Plate Recognition Using a Lightweight Modified YOLOv5 Model

Abdelhak Fadili * , Mohamed El Aroussi , Rachid Saadane , Youssef Fakhri

Posted Date: 29 September 2023

doi: 10.20944/preprints202309.2107.v1

Keywords: License plate detection; deep learning; YOLOv5; ShuffleNet

Article

Fastest Moroccan License Plate Recognition Using a Lightweight Modified YOLOv5 Model

A. Fadili [1,*], M. El Aroussi [2], R. Saadane [2] and Y. Fakhri [1]

[1] LARIT Laboratory, team Network, Telecommunication and intelligence, University Ibn Tofail, Faculty of Science BP 242, Kenitra, Morocco; fadili101@gmail.com
[2] SIRC/LAGES, EHTP BP 8108, Casablanca, Morocco
* Correspondence: abdelhak.fadili@uit.ac.ma

Abstract: The rate of accidents in Morocco is experiencing a significant increase. Automatic license plate detection and recognition (ALPR) is an essential road safety technology. It facilitates applications such as traffic control, law enforcement, and toll collection by allowing for the automated recognition of vehicles on the road. In this study, we incorporated ShuffleNet V2 into the end-to-end YOLOV5 object detection system. The objective was to develop a model capable of identifying Moroccan license plates with an accuracy of 87%. The proposed model is intended to attain a high processing performance of 60 frames per second (FPS) while maintaining a low weight of 1.3 megabytes (MB) and a parameter count of 0.44 million floating point operations (MGFLOP). Our model maintains superior performance and is highly compatible with embedded systems compared to other models utilized in the same context.

Keywords: license plate detection; deep learning; YOLOv5; ShuffleNet

0. Introduction

According to 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 due to 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.

Automatic License Plate Recognition (ALPR) systems are a technology that uses 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.

In order to address these challenges, Automatic License Plate Recognition (ALPR) systems frequently employ sophisticated image processing techniques to enhance the quality of captured images. Additionally, machine learning algorithms are utilized to improve the precision for license plate detection and recognition. Convolutional Neural Networks (CNNs) are deep learning algorithm that enhances image object detection. This article presents a novel and robust method for real-time detection and recognition of Moroccan license plates. The primary contributions of this study can be briefly summarized as follows:

 doi:10.20944/preprints202309.2107.v1

- We propose 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 ShufleNetV2 network and the attention coordination mechanism, is proposed as Backbone. However, modifications in the algorithm would be required to work on different Moroccan license plate layouts.
- we also present a Moroccan car plate dataset containing more than 7,000 full HD images of Moroccan cars.
- Based on the experimental results on the VisDrone DET 2021 dataset and our Moroccan license plate dataset, the proposed approach performs better than previous works on embedded systems.

This paper is organized as follows. We briefly review the relevant work in the second section. The third section introduces the proposed ALPR algorithm. We report and discuss the results of our experiments in section IV. Conclusions and future work are given in Section V.

1. Related Work

Previous studies presented in [2][3] used for license plate detection propose methods that typically capture some morphological, color, or texture features. These techniques are either computationally expensive and thus unsuitable for real-time systems or very easily affected by plate color change. Hough transformation methods [4] assume that license plates are defined by lines that surround them.

Schreiber et al [5] used the vanishing point to detect line segments. This method requires a lot of memory space and computation time. Nejati et al. [6] use a histogram-based approach. In this method, once the image is captured, pre-processing algorithms, such as dilation vertical and horizontal edge processing, are implemented on the image to obtain horizontal and vertical histograms. These histograms represent the sum of the differences in gray levels between neighboring pixels in an image, columns, and rows. This approach does not work well on images with heavy noise or tilted license plates.

After the development of deep learning, researchers turned to the object detection model based on convolutional neural network (CNN) [7] and its derivatives such as R-CNN (Regions with CNN features) [8], YOLO (You Only Look Once) [9] and SSD (Single Shot MultiBox Detector) [10] which have been widely used for LPR systems. In [11], Li et al. proposed a method to jointly solve license plate detection and recognition by a single deep neural network. They used VGG to extract the low-level CNN features. This method is time-consuming in the training phase.

Xie et al. [12] proposed a CNN-based MD-YOLO framework for multidirectional car license plate detection. This technique efficiently solves rotation problems in real-time scenarios. However, it finds difficulties in detecting small-sized objects.

