A Robust Automatic License Plate Recognition System for Embedded Devices

Lucas S. Fernandes[4], Francisco H. S. Silva[1,2], Elene F. Ohata[1,2], Aldísio Medeiros[1,2], Aloísio V. Lira Neto[3], Yuri L. B. Nogueira[5], Paulo A. L. Rego[4,5], and Pedro Pedrosa Rebouças Filho[1,2](✉)

[1] Laboratório de Processamento Digital de Imagens, Sinais e Computação Aplicada, Instituto Federal de Federal de Educação, Ciência e Tecnologia do Ceará, Fortaleza, Ceará, Brazil
pedrosarf@ifce.edu.br,
{herculessilva,eleneohata,aldisio.medeiros}@lapisco.ifce.edu.br

[2] Programa de Pós-Graduação em Engenharia de Teleinformática, Universidade Federal do Ceará, Fortaleza, Ceará, Brazil

[3] Polícia Rodoviária Federal, Rio Largo, Brazil
aloisio.lira@prf.gov.br

[4] Group of Computer Networks, Software Engineering and Systems (GREat), Universidade Federal do Ceará, Fortaleza, Brazil
lucasfernandes42@lia.ufc.br, pauloalr@ufc.br

[5] Department of Computing, Federal University of Ceará, Fortaleza, Ceará, Brazil
yuri@dc.ufc.br

Abstract. Automatic License Plate Recognition (ALPR) systems are used in many real-world applications, such as road traffic monitoring and traffic law enforcement, and the use of deep learning can result in efficient methods. In this work, we present an ALPR system efficient for edge computing, using a combination of MobileNet-SSD for vehicle detection, Tiny YOLOv3 for license plate detection and OCR-net for character recognition. This method was evaluated in two datasets on a NVIDIA Jetson TX2 system, obtaining 96.87% of accuracy and 8 FPS of framerate in a public real-world scenario dataset and achieving 90.56% of accuracy and 11 FPS of framerate in a private dataset of traffic monitoring images, considering the recognition of at least six characters. It is faster than related works with similar deep learning approaches, that achieved at most 2 FPS, and slightly inferior in accuracy, with less than 10% of difference in the worst scenario. This shows the proposed method is well balanced between accuracy and speed, thus, suitable for embedded devices.

Keywords: ALPR · OCR · Deep learning · Edge computing · Embedded systems · Jetson TX2

© Springer Nature Switzerland AG 2020

R. Cerri and R. C. Prati (Eds.): BRACIS 2020, LNAI 12319, pp. 226–239, 2020.
https://doi.org/10.1007/978-3-030-61377-8_16

1 Introduction

The Automatic License Plate Recognition (ALPR) problem consists in detecting and reading one or more license plates (LP) in images. The ALPR systems are used in a relevant number of real-world applications, such as road traffic monitoring, automatic toll collection, traffic law enforcement, and parking lot access control [3].

In general, ALPR systems are composed of the following stages: the vehicle's LP region is located and cropped from the input image, LP's characters are segmented and classified, thus, reading the LP. However, recent approaches first detect the regions of the vehicle before the LP detection in order to reduce false positives and the processing time [7,9,10,16,17]. Also, recent works replace character recognition and classification for character detection, combining both stages [10,16,17] while some works use completely segmentation-free approaches [2,5,6].

In each country or region, LPs obey some established patterns. In Brazil, for example, most LPs are composed of a white background with black characters or a red background with white characters. Besides considering the possible differences between LPs, ALPR systems have to deal with the variation of the quality of the images, which may differ in illumination, shadows, blur, inclinations, and other kinds of distortions.

Therefore, many recent approaches have used robust deep learning techniques [9,10,16,17], as it has improved the state-of-the-art of object detection, speech recognition, among others [11]. These methods achieve good efficiency on a high-end Graphic Processing Unit (GPU) but may be too costly for local processing in weaker devices, being more suitable for cloud computing deployment. This approach using heavy deep learning methods incurs significant latency, energy, and financial overheads and also raises privacy concerns [13], thus, limiting the possibilities of real-world applications.

