Automatic Number Plate Recognition using YOLOv5

Agnia CODREANU,
R&D Department, Beia CONSULT INTERNATIONAL, Bucharest, Romania
agnia.codreanu@stud.etti.upb.ro

Razvan-Alexandru BRATULESCU,
R&D Department, Beia CONSULT INTERNATIONAL, Bucharest, Romania
razvan.bratulescu@beia.ro

Pavel MURESAN,
R&D Department, Beia CONSULT INTERNATIONAL, Bucharest, Romania
pavel.muresan@beia.ro

Mari-Anais SACHIAN,
R&D Department, Beia CONSULT INTERNATIONAL, Bucharest, Romania
anais.sachian@beia.ro

George SUCIU,
R&D Department, Beia CONSULT INTERNATIONAL, Bucharest, Romania
george@beia.ro

Abstract
Automatic number plate recognition (ANPR) is a system that uses a vehicle's number plate to identify it. The popularity of ANPR technology has increased over the past years as a result of its variety of applications in different fields. To build the ANPR system two stages are considered. In the first stage, to segregate the number plate from the rest of the image, we utilize the pre-existing open-source system YOLOv5. You Only Look Once (YOLO) is a state-of-the-art, real-time object detection algorithm that utilizes regression to predict bounding boxes and associated class probabilities for the whole image using a single convolutional neural network (CNN). In the second stage, once the number plate is detected, it is cropped from the image and is passed on to detect characters using optical character recognition (OCR). OCR identifies text inside an image file and converts it into a machine-readable text form to be used for data processing. For license plate recognition, compact classifier models combined with YOLOv5 are experimented from the perspective of obtaining high efficiency of implementation in Raspberry-Pi embedded systems. Based on investigated compact solutions analysis, we determine the optimal solution with maximum accuracy in terms of minimum response time. The solution is integrated into the Raspberry Pi platform together with software modules that allow access via the Internet and the management of the database containing the recognized numbers. This paper aims to create an optimal ANPR system for embedded systems by implementing a solution that increases the performance, efficiency, scalability and accuracy of automatic license plate recognition software.

Keywords: Optical Character Recognition (OCR), Automatic Number Plate Recognition (ANPR), YOLO, Real-Time Object Detection System, Convolutional Neural Network (CNN).

1. Introduction
Over the past years, several cities throughout the world have placed the transformation of smart cities at the top of their priority lists. One of the most often used sensing technologies for providing huge data and fresh insights on observing vehicle movements and mobility patterns in controlling urban mobility is camera technology, in particular, Automatic Number Plate Recognition (ANPR) [1]. This technology represents an advanced and intelligent system that has gotten popular due to its various real-life use cases (such as

traffic violation, parking management, toll automation, intelligent transportation etc.) without any human intervention [2].

Utilizing a camera, ANPR entails gathering footage of license plates from the intended scene. The acquired image is then further processed by a number of recognition algorithms based on image processing in order to convert the image into a text entry with alphanumeric characters [3].

This study aims to create an optimal ANPR system for Raspberry-Pi embedded systems by using object detection algorithms combined with different feature extractors for maximum accuracy in terms of minimum response time. The structure of this paper is as follows: we review some related works regarding ANPR systems in Chapter 2. Chapter 3 encompasses details of the proposed system. Then, we present the implementation in Chapter 4 and conclude our work in Chapter 5.

2. Literature Survey

In the past years, a variety of ANPR systems have been offered. License plates and characters were found using image binarization or grey-scale analysis, which were then followed by custom feature extraction techniques and traditional machine learning classifiers. The state-of-the-art began shifting with the emergence of Deep Learning, and today, many works use convolutional neural networks (CNNs) because of their excellent accuracy for general object identification and recognition [4].

Venkatanaresh et al. [5] proposed research that used CNNs to deliver effective deep learning principally based on ANPR system for the detection and recognition of license plates (LPs). To simultaneously identify and classify LPs and characters, two fully convolutional one-stage object detectors were used, followed by an assembly module that outputs the LP strings. The VLPR model was employed, using YOLOv2 to detect license plates and optical character recognition (OCR) to identify characters on license plates. The suggested solution treats license plate detection and recognition as a single task accomplished using a single network.