Li et al. [13] proposed a cascade framework to improve the detection and recognition performance, but the CNN classifiers only detect Latin language plates. In our previous works [14,15], we used a modified Tiny yolov3 method for Moroccan license plate detection and recognition. They modified the number of layers in the feature extraction phase to improve accuracy and added a layer in the detection phase to facilitate detecting small objects. Nevertheless, this method requires a lot of resources in terms of execution time and memory space.

In [16], Slimani et al. proposed a system comprising two stages: (1) the license plate detection stage and (2) the character recognition stage. In the first stage, the license plate candidates are generated based on vertical edges to detect the potential license plate candidates. First, the 2D-WD is used to extract the vertical edges. Then, the high-density areas of the vertical edges are extracted by calculating the maximum entropy areas to detect the potential license plate candidates. These candidates are then classified by training a plate/non-plate CNN classifier to remove the false positive. In the second stage, the characters are segmented first by detecting the empty lines between the characters, and then these segmented candidates are classified by training 42-class CNN classifier to recognize the license plate characters.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 29 September 2023 doi:10.20944/preprints202309.2107.v1

Based on the above, we note that generally, to obtain an accurate detection and recognition of a license plate, the methods proposed by most researchers have two main modules:

- 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.

2. THE MATERIALS AND METHODS

In this section, we dissect our ALPR network architecture. In this article, 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.

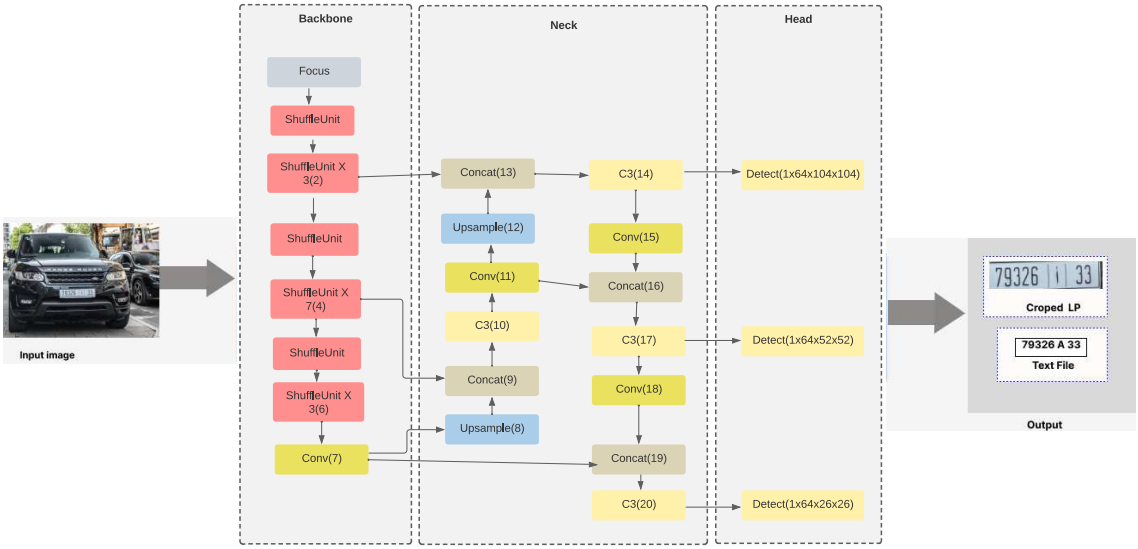

Figure 1. The diagram flowchart of our system.

2.1. YOLOv5

In YOLOv1 [9], a single neural network predicts bounding boxes and class probabilities directly from complete images in a single evaluation, allowing to build of faster models usable in real-time. To do this, the system creates an S X S grid from the input image. Then, it verifies if the searched object is in the center of a grid. Each grid cell predicts B bounding boxes and the confidence scores for these boxes. These confidence scores reflect the model's confidence that the box contains an object and also the accuracy with which it thinks the box predicts. Formally, we define confidence as (1):

$$Pr(Object) * IOU \tag{1}$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 29 September 2023 doi:10.20944/preprints202309.2107.v1

The bounding box consists of (x, y), which represent the center of the box relative to the grid cell boundaries. The width (W) and height (H) are predicted relative to the whole image, and a confidence score that IOU between the predicted box and any ground truth box.

To improve yolov1, YOLOv2 [17] uses Darknet-19 as a backbone, batch normalization to enhance neural network stability, a high-resolution classifier increased from 224*224 to 448*448 for more accuracy, and the use of anchor boxes which are responsible for the prediction of the bounding box and are designed for a given dataset using clustering (k-means clustering)

Yolov3 [18], Yolov3, also called Darknet-53, is a model that has been developed based on ResNet [19] and FPN (Feature-Pyramid Network). It uses skip connections like ResNet and three prediction heads like FPN topology, which allows YOLO-V3 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 mAP.