In this context, a better approach is to use edge computing, i.e., computing the data locally on small and low powered edge devices, as this approach is more attractive to several applications, such as robotics, drone-based surveillance, and autonomous driving [13].

In this work, we propose a complete ALPR system for Brazillian LPs based on the combination of deep learning object detection techniques and efficient enough for embedded system execution. Our main goal is to achieve a balance between accuracy and timing using convolutional neural networks (CNNs) to detect and read an LP aiming a suitable performance in realistic scenarios, allowing the extension of real-world applications through edge computation.

The remainder of this paper is organized as follows. We review related works in Sect. 2. The materials used in our experiments are presented in Sect. 4. The details of the proposed system are described in Sect. 3. In Sect. 5, we report and discuss the experimental results. Finally, conclusions and future works are given in Sect. 6.

2 Related Works

In this section, we will present several works proposed for the different stages of an ALPR system, Vehicle Detection, and LP Detection and Recognition (LPDR). Bringing performance data in precision and execution time of the methods, since the focus of our work is on-time processing real, we must pay attention not only to the efficiency of the method but also to the response time.

2.1 Vehicle Detection

The first stage of an ALPR system is the detection of the vehicle since the LP must be attached to its body. The system's hit rate is highly dependent on the quality of the vehicle's detection, and once the detection method returns a closed image of the vehicle where the LP is cut, or even no LP appears, the system will not be able to recognize all characters and no characters belonging to that vehicle's LP. Next, we will discuss vehicle detection methods proposed by different authors.

Wang *et al.* [18] proposed a new structure called Envolving Boxes, which determines and refines the object boxes through different representations of attributes of each object. A gyro-fine network (FTN) is responsible for the refinement of the boxes. The method was evaluated using Faster R-CNN in the DETRAC benchmark, where it achieved an improvement of 9.5% mAP, running at 9–13 FPS on an Nvidia Titan X GPU.

Sang *et al.* [15] proposed a CNN based on Yolov2, Yolov2_Vehicle. During the training of the network, the K-means ++ algorithm groups the bounding and anchoring boxes. Other improvements were imposed, such as the normalization of object boxes to improve the method of calculating losses, removal of repeated convolutional layers, and the merging of attributes from different layers in order to improve the extraction of attributes. The Yolov2_Vehicle was tested in the vehicle dataset of the Beijing Institute of Technology (BIT), reaching 94.78% mAP and running on 4 Nvidia Tesla K80 GPUs.

The Faster R-CNN with Envolving Boxes proposed by Wang et al. [18] proved unable to process real-time images (30 FPS, for example) on a medium-performance GPU. While Yolov2_Vehicle, proposed by Sang et al. [15], required 4 GPUs for use in training and validation. Therefore, both methods may not achieve a satisfactory framerate in limited systems used in embedded applications, such as Jetson TX2.

2.2 License Plate Detection and Recognition

After detecting the vehicle, there are two more crucial steps for the operation of the ALPR, and these are the Detection and Recognition of LP characters (LPDR). Below we describe works that proposed LPDR methods, which used classical attribute extractors in Computer Vision, Convolutional Neural Networks (CNNs), as well as Machine Learning algorithms.

Bulan *et al.* [2] proposed a method for recognizing LP where the first step consists of identifying flaws in the detection of the LP through the legibility classification of the characters present in the LP—using the Transfer Learning technique, with CNN AlexNet as feature extractor and Linear kernel Support Vector Machine (SVM) as classifier. The character recognition stage, on the other hand, used the HOG and LeNet extractors in conjunction with the SVM-Linear classifier. The method achieved more than 99% accuracy, running on an Nvidia GTX 570 GPU, but with a frame rate of 0.5 FPS.

Björklund *et al.* [1] used LP synthetic images from the European Union (EU) to train LP detection and recognition CNNs. The detection task validation was performed in the AOLP dataset, where it performed an accuracy of 99.30%, as well as the recognition task that reached 99.80%. The methods together required an average of 845 ms to process each 640 × 480 image on a Jetson TX1 embedded system, while on an Nvidia GTX GPU 1080, the same procedure took 25.5 ms.