Adak et al. [6] proposed a system divided into two regions of interest. In the 1st region of interest, the number plate was separated from the rest of the image using YOLOv3. After the license plate was located, it was clipped from the image and used to find the individual characters in the second region of interest. Filtering of content for plate license recognition was done using a variety of image processing algorithms. The program was created and put into use on images that were impacted by different environmental conditions (such as over-exposed image, blurred image, image with angular perspective, randomly chosen custom font, rotated image, shaded image, straight image, image with a coloured number plate background, multiline image). It provided good accuracy for images with a straight front, and for images affected by environmental factors, accuracy was hindered up to 12%, compared to other systems without the right pre-processing techniques that have up to 42% hindrance.

Baghdadi [7] presented a comparative analysis between ANRP systems implemented using one-stage object detection algorithms. The work consisted of analyzing the mean Average Precision (mAP) of Single Shot MultiBox Detector (SSD), and YOLOv4 on CENPARMI (Centre for Pattern Recognition and Machine Intelligence) and UFPR-ALPR (Federal University of Parana - Automatic License Plate Recognition) datasets. The CENPARMI dataset consists of license plates' images from the United States and Canada, while the UFPR-ALPR dataset consists of license plates' images from Brazil. Although they achieved good mAP results of 95.47 % (ResNet-SSD) and 95.45 % (InceptionV2-SSD) with the SSD model during the experiment, they reached the highest mAP of 97.46 % and 97.78 % with the YOLOv4 model on CENPARMI and UFPR-ALPR datasets, respectively. They closely examined the aforementioned object detectors to build a model that can balance mAP, speed, and memory and discovered that the more parameters a model has, the better the outcomes of detection. However, this also affects the speed of an object identification operation.

3. Proposed System

To build the ANPR system, two stages are considered, as shown in the following figure:

- Plate Region Detection and Extraction;
- Characters Segmentation and Recognition.

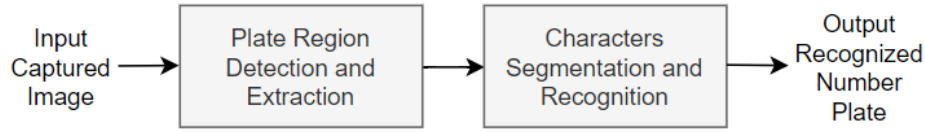

Fig. 1. Diagram of the ANPR system.

3.1. Plate region detection and extraction

In the first stage, to segregate the number plate from the rest of the image, we utilize the pre-existing open-source system YOLOv5. You Only Look Once (YOLO) is a state-of-the-art, real-time object detection algorithm that utilizes regression to predict bounding boxes and associated class probabilities for the whole image using a single convolutional neural network.

Three components make up the YOLO network:

- Backbone: a convolutional neural network that gathers and creates picture features on various scales;
- Neck: a number of layers that blend and mix visual characteristics before sending them on to prediction;
- Head: consumes neck features and performs class and box prediction procedures [8].

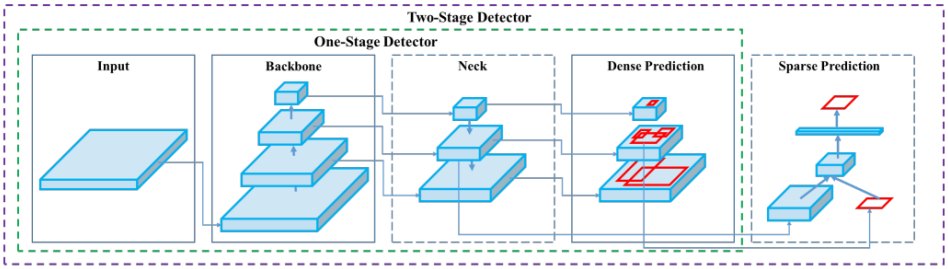

Fig. 2. Object detector.
Source: Figure from [9]

Every member of the YOLO family is based on an earlier model and aims to make that model better. For the backbone of the YOLOv5 model, the authors created CSPDarknet, which consists of an incorporated cross-stage partial network (CSPNet) into Darknet. By integrating the gradient changes into the feature map and resolving the issues with repeated gradient information in large-scale backbones, CSPNet reduces the model's parameters and FLOPS (floating-point operations per second), ensuring inference speed and accuracy while also shrinking the model's size. The authors used a path aggregation network (PANet) for the neck of the model to improve information flow. The transmission of low-level features is boosted by the use of a feature pyramid network (FPN) structure by PANet with an improved bottom-up path. In addition, adaptive feature pooling, which connects feature grid and all feature levels, is employed to enable each feature level's important information to spread instantly to the subsequent subnetwork. In lower layers, PANet enhances the use of precise localization signals, which can obviously increase the object's location accuracy. The Yolo layer, which is the model's head, creates feature maps in three distinct sizes (18 × 18, 36 × 36, 72 × 72) to accomplish multi-scale prediction. This allows the model to handle tiny, medium, and large objects [7],[10].

