Real-Time Brazilian License Plate Detection and Recognition Using Deep Convolutional Neural Networks

Sérgio Montazzolli
Informatics Institute
Federal University of Rio Grande do Sul
Porto Alegre, Brazil
Email: smsilva@inf.ufrgs.br

Claudio Jung
Informatics Institute
Federal University of Rio Grande do Sul
Porto Alegre, Brazil
Email: crjung@inf.ufrgs.br

***Abstract*—Automatic License Plate Recognition (ALPR) is an important task with many applications in Intelligent Transportation and Surveillance systems. As in other computer vision tasks, Deep Learning (DL) methods have been recently applied in the context of ALPR, focusing on country-specific plates, such as American or European, Chinese, Indian and Korean. However, either they are not a complete DL-ALPR pipeline, or they are commercial and utilize private datasets and lack detailed information. In this work, we proposed an end-to-end DL-ALPR system for Brazilian license plates based on state-of-the-art Convolutional Neural Network architectures. Using a publicly available dataset with Brazilian plates [1], the system was able to correctly detect and recognize all seven characters of a license plate in 63.18% of the test set, and 97.39% when considering at least five correct characters (partial match). Considering the segmentation and recognition of each character individually, we are able to segment 99% of the characters, and correctly recognize 93% of them.**

I. Introduction

Automatic License Plate Recognition (ALPR) is an important task in Intelligent Transportation and Surveillance, which has many practical and relevant applications such as automatic traffic law enforcement, detection of stolen vehicles or toll violation, traffic flow control, etc. It has been studied in the past decades and still is an open problem due to the large variability in image acquisition conditions (illumination, capture angle, distance from camera, etc.) and license plate (LP) layouts, which vary from one country to another.

The ALPR problem can be divided into the following three subtasks: License Plate Detection (LPD), License Plate Segmentation (LPS) and Character Recognition (CR). These subtasks compose the common pipeline for ALPR systems found in the literature [2]–[4], and many works are focused on only one or two of the subtasks [1], [5]–[7]. Furthermore, LPS and CR are highly related to Optical Character Recognition (OCR), which is a noticeably known research field in computer vision and presents several solutions [8]–[11].

Even though a single CR approach could work for a variety of ALPR systems, the same is not true for LPD and LPS. An ideal LPD method should be capable of extracting cars and license plates regardless of the camera setup (static or mobile) and environmental conditions. However, many surveillance systems assume the camera to be static, performing poorly on mobile cameras. The opposite is also true when systems consider the camera to be mobile as they do not take advantage of the small angle and scale variance found in static camera setups. In LPS, the layout and number of characters usually change from one country to another – and sometimes even inside the same country, as in the USA. This variation in conditions and layout motivated the creation of many small license plate databases around the world, such as Greek, Chinese, Indian, American, European, Brazilian. Here we used the Brazilian SSIG Database [1], which consists of 2,000 high definition pictures from 101 different cars taken from a static camera.

With the rise of Deep Learning (DL) techniques [12]–[15], the accuracy of many pattern recognition tasks was greatly improved. In computer vision tasks, it is common to see methods that explore some kind of Convolutional Neural Network achieving state-of-the-art performance. However, the problem with DL techniques is that they need tons of data to be trained from scratch. The more data you have, the deeper your network can be, allowing the recognition of progressively more complex patterns. This is particularly a problem for ALPR systems, since there is no large database available (in particular for Brazilian license plates).

One of our main contributions is the introduction of a complete DL-ALPR system based on state-of-the-art regional CNNs. The system presents two considerably fast and small YOLO-based networks, operating in a cascaded mode, with weights transferred from extensively trained networks. As far as we know, the ALPR system presented here is the first DL-based method focused on Brazilian license plates.

To overcome the lack of annotated car labels in the SSIG database, another contribution is the detection of cars in mostly frontal views without using any external dataset to aid car detection. We simply took advantage of the fully annotated LP regions present in the dataset to estimate good training regions around them, generating a full set of weakly labeled data. Since the estimated regions comprise the car headlights,

2377-5416/17 $31.00 © 2017 IEEE
DOI 10.1109/SIBGRAPI.2017.14

licence plate, and front tires, there is enough information for our Deep CNN to find more than 99% of them on the validation set. As the estimated frontal views are in average $19\times$ larger than license plates, detecting them is easier, faster and less prone to false positives than LP detection itself.