The method proposed by Bulan *et al.* [2] is not feasible for a real-time application, since its framerate during the validation process was 0.5 FPS. Björklund *et al.* [1] used synthetic images for training its detection and reconnaissance networks, but does not concern itself with vehicle detection, and its validation was performed with images with and without vehicles, but all with license plates, which does not match the purpose of our method.

2.3 Complete ALPR Systems

Some studies have proposed the complete ALPR system, from vehicle detection, through detection to LP recognition. This type of system receives an image with vehicles present, returning the license plate characters of each vehicle in the image. Some real-time applications use this type of system, such as parking lots, speed cameras, and police vehicles. We cite some examples of Complete ALPR Systems below.

Laroca *et al.* [10] implemented a complete Automatic Plate Recognition (ALPR) system composed of three versions based on the YOLO (You Only Look Once) architecture: Yolov2 used in vehicle detection, Fast-Yolov2 responsible for plate detection on a given vehicle, and CR-Net which is a version of YOLO adapted for the detection and recognition of license plate characters. This method reached 95.90% accuracy among the license plates present in Dataset UFPR-ALPR [9], also proposed by the author. The experiment ran at 73 FPS on a high-capacity GPU.

Silva and Jung [17] proposed an ALPR method for LP images in different conditions of visibility, perspectives, and projections. This method uses a network architecture called Warped Planar Object Detection Network (WPOD-NET), which detects the LP and performs a perspective readjustment to assist in character recognition, which is the next step to LP detection. This method was evaluated in OpenALPR Datasets (types BR and EU), SSIG, and AOLP (RP), as well as in the Dataset proposed in their work, the CD-HARD, which contains LP at different angles and distances, presenting greater difficulty for systems of ALPR. The method reached 93.52% on OpenALPR-US, 91.23% on

OpenALPR-BR, LP datasets in frontal perspectives, while on CD-HARD, it reached 75.00% LP accuracy at higher angles and distances, running at 5 FPS on an Nvidia GPU Titan X.

In our work, we propose a complete ALPR system for embedded applications in real-time, since, every day, the need for recognition of LP in security systems and public security in general, both for storing information and in parking with access control, as well as consultations carried out by security agents and also record of infractions in the house of speed cameras. We will compare our method with those previously mentioned, Laroca *et al.* [10] and Silva *et al.* [17], both using a server machine and a device created for embedded applications, since both validated their methods based on LP from Brazil, which is the type that we will validate our work.

3 Proposed Methodology

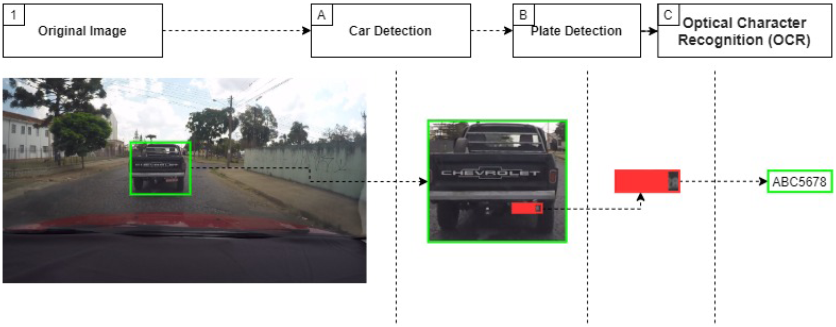

Fig. 1. Proposed methodology flow chart (we covered the LP to maintain the vehicle's anonymity).

The proposed pipeline for LP recognition illustrated in Fig. 1, and is composed of: (A) car detection, (B) LP detection, and (C) LP character recognition. The first step is the vehicle detection using MobileNet-SSD; the detected vehicles are isolated from the rest of the input image. These isolated vehicles are the input for the LPD-net, because detecting the vehicle first may decrease the number of false positives and result in better images with larger and easier LPs for posterior detection. The LPD-net identifies LPs for each vehicle image. The next step consists of the LP character recognition using the OCR-net. Finally, the characters are replaced according to Brazilian LPs patterns.