3.1.1. The choice of feature extractor

Although YOLOv5 can do classification tasks by itself, using object detection alongside classification models can boost performance by around 5–25%.

A number of feature extractor models are available for each object detector. The accuracy of the model improves along with the number of parameters. However, when selecting one of the feature extractor models, there are still more aspects to take into account. Depending on the use of an ANPR system, one should attempt to strike a balance and make a decision between speed, accuracy, and memory [7].

3.1.1.1. MobileNet

The depthwise separable convolutions, a type of factorized convolutions that factorize a standard convolution into a depthwise convolution and a 1 × 1 convolution known as a pointwise convolution, are the foundation of the MobileNet model. Each input channel for MobileNet receives a single filter due to depthwise convolution. In order to merge the outputs with the depthwise convolution, the pointwise convolution is complemented by a 1 × 1 convolution. In one step, a standard convolution filters and combines inputs to create a new set of outputs. This is divided into two layers by the depthwise separable convolution:

a layer for combining and a layer for filtering. The computation and model size are significantly decreased as a result of this factorization.

In comparison to other neural networks with the same accuracy, MobileNet uses 3 × 3 depthwise separable convolutions, which requires 8–9 times less processing [11].

3.1.1.2. EffNet

EffNet architecture is a variation of the MobileNet building block. Two linear layers are created from the 3 × 3 depthwise convolution. This enables pooling after the first spatial layer, saving calculations in the subsequent layer. Following that, the subsampling is divided along the spatial dimensions. A 1 × 2 max pooling kernel is then applied after the initial depth-wise convolution. The common pointwise convolution is swapped out for 2 × 1 kernels and a commensurate stride for the second subsampling. This almost equals the number of FLOPs while providing marginally higher accuracy [13].

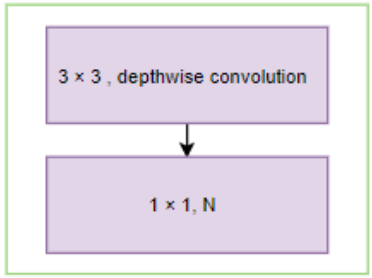

Fig. 3. MobileNet convolutional block.
Source: Figure from [12]

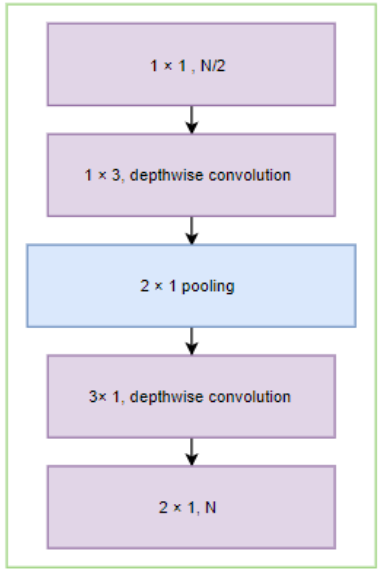

Fig. 4. EffNet convolutional block.
Source: Figure from [12]

3.2. Characters segmentation and recognition

In the second stage, once the number plate is detected, it is cropped from the image and is passed on to detect characters using optical character recognition. OCR software converts a two-dimensional image of text (which could be either handwritten or machine-printed) into text that is readable by machines. The optical character recognition process consists of multiple sub-processes [14].

The general sub-processes are:

- Pre-processing of the Image;
- Text Localization;
- Character Segmentation;
- Character Recognition;
- Post-processing.