The YOLOv4 backbone is a deep neural network composed mainly of convolution layers. The main purpose of the Backbone is to extract essential features. The selection of the Backbone is a key step that will improve the object detection performance.

The system consists of three components: a Bag of gifts, which serves to enhance the training cost or modify the training strategy 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 CSPDarknet53, which employs a CSPNet approach to divide the backbone feature map into two segments and subsequently combines 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.

YOLOv5 [21], [22] is an evolution of YOLO algorithm (YOLOv1 [9], YOLOv2 [17], YOLOv3 [18] and YOLOv4 [20]) and a single-stage object detector, it has three important parts like any other single-stage object detector: Model Back Structure, Model Neck and Model Head.

Model Backbone mainly extracts important features from the given input image. Yolov5 is based on the YOLOv4 algorithm and builds on the idea of CSPNet [23]. In YOLO v5, CSP - Cross Stage Partial Networks are used as the Backbone to extract information-rich features from an input image. CSPNet has shown significant improvement in processing time with deeper networks.

Model Neck is primarily used to generate feature pyramids. Feature pyramids are used to extract feature maps of different scales in each layer and fuse the feature maps of the deeper layers with the feature maps of the previous level, which can bring deep semantic information to the shallow layer.

Feature pyramids are very useful and help models work well on invisible data. Other models use different types of feature pyramid techniques like FPN, BiFPN, PANet, etc.

YOLOv5 opts for PANet and adds a bottom-up process after the top-down process. The schematic diagram of the Path Aggregation Network (PAN) structure is shown in Figure 2.

The PAN [24] structure receives the rich semantic information conveyed from the FPN layer from top to bottom. Then, it continues to convey rich spatial information from the bottom to the top. Finally, parameter aggregation is performed, and the feature maps of different scales are obtained through upsampling each time and output to the detection layer. The operation of the Concat layer is the concatenation and fusion of the feature maps from two layers, concatenating the features from the upper layer of the network and the output of the features by each layer in the FPN structure, and output the new features to the next layer of the network.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 29 September 2023 doi:10.20944/preprints202309.2107.v1

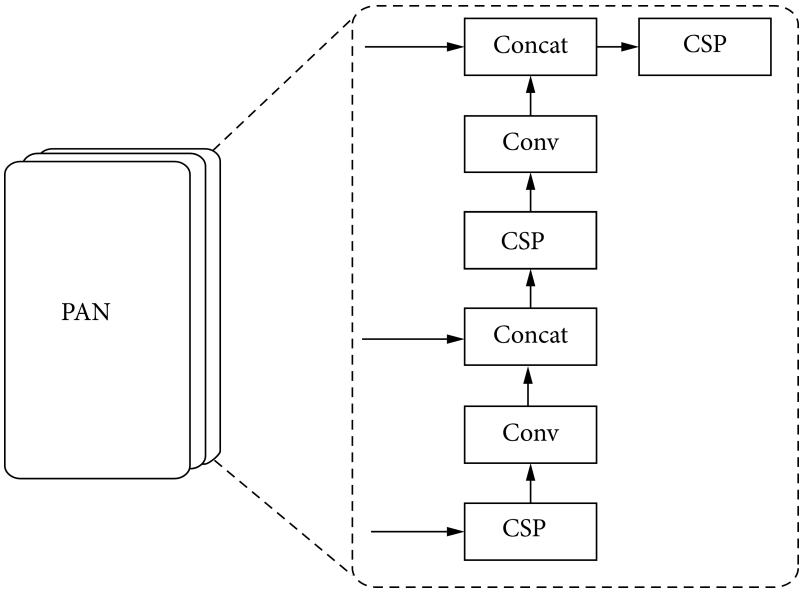

Figure 2. PAN structure.

Finally, to obtain the final detection, the model head is used. It applied anchor boxes on features and generated final output vectors with class probabilities, objectness scores, and bounding boxes. The network structure of the YOLOv5 algorithm is shown in Figure 3.