3.1 Vehicle Detection

Since vehicles are common objects in pre-trained weights of usual deep learning object detection approaches, we decided not to train new weights from scratch.

The SSD-300 [12] with MobileNet [8] as the backbone and PASCAL-VOC [4] pre-trained weights is fast and accurate enough for this approach, even without any additional changes or training in the model. So, the MobileNet-SSD detects the vehicles in the input image, and then they are isolated in separate images that will be used in the next stages.

3.2 License Plate Detection

For each vehicle image, the LPs must be detected and isolated. In this work, we use the Tiny YOLOv3 architecture [14] changing the last layer for one class detection, resulting in the License Plate Detection Network (LPD-net), as shown in Table 1. The network is small and, thus, its speed should be efficient for most of the systems, including embedded systems.

Table 1. License Plate Detection Network (LPD-net): a modification of Tiny YOLOv3 for one class output

	Layer	Filters	Size	Input shape	Output shape
0	conv	16	$3 \times 3/1$	$414 \times 414 \times 3$	$414 \times 414 \times 16$
1	max		$2 \times 2/2$	$414 \times 414 \times 16$	$207 \times 207 \times 16$
2	conv	32	$3 \times 3/1$	$207 \times 207 \times 16$	$207 \times 207 \times 32$
3	max		$2 \times 2/2$	$207 \times 207 \times 32$	$104 \times 104 \times 32$
4	conv	64	$3 \times 3/1$	$104 \times 104 \times 32$	$104 \times 104 \times 64$
5	max		$2 \times 2/2$	$104 \times 104 \times 64$	$52 \times 52 \times 64$
6	conv	128	$3 \times 3/1$	$52 \times 52 \times 64$	$52 \times 52 \times 128$
7	max		$2 \times 2/2$	$52 \times 52 \times 128$	$26 \times 26 \times 128$
8	conv	256	$3 \times 3/1$	$26 \times 26 \times 128$	$26 \times 26 \times 256$
9	max		$2 \times 2/2$	$26 \times 26 \times 256$	$13 \times 13 \times 256$
10	conv	512	$3 \times 3/1$	$13 \times 13 \times 256$	$13 \times 13 \times 512$
11	max		$2 \times 2/1$	$13 \times 13 \times 512$	$13 \times 13 \times 512$
12	conv	1024	$3 \times 3/1$	$13 \times 13 \times 512$	$13 \times 13 \times 1024$
13	conv	256	$1 \times 1/1$	$13 \times 13 \times 1024$	$13 \times 13 \times 256$
14	conv	512	$3 \times 3/1$	$13 \times 13 \times 256$	$13 \times 13 \times 512$
15	conv	18	$1 \times 1/1$	$13 \times 13 \times 512$	$13 \times 13 \times 18$
16	yolo				
17	route	13			
18	conv	128	$1 \times 1/1$	$13 \times 13 \times 256$	$13 \times 13 \times 128$
19	upsample		$2\times$	$13 \times 13 \times 128$	$26 \times 26 \times 128$
20	route	198			
21	conv	256	$3 \times 3/1$	$26 \times 26 \times 384$	$26 \times 26 \times 256$
22	conv	18	$1 \times 1/1$	$26 \times 26 \times 256$	$26 \times 26 \times 18$
23	yolo				

For training the LPD-net, we used a private dataset, which is presented in Sect. 4.2. The four corners of each LP were manually labeled and no data augmentation techniques were required. The training of the network used the follow parameters: 416 × 416 for input size; 50k iterations of mini-batches containing 64 images; learning rate of 0.001 in the first 25k iterations and 0.0001 in the rest of them.

3.3 Optical Character Recognition

Eventually, since the LP image is isolated, the characters can be recognized using an Optical Character Recognition network (OCR-net) [16]. We decided not to train a model from scratch since there are pre-trained weighs with satisfying results [17]. Also, the vast majority of Brazilian LPs have a pattern of three letters, followed by four numbers in a uniform background color. Thus, some heuristics are applied to replace digits and letters when it is needed, as shown in Table 2.