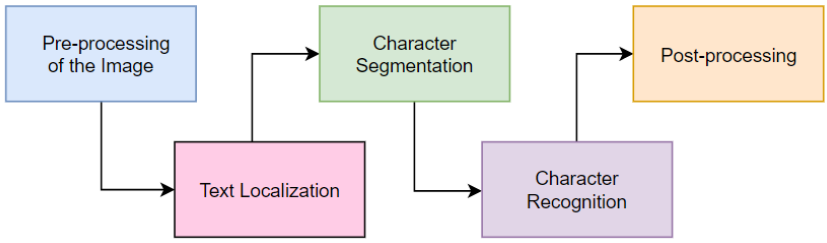

Fig. 5. OCR sub-processes.

3.2.1. Tesseract OCR

Tesseract is a free and open-source OCR engine that complies with the Apache 2.0 license. Using a command line or an API, Tesseract may be used to directly extract written text from images. Many different languages are supported. It is compatible with a wide range of programming languages and frameworks due to its fully featured API. Tesseract can be used in conjunction with the current layout analysis to identify text within a large document or with an outside text detector to identify text from an image of a single text line [14].

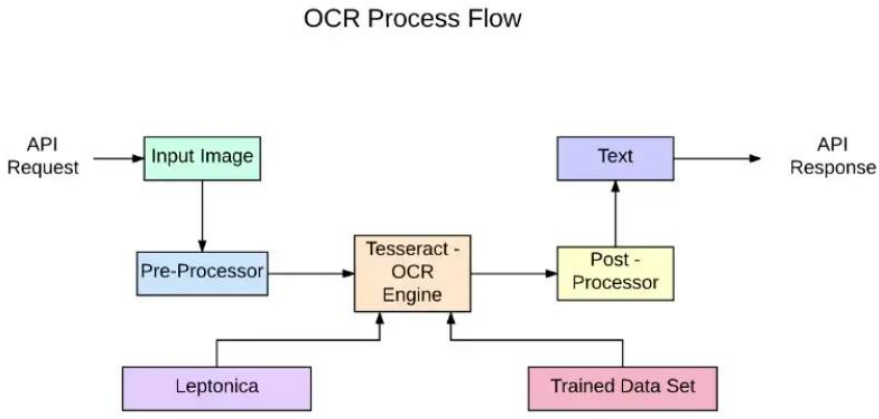

Fig. 6. Tesseract OCR process flow.
Source: Figure from [15]

4. Implementation

The main part of this work is purely software-based. Therefore, we only require a small amount of hardware to set up the text recognition for license plates. The following components are used: Raspberry Pi 4 (with installed software - Raspberry Pi OS), USB webcam and power connection (micro USB cable and USB adapter).

The Raspberry Pi is a credit card-sized computer that can perform a variety of low-power applications, including processing text and connecting to the Internet. The USB webcam is connected to the Raspberry Pi setup in order to give real-time input. And lastly, the power connection is used for power supplying the board.

The USB camera captures video frames. The taken image is sent to Raspberry Pi to process the data. To detect the vehicle license plate, we utilize YOLOv5 alongside compact classifier models (MobileNet and EffNet), which are greatly increasing the performance, efficiency, scalability and accuracy of the model. To train the model, we use a freely available dataset from Kaggle, which consists of nearly 500 images of vehicles. To extract the text, we apply OCR Tesseract for character recognition. The recognized number plate is then recorded in a database.

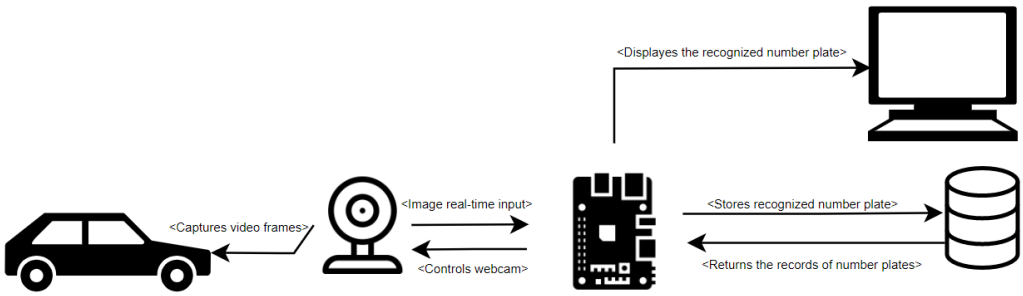

Fig. 7. Implementation diagram.