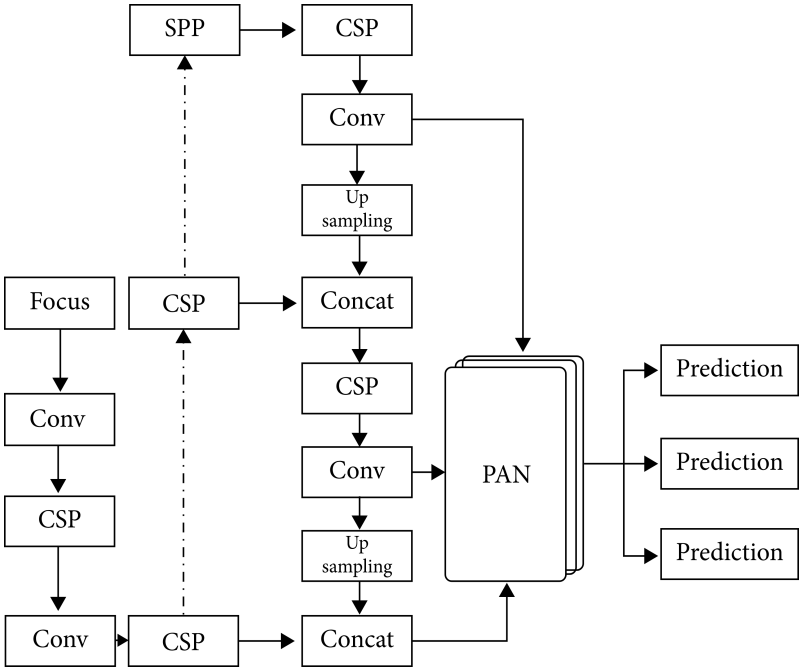

Figure 3. Yolov5 network structure.

2.2. Improved YOLOv5 Network

Yolov5 contains five models in total. YOLOv5 nano is the smallest and fastest model intended for the edge, IoT devices, and with OpenCV DNN support as well. It is less than 2.5 MB in INT8 format and about 4 MB in FP32 format. It is ideal for mobile solutions. YOLOv5s is the smallest model in the family, with about 7.2 million parameters, and is ideal for running inference on the CPU. YOLOv5m is a medium-sized model with 21.2 million parameters. It is the best model for many datasets and training, as it balances speed and accuracy well. YOLOv5l is the large model in the YOLOv5 family with 46.5 million parameters. It is ideal for data sets where we need to detect smaller objects. The last

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 29 September 2023 doi:10.20944/preprints202309.2107.v1

model is YOLOv5x, the largest among the five models and has the highest mAP. However, it is slower than the others and has 86.7 million parameters. Figure 4 compares the different versions of YOLOv5 models.

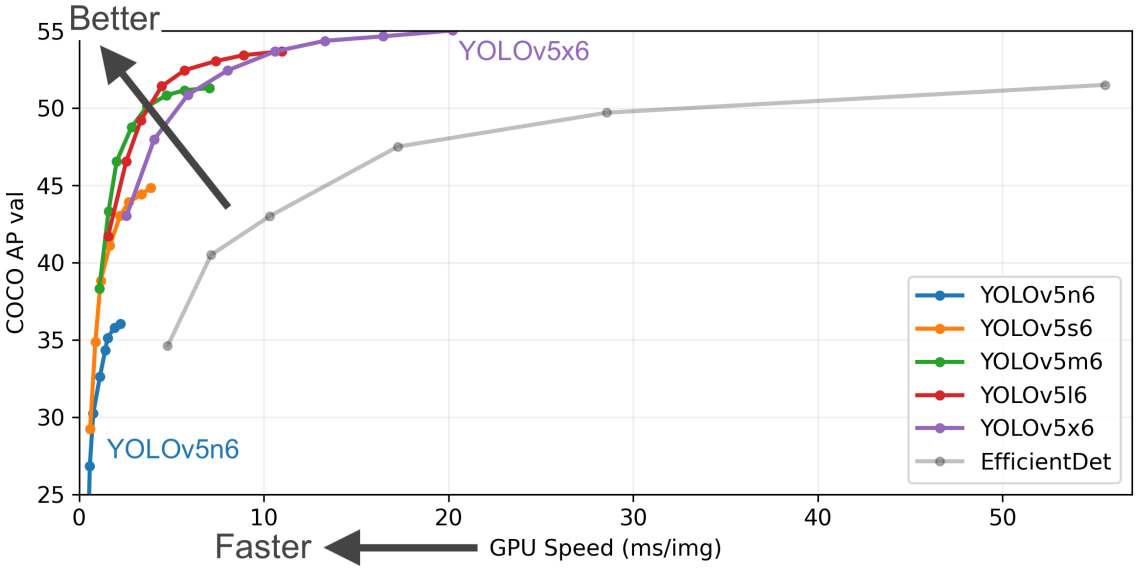

Figure 4. comparison of performance between the different versions of YOLOv5.