The final important contribution is a carefully designed and trained network to perform character detection and recognition, achieving very good recall and precision rates with an average execution time of 2.2ms in our GPU.

This work is organized as follows: Section II provides a brief description of traditional methods for ALPR in all three subtasks. The proposed system is presented in Section III, detailing the car frontal view detection approach, network architectures, and training procedures. The overall ALPR performance, state-of-the-art comparisons and independent evaluations for each subtask are shown in Section IV. Finally, Section V presents our conclusions and future work plans.

II. Related work

This section revises a few basic concepts about deep learning and it relationship with ALPR. It also presents other computer vision and pattern recognition methods commonly used for ALPR (or its subtasks).

A. Deep Learning (DL)

In 2014, Yosinski et al. [16] studied the transferability of weights from largely trained networks to new networks. For example, the filters found by a large network trained for days on a huge dataset of millions of images, can be transferred to a smaller network and refined for a specific domain. This motivated us to employ DL on ALPR using an open dataset. Furthermore, in the past two years, different regional CNN methods for fast and precise object detection were developed, such as YOLO [14] and Faster-RCNN [13]. However, to the best of our knowledge, only in [2] and [17] these networks were used in the context of ALPR. In the former, the training details and datasets are not available, since it is part of a commercial product; in the latter, they perform only LPD and no detailed information was written in English (the paper is in Korean).

B. Automatic License Plate Recognition (ALPR)

Most of ALPR-related methods tackle only a subset of the required tasks: LPD, which aims to detect the region that contains the plate; LPS, which focuses on segmenting individual characters within a detected plate; and CR, which recognizes each segmented character and provides the final plate string. Some methods that tackle these subtasks are revised next.

License Plate Detection (LPD). Detecting the car before the plate is a common strategy in the LPD pipeline, aiming to reduce the search region and the number of false positives. To perform these detections, approaches using Histogram of Oriented Gradients (HOG) as feature descriptors and Support Vector Machine (SVM) for classification are popular [5], [6], [18]. Another feature descriptor successfully used in this context is Local Binary Patterns, employed by OpenALPR [19] to detect European and USA license plates. In [2], the authors used three deep CNNs to perform LPD, LPS and CR, respectively. Their results outperform OpenALPR by a substantial margin. However, this is a commercial method and no detailed description about the network architecture and training procedures were given. Moreover, they used large private databases to train the networks, while OpenALPR is a totally open source project. Having such databases is a considerable advantage because DL performance is directly connected to the amount of data used.

License Plate Segmentation (LPS). Individual character segmentation is usually performed by some kind of binarization method in order to separate the foreground from the background, following the application of connected components analysis to extract the bounding boxes [20]. This is highly dependent of the chosen binarization method and non-uniformity of the plate (e.g. due to textured background, low contrast between the background and the characters, or non-even illumination). Furthermore, those methods tend to fail (or require additional post-processing) when two characters are connected or one character is broken or divided. In [21], the authors extract the Extremal Regions [8], which is a known technique for text segmentation [9], [10].

Character Recognition (CR). It is a crucial task in ALPR systems, since a single error can invalidate an entire LP. Moreover, some characters have a similar shape (e.g. O/0, I/1, A/4, K/X, C/G, D/O, 2/Z, 5/S, 8/B) and may be even more similar when distorted [4], [20]. Thus, having a CR method with high accuracy is mandatory. In [18], the authors used one SVM per character, leading to a total of 36 SVMs. Gou et al. [21] used HoG features with Restricted Boltzmann Machines (RBM), which is similar to Deep Belief Networks [12], to recognize Chinese characters – which generally are more complex than Roman characters. They argue that RBM is more reliable then SVM for this task. Deep CNNs were used for character recognition in [3] and [2]. In [3], the authors used a 16-layer CNN based on Spatial Transformer Networks [22], aiming to make CR less sensitive to spatial transformations.

In general, we noted that the use of DL in ALPR systems is still limited, and just a few works in fact explore it in an end-to-end fashion. Most of the methods found in the literature rely on hand-engineered features, like HoG and LBP. The data-driven features provided by DL techniques are generally more discriminative and can be coupled with classification in a single network. Furthermore, recent networks like Faster R-CNN and YOLO are capable of performing detection without the need for pyramids of images and features to test multiple scales, achieving high accuracy rates at limited execution times.