5. Conclusion

The popularity of ANPR technology has increased over the past years as a result of its variety of applications in different fields. The recent advances in Parallel Processing and Deep Learning have contributed to improving many computer vision tasks, such as Object Detection/Recognition and OCR, which clearly benefit ANPR systems. In fact, deep CNNs have been the leading machine learning technique applied for vehicle and license plate detection.

The first and most important phase in the ANPR system is object detection, which we examined in this research. High precision is no longer the only important criterion in object detection. But it's essential to use a model that can strike a balance between precision, speed, and memory. Since the ANPR systems are mostly useful in real-time applications, we evaluated the YOLOv5 model, known as one of the fastest real-time object detection models. Feature extraction is a sub-task of object detection; hence we examined different feature extractors with the YOLOv5 model and found out that algorithms perform similarly if tuned properly.

The goal of this work is to determine the optimal solution for embedded systems in terms of minimum response time. For this purpose, YOLOv5 with MobileNet as feature extractor does a better job, conditioning the reduction of parameters, thus reducing the size of the model and improving the interface time. However, in terms of performance, efficiency and accuracy, it is slightly exceeded by YOLOv5 combined with the EffNet classifier model.

Although we used pre-trained networks to train with the chosen models, we expect that more data will enable the model to generalize more effectively and deliver more accurate results. The dataset properties, the model, and the hardware system's accuracy, speed, and memory performance should therefore be taken into account in order to have the required ANPR system.

Acknowledgements
This paper was partially supported by UEFISCDI Romania and MCI through BEIA projects SOLID-B5G, 5G-SAFE+, IMMINENCE and by European Union's Horizon 2020 research and innovation program under grant agreement No. 101016567 (VITAL-5G).

References

[1] Tang, J., Wan, L., Schooling, J., Zhao, P., Chen, J., & Wei, S. (2022), Automatic number plate recognition (ANPR) in smart cities: A systematic review on technological advancements and application cases, Cities, 129, 103833.
[2] Klingler, N. (2022, August 22), Automatic Number Plate Recognition (ANPR) – 2022 Guide. viso.ai. https://viso.ai/computer-vision/automatic-number-plate-recognition-anpr/
[3] Mufti, N., & Shah, S. A. A. (2021), Automatic number plate Recognition: A detailed survey of relevant algorithms, Sensors, 21(9), 3028.
[4] Silva, S. M., & Jung, C. R. (2018), License plate detection and recognition in unconstrained scenarios, In Proceedings of the European conference on computer vision (ECCV) (pp. 580-596).
[5] Venkatanaresh, M., Roshini, V., & Harshitha, N. Recognition and Detection of Vehicle License Plates Using Convolutional Neural Networks.
[6] Adak, R., Kumbhar, A., Pathare, R., & Gowda, S. (2022), Automatic Number Plate Recognition (ANPR) with YOLOv3-CNN. arXiv preprint arXiv:2211.05229.
[7] Baghdadi, N. (2020), License Plate Detection Using One-stage Object Detection Algorithms (Doctoral dissertation, Concordia University).
[8] Solawetz, J. (2022, November 21), What is YOLOv5? A Guide for Beginners, Roboflow Blog. https://blog.roboflow.com/yolov5-improvements-and-evaluation/
[9] Bochkovskiy, A., Wang, C. Y., & Liao, H. Y. M. (2020), Yolov4: Optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934.
[10] Xu, R., Lin, H., Lu, K., Cao, L., & Liu, Y. (2021), A forest fire detection system based on ensemble learning. Forests, 12(2), 217.
[11] Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., ... & Adam, H. (2017), Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861.
[12] Lebedev, V. (2018), Algorithms for speeding up convolutional neural networks (Doctoral dissertation, Ph. D. thesis, Skoltech, Russia).
[13] Freeman, I., Roese-Koerner, L., & Kummert, A. (2018, October), Effnet: An efficient structure for convolutional neural networks, In 2018 25th ieee international conference on image processing (icip) (pp. 6-10). IEEE.
[14] Zelic, F. (2022, October 17), Tesseract OCR in Python with Pytesseract & OpenCV. Nanonets AI & Machine Learning Blog. https://nanonets.com/blog/ocr-with-tesseract/
[15] Parthasarathy, B. (2018, April 7), Build your own OCR(Optical Character Recognition) for free. Medium. https://medium.com/@balaajip/optical-character-recognition-99aba2dad314