Table 2. Replacing heuristics for correcting the recognized text

Letter		Digit
O	⇔	0
Q	⇒	0
D	⇒	0
I	⇔	1
Z	⇒	2
A	⇔	4
S	⇔	5
G	⇔	6
T	⇔	7
B	⇔	8

4 Materials

This section provides information about the Jetson TX2, and the datasets evaluated in this paper.

4.1 Jetson TX2

Jetson TX2 is a power-efficient computing device. It has a powerful processor that helps to bring artificial intelligence processing power to end products. The Jetson TX2 is composed of a GPU with Nvidia Pascal and a dual-core 64 bit

ARM processor. Furthermore, it has an 8 GB RAM with a speed of 59.7 GB/s. It has standard connections for cameras, displays, mouse, and keyboard, as well as GPIO pins, which allow for fast prototyping.

4.2 Datasets

In this paper, we used two datasets of Brazilian LP images to evaluate the proposed methodology. The first one is a private dataset composed of 1988 images of cars obtained from traffic monitoring cameras from Brazil. The resolution of the images is 752 × 540 and they were captured during the daytime and also during nighttime, resulting in some black and white pictures. This dataset was split into 1331 images for training and 657 for testing. Samples of the private dataset are presented in Fig. 2. Since this is a private dataset, we omitted characteristics from the vehicle that can identify it, including the license plate.

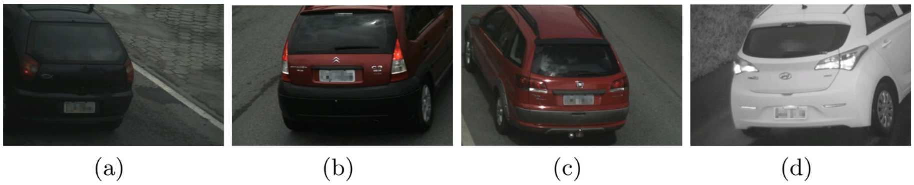
(a) (b) (c) (d)

Fig. 2. Samples of the private dataset of car images

The second dataset, which is called UFPR-ALPR dataset, is from real-world scenarios, where a camera was placed inside a moving vehicle. Three different cameras were used on the images acquisition; for each camera, approximately 1,500 images with 1920 × 1080 pixels of size were captured, totaling 4,500 images, 150 vehicles, and over 30,000 characters. This dataset was split in 40% for training, 40% for testing, and the remaining for validation. The UFPR-ALPR dataset can only be used for academic research. Because of the private dataset has only images of cars, we filtered the UFPR-ALPR validation set in order to have only car images. Samples of the UFPR-ALPR dataset are shown in Fig. 3. We can observe that this dataset has a proper perspective for an embedded system.

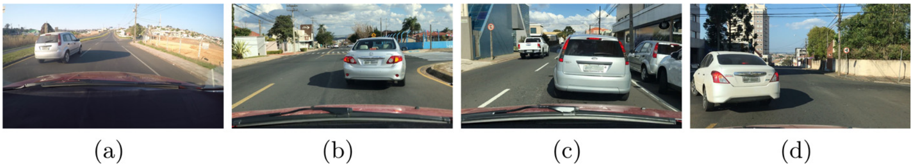
(a) (b) (c) (d)

Fig. 3. Samples of the UFPR-ALPR of car images.

5 Experimental Results

In this section, we evaluate the proposed ALPR system in two steps. The first step consists of experiments using the UFPR-ALPR dataset using a computer composed by Nvidia GTX 1070 as GPU, 8 GB of RAM, with Ubuntu 16.04 LTS as the operating system. In the second step, we evaluated the proposed ALPR system in an embedded platform, which is the Jetson TX2.

Since the LPD-net was trained only with Brazilian plates and cars, we compare our results with the works of Silva and Jung [17] and Laroca et al. [10]. Both papers proposed a complete ALPR system, and used a larger dataset than ours; Silva and Jung [17] affirmed to be tuned for Brazilian plates, and Laroca et al. [10] used Brazilian plates in its plate recognition training.