In this article, we are looking for a model that can be both usable in IOT devices and more accurate. In this context, there are not many choices. Yolov5n and Yolov5s are the best. Yolov5n is a smaller model, faster but less accurate. Yolov5s, as shown in Figure 4, remains the most suitable choice as it offers the best balance between speed and accuracy.

Additionally, real-world tasks often aim for the best accuracy with reduced computation time, and application scenarios such as license plate detection on highways require low latency. This motivates a series of works towards lightweight architecture design and better speed-accuracy trade-off, including Xception[25], MobileNet[26], MobileNet V2[27], ShuffleNet[28] and CondenseNet [29], etc. To measure computational complexity, there are three widely used metrics: the number of floating point operations or FLOPs, memory access cost (MAC) and degree of parallelism. Ma et al. [30] showed that Shufflenetv2 is the best in comparison with the others.

2.2.1. Shufflenetv2

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. At the beginning of each unit, the input of feature channels is split into two branches with c-c' and c' channels, respectively. 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 1x1 convolutions are no longer group-wise, unlike the original ShuffleNet. This is partially to follow G2 and partially because the split operation already produces two groups. After convolution, the two branches are concatenated. 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. Figure 5

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 29 September 2023 doi:10.20944/preprints202309.2107.v1

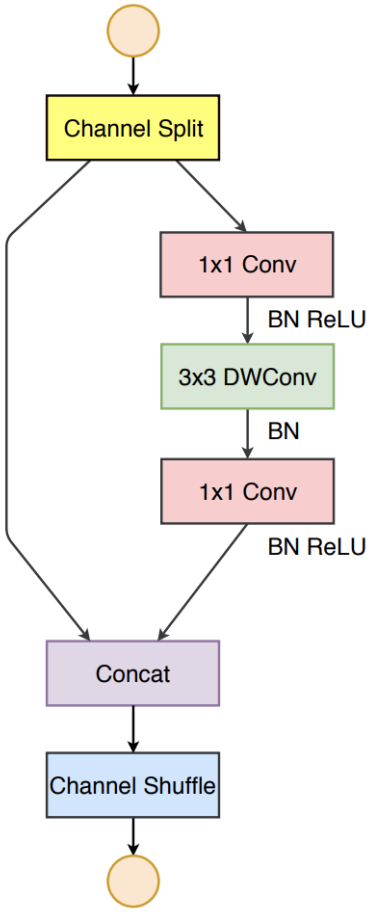

Figure 5. ShufflenetV2 basic unit.

The motivation behind channel split is that alternative architectures, where pointwise group convolutions and bottleneck structures are used, lead to increased memory access costs. Additionally, more network fragmentation with group convolutions reduces parallelism (less friendly for GPU), and the element-wise addition operation, while they have low FLOPs, has high memory access cost. Channel split is an alternative where we can maintain a large number of equally 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 use 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.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 29 September 2023 doi:10.20944/preprints202309.2107.v1

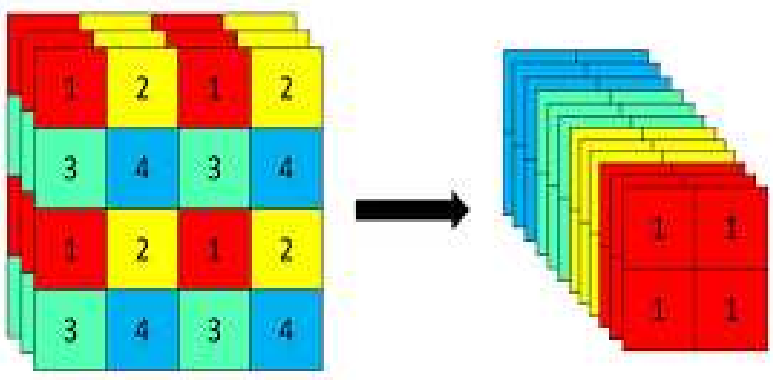

Figure 6. Focus structure slice operation.

3. Experiments and Results

3.1. Dataset and Configurations

To our knowledge, there is no publicly available database 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 divided into three parts: the first part (right) contains the code associated with the identifier of the plate's emission region.