III. The Proposed System

As mentioned earlier, the traditional ALPR pipeline typically involves car detection. Although it might be an optional task for some methods, it is essential in our approach: all images in the dataset have high resolution (1920×1080 pixels)

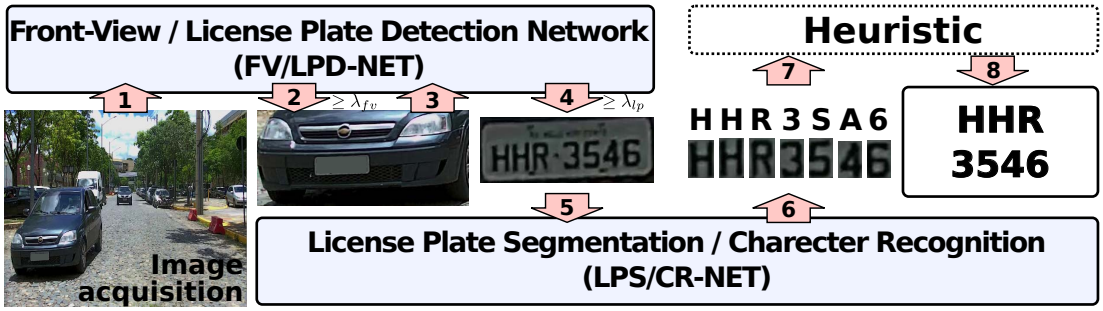

Fig. 1. Proposed System pipeline.

and the cars do not occupy a large portion in the scene (they might be relatively far from the camera). Moreover, the average size of LPs is 0.26% of the full image area. Thus, the computation resources demanded to analyze an image of that size looking for such a small object is considerably high.

Taking into account that we are using DL techniques known to be computationally expensive, avoiding small objects becomes even more important when one seeks for a real-time application. Shrinking the image is not an alternative because the plate may become excessively small to the point of being undetectable. Considering the fact that the available database does not provide annotated car labels or bounding boxes, detecting cars using only its training information turns out to be a challenging task.

We propose a simple method to extract the frontal portion of the cars only based on the annotated LP. Basically, assuming the LP is always attached to a car (and roughly aligned horizontally), we decided to use the region around it to train a detection network, as described in Section III-A.

One of our biggest concerns was to come out with a real-time ALPR system, without the need for expensive hardware. Therefore, we need to use a deep network capable of performing detection and recognition in a very short time. In order to accomplish those objectives, we choose to use a YOLO-based network [14]. As far as we know, these are the fastest networks reported in the literature – even faster than Faster-RCNN [13] that perform object detection/recognition using region proposals, without the need of (costly) image pyramid and sliding windows.

In our tests, the FAST-YOLO network was capable of performing a 20-class detection and classification in 800×600 image in less than 5.5ms[1], or around 180 frames per second (FPS). The precision reported for this network was 57.1% on VOC [23] database, which is not much when compared to other deeper networks like YOLO (76.8%) and Faster-RCNN (76.4%). However, YOLO has $3\times$ more parameters than FAST-YOLO, which makes it slower and data-hungry for training.

As our intention is to tackle a two-class problem (car frontal views and plates) instead of twenty, we hypothesized that refining the FAST-YOLO network would be enough. Its architecture and the training procedure adopted are described in Section III-B.

For the detection and recognition of characters, we built a new network inspired on the YOLO architecture, with fundamental technical differences to accommodate 35 classes (0-9, A-Z except the letter O, which is detected jointly with the number 0) and outputs a feature map having the same aspect ratio of LP (width is three times larger than height). Please refer to Section III-C for more details.

Finally, we perform a post-processing step in order to swap highly confused letters to digits and digits to letters, since the Brazilian license plate is formed by exactly three letters followed by four digits.

A. Frontal-View Extraction

One problem with DL techniques is that they struggle to detect small objects [24], and this indeed happened in our experiments. We trained a FAST-YOLO network to detect only LPs, and the result was a low 82% of recall (which tends to be further reduced after CR).