We performed ten runs on the validation set of each dataset on each system. All runs resulted in the same value. The final results of the proposed system are presented in Table 3 in bold. This table also presents the results for the methods in both datasets that we compare the proposed method.

For the UFPR-ALPR dataset, our system recognized all seven characters in 85.27% of the images, resulting in an improvement of 6.24% when compared to [17], but a reduction of 9.52% when compared to [10]. However, the proposed system identified at least six characters in 96.87% of the LPs, while [10] recognized at least six characters in 97.57% of the images, then the proposed system is just 0.7% inferior.

For the private dataset, we can note that the proposed approach achieved a recognition rate 3.51% inferior than [10] on all characters correct, but obtained a result 2.13% better on at least six characters correct. This result can indicate that the dataset used on the training stage influences the results. In both datasets, the OCR-net could be responsible for the big difference between a completely correct LP and a correct six-characters plate.

In Fig. 4, we can observe that the proposed method surpass the method proposed by [17] in both datasets, and is better than [10] in the private dataset. In addition, even though the proposed method does not have samples of the UFPR-ALPR dataset in its training, it has similar results to the method proposed by [10].

Table 3. Recognition rates of the proposed ALPR system in the Nvidia GTX 1070 and Jetson TX2.

Dataset	ALPR	≥6 characters	All correct
UFPR-ALPR cars	Silva and Jung [17]	87.57%	79.03%
	Laroca et al. [10]	97.57%	94.79%
	Proposed	**96.87%**	**85.27%**
Private dataset	Silva and Jung [17]	86.76%	75.34%
	Laroca et al. [10]	88.43%	81.74%
	Proposed	**90.56%**	**78.23%**

5.1 Evaluation Using Nvidia GTX 1070

In Table 4, we present an average time required for processing the proposed ALPR system in the UFPR-ALPR dataset using an Nvidia GTX 1070 as GPU divided into stages. The LPD-net runs at 105 FPS, making it feasible for embedded systems. The slowest step is the vehicle detection, running at 22 FPS. This speed is due to the MobileNet-SSD with Pascal-VOC weights, which contains 20 classes, then one image can have additional classes, making the time increase.

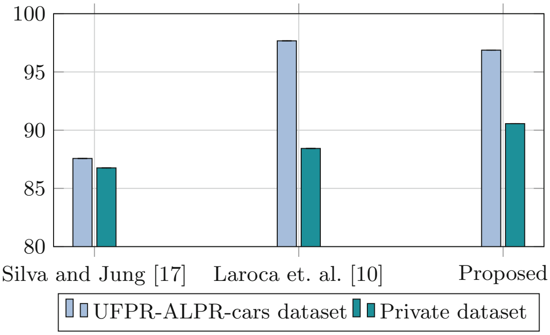

Fig. 4. Graphical representation of the results for at least six characters correct.

Table 5 shows a comparison of the average of the processing times between different ALPR systems using an Nvidia GTX 1070 as GPU. For the UFPR-ALPR-cars dataset, we can observe that [17] achieved only 2 FPS, and [10] reached 5 FPS, meaning that the proposed system is three times faster than the system proposed by [10]. In the private dataset, the proposed system stood out again, reaching about fives faster than the compared approaches. All three systems had a better FPS in the private dataset because it has only one car per image.

Table 4. Results of average time required for processing the ALPR system in **UFPR-ALPR dataset** using a Nvidia GTX 1070.

ALPR Stage	Time (ms)	FPS
Vehicle detection	45.2018	22
LP detection	9.5340	105
Character recognition	10.2778	97
ALPR	65.0137	15

5.2 Evaluation Using Jetson TX2

Since the results on Nvidia GTX 1070 were promising, we decided to embed the system on a Jetson TX2. Table 6 exhibits the average time required for processing the proposed ALPR system in the UFPR-ALPR dataset using a Jetson TX2 divided into stages. We can note that the complete system took approximately 122 ms to execute the three stages per image. Despite the time being double than the Nvidia GTX 1070, this time still is efficient for an embedded platform, considering that we have a complete ALPR-system.

Table 5. Comparison of average processing time between different ALPR systems using a Nvidia GTX 1070.