This identification goes from 1 to 89. Then, a letter of the Arabic alphabet 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 has the capability to store the XML files corresponding to the annotated photographs automatically.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 29 September 2023 doi:10.20944/preprints202309.2107.v1

Figure 8. Sample of database images.

In the experiment, the basic environment of the experiment was conducted in Google Colab Pro. Colab is well organized with a GPU (NVIDIA P100) environment, so we used it.

3.2. Evaluation Metrics

Object detection is both a regression and a classification problem. The mean average precision (mAP) metric is used in the experiment. Firstly, the coverage of the predicted detection box P to the ground-truth bounding box G is calculated as (2):

$$UoI(P,G) = \frac{P \cap G}{P \cup G} \tag{2}$$

where IoU (Intersection over Union) represents the degree of overlap between the predicted bounding box and the ground-truth bounding box.

The threshold of the experiment datasets is 0.5; if the IoU is greater than 0.5, the detection is considered successful. Afterward, the recall (3) and precision (4) of each category are calculated separately, where TP is the number of correctly predicted samples, FP is the number of incorrectly predicted samples, and is the actual number of samples in this category:

$$Recall = \frac{TP}{N_c} \tag{3}$$

$$precision = \frac{TP}{TP + FP} = \frac{Recall.N_c}{Recall.N_c + FP} \tag{4}$$

The average precision (AP) for each category is then calculated separately as follows. 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 calculated as the average precision of the category. Finally, the mAP of the entire test set is obtained by averaging the mean accuracy of all categories.

The detection speed is used to evaluate the timeliness of object detection in application scenarios, and the frames per second (FPS) metric is used to evaluate the detection speed, which is the number of images that can be processed per second.

3.3. Performance Comparison

Before starting our test, we first studied the previous comparison between the different models of Yolov5 (Figure 4)

Although YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5x methods have larger mAP than the other methods, they also have a larger FPS than the other methods. These methods have complex network structures and many parameters. This gives them better performance in mAP and worse

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 29 September 2023 doi:10.20944/preprints202309.2107.v1

performance in FPS. They require the platform to be very powerful. This limits their deployment on mobile and embedded devices. YOLOv5n belongs to the lightweight deep learning method. It has a simple network structure and few parameters. Therefore, it has better FPS and worse mAP performance and is more suitable for mobile and embedded devices deployment. However, according to Figure 4, this method is still unable to detect small objects such as license plate components.

Given that YOLOv5x and YOLOv5l are not applicable within the scope of our context, we have made the deliberate decision to solely conduct a comparative analysis between YOLOv5n, YOLOv5s, and YOLOv5m.

We first compared our proposed method with YOLOv5n, YOLOv5s, and YOLOv5m on the VisDrone2021-DET dataset to test their performance in Precision, Recall and mAP. The results are shown in Table 1.

The YOLOv5n, YOLOv5s, YOLOv5m, and our model were trained for a total of 100 epochs using the Visdrone dataset, which included training images, validation images, and test images. The comparative results of the object detection models are presented in Table 1. In addition, this table shows the precision, recall, $mAP_0.5$, and mAP of all models.

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 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**

Generally, running computationally demanding and memory-intensive CNN-based models on resource-constrained mobile and embedded devices is quite difficult. One of the main problems when running CNNs on such devices is the limited amount of available memory. Thus, reducing the CNN memory footprint is crucial for CNN inference on mobile and embedded devices. The CNN memory footprint is determined by the amount of memory required to store the CNN parameters (weights and biases) and intermediate data exchanged between the CNN operators.

In this context, we compared several parameters and memory required between YOLOv5s, YOLOv5m, and our model. As shown in Table 2, the total parameters of our modified YOLOv5 model are 6.60(M) less than YOLOv5s model and 20.4(M) less than 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.

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

Method	Params (Million)	Memory (MB)
YOLOv5n	1.77	3.8
YOLOv5s	7.03	14.4
YOLOv5m	20.88	42.2
Our	**0.44**	**1.3**

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 29 September 2023 doi:10.20944/preprints202309.2107.v1

Considering that for a model to be executed on an embedded system, it must have a reduced number of parameters, be light in size, and be faster. In the previous sections, we have shown that our model is the best in terms of the number of parameters and the size occupied. To better see if our model meets the requirements of real-time detection, i.e., the processing speed (FPS), we have exhaustively compared the average detection accuracy (mAP) and detection speed (FPS) this time on our dataset to intuitively find the best compromise between accuracy and detection speed. The detection speed and mAP results of the proposed method and other comparison methods on our dataset are shown in Table 3 and Figure 9.