Dataset	ALPR	Time (ms)	FPS
UFPR-ALPR cars	Silva and Jung [17]	413.9287	2
	Laroca et al. [10]	198.5857	5
	Proposed	65.0137	15
Private dataset	Silva and Jung [17]	79.2450	13
	Laroca et al. [10]	90.9189	11
	Proposed	19.2940	52

Table 7 presents a comparison of the average processing time between different ALPR systems using a Jetson TX2. For the UFPR-ALPR-cars dataset, we can observe that the proposed system is significantly faster than [17] and [10]. For the private dataset, all systems improved their times due to the characteristics of the dataset, highlighting the proposed system that is approximately five times faster than the other approaches.

In Fig. 5, we can observe that the proposed system is faster than the other approaches in both datasets used in the experiments, processing more frames per seconds. Thus, the proposed system is feasible and can be applied as a real-time application in a Jetson TX2.

Table 6. Results of average time required for processing the ALPR system in **UFPR-ALPR dataset** using a Nvidia Jetson TX2.

ALPR stage	Time (ms)	FPS
Vehicle detection	63.3129	16
LP detection	40.9666	24
Character recognition	18.5270	54
ALPR	122.8065	8

Table 7. Comparison of average processing time between different ALPR systems using a Nvidia Jetson TX2

Dataset	ALPR	Time (ms)	FPS
UFPR-ALPR cars	Silva and Jung [17]	755.3228	1
	Laroca et al. [10]	647.4782	1
	Proposed	122.8065	8
Private dataset	Silva and Jung [17]	467.7464	2
	Laroca et al. [10]	436.7330	2
	Proposed	93.9716	11

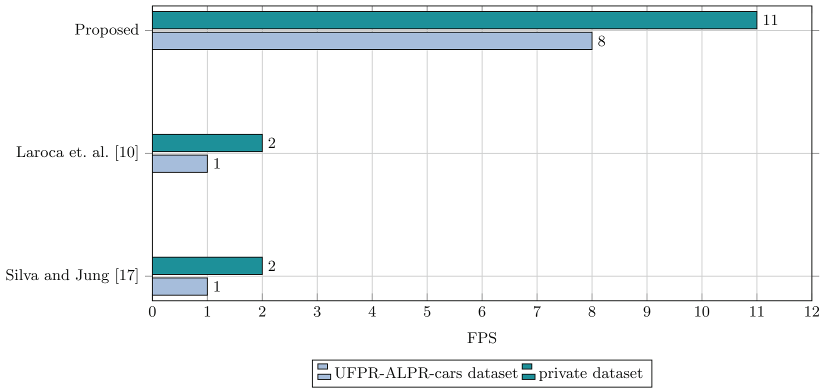

Fig. 5. Chart comparing the average framerate of the ALPR systems in the UFPR-ALPR-cars and private dataset using Jetson TX2.

6 Conclusion and Future Works

In this paper, we proposed a complete ALPR system for Brazilian LPs. We used two existing CNN networks: MobileNet-SSD with Pascal-VOC weights for the detection of the cars, and OCR-net for character recognition. In addition, we created the LPD-net, a CNN network modified from the Yolov3 Tiny for the plate detection.

In order to evaluate the proposed system, we used two datasets: one private dataset and one public dataset, the UFPR-ALPR dataset. Furthermore, we assessed the results in two platforms, Nvidia GTX 1070 and Nvidia Jetson TX2. This comparison was made, mainly because many papers published make the assumption that there is unlimited computing power. However, this is not the case when dealing with mobile or portable systems.

When considering the complete ALPR system and the recognition of at least six characters, the proposed approach achieved 96.87% in the UFPR-ALPR dataset and 90.56% in the private dataset. Besides, the proposed system accomplished the best processing times in both datasets in both platforms, Nvidia GTX

1070 and Nvidia Jetson TX2; in Nvidia GTX 1070, the system obtained 65.01 ms and 19.29 ms for the UFPR-ALPR dataset and the private dataset, respectively. In Jetson TX2, the system reached the times 122.81 ms and 93.97 ms for the UFPR-ALPR dataset and the private dataset, respectively. Thus, these results indicated to be efficient and feasible for embedded systems.