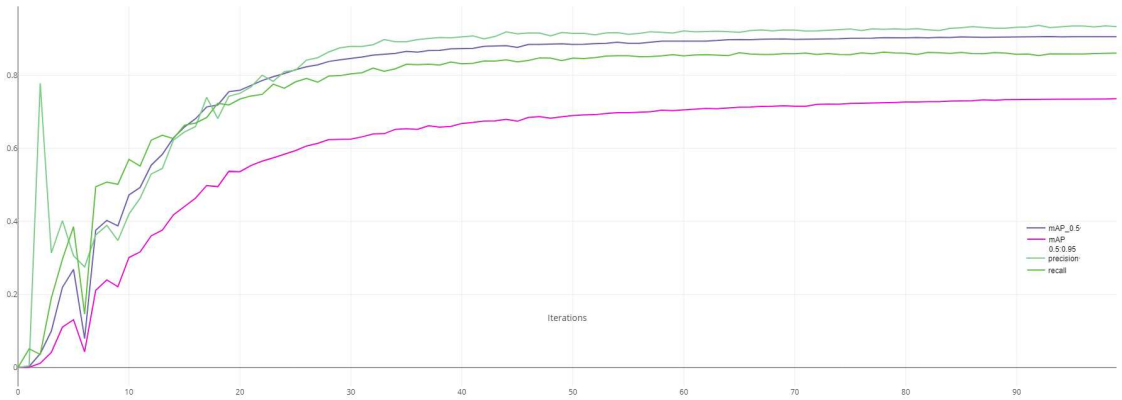

Figure 9. Performance metrics curve of our model on our dataset.

Table 3. 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**

The experimental results show that the proposed method achieves the best balance between real-time detection and detection accuracy. Despite this, our algorithm still meets the requirements of real-time systems.

To prove the effectiveness and the validity of our approach, we have collected a set of videos of real road scenes, and we have selected different images. The results of the detection are shown in Figure 10. We can see that the proposed network can detect small objects such as far LPs. This demonstrates that our proposed improvement works well even for complex real-time road environments.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 29 September 2023 doi:10.20944/preprints202309.2107.v1

Figure 10. ALPRS results.

4. Conclusion

In this paper, we introduced a new lightweight and powerful ALPR system based on YOLOv5, which can achieve an excellent balance between accuracy and speed. In this approach, we used the ShuffleNetV2 method in the backbone part to extract rich features and focus on useful information. The introduction of ShuffleNetV2 allowed us to reduce the detection time, the number of parameters, and the size of our model, which allows the implementation of our approach for real-time license plate detection and recognition on an embedded system with limited computing power and memory.

Author Contributions: Conceptualization, M.E. and R.S.; formal analysis, A.F.; data curation, A.F.; writing—original draft preparation, A.F.; writing—review and editing, Y.F., M.E., R.S. and A.F. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not Applicable.

Informed Consent Statement: Not Applicable.

Data Availability Statement: The data presented in this study are available on request from the first or corresponding author.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

ALPR	Automatic License Plate Recognition
CNNs	Convolutional Neural Networks
YOLO	You Only Look Once
SSD	Single Shot MultiBox Detector
VGG	Visual Geometry Group
FPN	Feature-Pyramid Network
CSPNet	Cross Stage Partial Network
BiFPN	Weighted bi-directional feature pyramid network
PANet	Path Aggregation Network
IoT	Internet of Things
FLOPs	floating point operations
MAC	Memory access cost
GPU	graphics processing unit
mAP	Mean Average Precision
IoU	Intersection over Union