For future works, first, we aim to implement the system as a real-time application inside a car, connecting cameras to the Jetson TX2. Also, we intend to create a CNN network to detect the cars so we can speed up this step and, consequently, the overall process. We also want to expand the proposed system for motorcycles and other types of vehicles.

Acknowledgments. This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001. Also Pedro Pedrosa Rebouças Filho acknowledges the sponsorship from the Brazilian National Council for Research and Development (CNPq) via Grants Nos. 431709/2018-1 and 311973/2018-3. Also, the authors would like to thank The Ceará State Foundation for the Support of Scientific and Technological Development (FUNCAP) for the financial support (6945087/2019).

References

1. Björklund, T., Fiandrotti, A., Annarumma, M., Francini, G., Magli, E.: Robust license plate recognition using neural networks trained on synthetic images. Pattern Recogn. **93**, 134–146 (2019)
2. Bulan, O., Kozitsky, V., Ramesh, P., Shreve, M.: Segmentation-and annotation-free license plate recognition with deep localization and failure identification. IEEE Trans. Intell. Transp. Syst. **18**(9), 2351–2363 (2017)
3. Du, S., Ibrahim, M., Shehata, M., Badawy, W.: Automatic license plate recognition (ALPR): a state-of-the-art review. IEEE Trans. Circuits Syst. Video Technol. **23**(2), 311–325 (2012)
4. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (VOC) challenge. Int. J. Comput. Vision **88**(2), 303–338 (2010)
5. Gonçalves, G.R., Diniz, M.A., Laroca, R., Menotti, D., Schwartz, W.R.: Real-time automatic license plate recognition through deep multi-task networks. In: 2018 31st SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI), pp. 110–117. IEEE (2018)
6. Gonçalves, G.R., Diniz, M.A., Laroca, R., Menotti, D., Schwartz, W.R.: Multi-task learning for low-resolution license plate recognition. In: Nyström, I., Hernández Heredia, Y., Milián Núñez, V. (eds.) CIARP 2019. LNCS, vol. 11896, pp. 251–261. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-33904-3_23
7. Gonçalves, G.R., Menotti, D., Schwartz, W.R.: License plate recognition based on temporal redundancy. In: 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC), pp. 2577–2582. IEEE (2016)
8. Howard, A.G., et al.: MobileNets: efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 (2017)
9. Laroca, R., et al.: A robust real-time automatic license plate recognition based on the YOLO detector. In: 2018 International Joint Conference on Neural Networks (IJCNN), pp. 1–10. IEEE (2018)

10. Laroca, R., Zanlorensi, L.A., Gonçalves, G.R., Todt, E., Schwartz, W.R., Menotti, D.: An efficient and layout-independent automatic license plate recognition system based on the YOLO detector. arXiv preprint arXiv:1909.01754 (2019)
11. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature **521**(7553), 436 (2015)
12. Liu, W., et al.: SSD: single shot multibox detector. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9905, pp. 21–37. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46448-0_2
13. Mittal, S.: A survey on optimized implementation of deep learning models on the NVIDIA Jetson platform. J. Syst. Architect. **97**, 428–442 (2019)
14. Redmon, J., Farhadi, A.: YOLOv3: an incremental improvement. arXiv preprint arXiv:1804.02767 (2018)
15. Sang, J., et al.: An improved YOLOv2 for vehicle detection. Sensors **18**(12), 4272 (2018)
16. Silva, S.M., Jung, C.R.: Real-time Brazilian license plate detection and recognition using deep convolutional neural networks. In: 2017 30th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI), pp. 55–62. IEEE (2017)
17. Silva, S.M., Jung, C.R.: License plate detection and recognition in unconstrained scenarios. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11216, pp. 593–609. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01258-8_36
18. Wang, L., Lu, Y., Wang, H., Zheng, Y., Ye, H., Xue, X.: Evolving boxes for fast vehicle detection. In: 2017 IEEE International Conference on Multimedia and Expo (ICME), pp. 1135–1140. IEEE (2017)