 doi:10.20944/preprints202309.2107.v1

References

1. Zerka, A.; Jawab, F. Contribution to the economic analysis of numerical data of road accidents in Morocco. In Proceedings of the International Conference on Digital Technologies and Applications. Springer, 2022, pp. 133–144.
2. Anagnostopoulos, C.N.E.; Anagnostopoulos, I.E.; Psoroulas, I.D.; Loumos, V.; Kayafas, E. License plate recognition from still images and video sequences: A survey. *IEEE Transactions on intelligent transportation systems* **2008**, *9*, 377–391.
3. Du, S.; Ibrahim, M.; Shehata, M.; Badawy, W. Automatic license plate recognition (ALPR): A state-of-the-art review. *IEEE Transactions on circuits and systems for video technology* **2012**, *23*, 311–325.
4. Le, W.; Li, S. A hybrid license plate extraction method for complex scenes. In Proceedings of the 18th International Conference on Pattern Recognition (ICPR'06). IEEE, 2006, Vol. 2, pp. 324–327.
5. Schreiber, G.; Haran, G.; Zhou, H.X. Fundamental aspects of protein- protein association kinetics. *Chemical reviews* **2009**, *109*, 839–860.
6. Nejati, M.; Paluszny, A.; Zimmerman, R.W. On the use of quarter-point tetrahedral finite elements in linear elastic fracture mechanics. *Engineering Fracture Mechanics* **2015**, *144*, 194–221.
7. Huang, Z.; Wu, J.; Xie, F. Automatic recognition of surface defects for hot-rolled steel strip based on deep attention residual convolutional neural network. *Materials Letters* **2021**, *293*, 129707.
8. Faster, R. Towards real-time object detection with region proposal networks. *Advances in neural information processing systems* **2015**, *9199*, 2969239–2969250.
9. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 779–788.
10. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. Ssd: Single shot multibox detector. In Proceedings of the European conference on computer vision. Springer, 2016, pp. 21–37.
11. Li, H.; Wang, P.; Shen, C. Towards end-to-end car license plates detection and recognition with deep neural networks. CoRR abs/1709.08828 (2017).
12. Xie, L.; Ahmad, T.; Jin, L.; Liu, Y.; Zhang, S. A new CNN-based method for multi-directional car license plate detection. *IEEE Transactions on Intelligent Transportation Systems* **2018**, *19*, 507–517.
13. Li, H.; Wang, P.; You, M.; Shen, C. Reading car license plates using deep neural networks. *Image and Vision Computing* **2018**, *72*, 14–23.
14. Fadili, A.; Aroussi, M.E.; Fakhri, Y. Real-Time Moroccan License Plate Recognition Based on Improved Tiny-YOLOv3. In Proceedings of the International Conference on Advanced Intelligent Systems for Sustainable Development. Springer, 2020, pp. 600–612.
15. Fadili, A.; El Aroussi, M.; Fakhri, Y. A one-stage modified Tiny-YOLOv3 method for Real time Moroccan license plate recognition. *International Journal of Computer Science and Information Security (IJCSIS)* **2021**, *19*.
16. Slimani, I.; Zaarane, A.; Al Okaishi, W.; Atouf, I.; Hamdoun, A. An automated license plate detection and recognition system based on wavelet decomposition and CNN. *Array* **2020**, *8*, 100040.
17. Redmon, J.; Farhadi, A. YOLO9000: better, faster, stronger. In Proceedings of the Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 7263–7271.
18. Redmon, J.; Farhadi, A. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767* **2018**.
19. Targ, S.; Almeida, D.; Lyman, K. Resnet in resnet: Generalizing residual architectures. *arXiv preprint arXiv:1603.08029* **2016**.
20. Bochkovskiy, A.; Wang, C.Y.; Liao, H.Y.M. YOLOv4: Optimal Speed and Accuracy of Object Detection, 2020, [arXiv:cs.CV/2004.10934].
21. Thuan, D. Evolution of Yolo algorithm and Yolov5: The State-of-the-Art object detention algorithm **2021**.
22. Jocher, G.; Stoken, A.; Borovec, J.; Christopher, S.; Laughing, L.C. ultralytics/yolov5: v4. 0-nn. SiLU () activations, Weights & Biases logging, PyTorch Hub integration. *Zenodo* **2021**.
23. Wang, C.Y.; Liao, H.Y.M.; Wu, Y.H.; Chen, P.Y.; Hsieh, J.W.; Yeh, I.H. CSPNet: A new backbone that can enhance learning capability of CNN. In Proceedings of the Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops, 2020, pp. 390–391.
24. Liu, S.; Qi, L.; Qin, H.; Shi, J.; Jia, J. Path aggregation network for instance segmentation. In Proceedings of the Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 8759–8768.

25. Chollet, F. Xception: Deep learning with depthwise separable convolutions. In Proceedings of the Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 1251–1258.
26. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861* **2017**.
27. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.C. Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 4510–4520.
28. Zhang, X.; Zhou, X.; Lin, M.; Sun, J. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In Proceedings of the Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 6848–6856.
29. Huang, G.; Liu, S.; Van der Maaten, L.; Weinberger, K.Q. Condensenet: An efficient densenet using learned group convolutions. In Proceedings of the Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 2752–2761.
30. Ma, N.; Zhang, X.; Zheng, H.T.; Sun, J. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In Proceedings of the Proceedings of the European conference on computer vision (ECCV), 2018, pp. 116–131.