The YOLO and FAST-YOLO networks have fixed input image dimensions, which in both cases are defined as 416×416. Since in SSIG Database all images have 1920×1080 pixels, a shrinking step is required. When the car is sufficiently far from the camera, that step reduces the LP size to the point of being undetectable.

In order to overcome this problem, we extracted larger regions around the LP, namely car Frontal-Views (FV), by scaling and translating the annotated LP bounding box. The parameters for this spatial transformation were estimated using the following procedure:

1) Randomly selected a set of N_{lp} images from the training dataset, excluding images from buses and trucks because their fronts are too large;
2) Manually labeled these images with a rectangular bounding box around the license plate, where each bounding box have the smallest region possible comprising the car headlights and tires (refer to the yellow dashed rectangle in Fig. 2);
3) Considering a 4-dimensional vector $\vec{p}$ containing the top-left point (p_x, p_y), width p_w and height p_h of the

[1]Hardware used: Intel Xeon E5620 2.4GHz processor, 12GB of RAM memory, and NVIDIA TITAN X GPU.

annotated LP, we want to find two translation parameters (α_x, α_y), and two scaling parameters α_w and α_h that relate the LP bounding box $\vec{p}$ with the FV bounding box $\vec{v}$. This transformation is described by a function

$$\vec{v} = \vec{f}(\vec{p}, \vec{\alpha}) = \begin{bmatrix} p_x + \alpha_x p_w \\ p_y + \alpha_y p_h \\ \alpha_w p_w \\ \alpha_h p_h \end{bmatrix} \quad (1)$$

where $\vec{\alpha}$ contains the scaling and translating parameters.

4) To find $\vec{\alpha}$, we maximized the sum of the Intersection over Union (also known as Jaccard Similarity) over the N_{lp} labeled images:

$$\vec{\alpha}_* = \underset{\vec{\alpha}}{argmax} \sum_{i=1}^{N_{lp}} IoU\left(f(\vec{p}_i, \vec{\alpha}), \vec{v}_i\right). \quad (2)$$

5) With the estimated parameter $\vec{\alpha}_*$, the FV bounding boxes for all remaining images in the dataset were labeled automatically, generating a weakly (but large) labeled set. An example of the weakly labeling process is illustrated in Fig. 2.

Despite discarding buses and trucks images in the training to estimate the transformation parameters, the network where this labels were used was still capable of detecting these kind of vehicles.

Related approaches can be found in [25], [26], where plate regions where used to estimate car frontal-views in order to recognize their model/manufacturer.

Fig. 2. Frontal-View illustration: The dashed yellow rectangle represents a manually annotated FV, and the red solid rectangle is the weakly labeled FV based on the LP (in blue).

B. Frontal-View and License Plate Detection CNN

In this work, the car FV and its LP are detected using a single classifier arranged in a cascaded manner (see Fig. 1): the first layer (arrows 1 and 2) detects the FV from the input image, and the second layer (arrows 3 and 4) extracts the LP from the detected FV image.

To achieve a good compromise between accuracy rates and running times, our classifier is based on the FAST-YOLO network architecture. This network was built (and trained) to handle 20 different classes of objects, and runs at 200 FPS on a good GPU. Thus, we hypothesized that the FAST-YOLO network customized for 2 classes could have the capacity to accommodate both tasks in a single network when executed in a cascaded way.

The FAST-YOLO architecture used is shown in Table I. The only modification to the original was made in layer 15, where we reduced the number of filters from 125 to 35 in order to output 2 classes instead of 20 (please refer to [14] for more information about the detection layer).

TABLE I
FRONTAL-VIEW AND LICENSE PLATE DETECTION NETWORK (FV/LPD-NET): BASICALLY A FAST-YOLO NETWORK ADJUSTED TO OUTPUT 2 OBJECT CLASSES.

nº	Layer	Filters	Size	Input	Output
1	conv	16	3 × 3 / 1	416 × 416 × 3	416 × 416 × 16
2	max		2 × 2 / 2	416 × 416 × 16	208 × 208 × 16
3	conv	32	3 × 3 / 1	208 × 208 × 16	208 × 208 × 32
4	max		2 × 2 / 2	208 × 208 × 32	104 × 104 × 32
5	conv	64	3 × 3 / 1	104 × 104 × 32	104 × 104 × 64
6	max		2 × 2 / 2	104 × 104 × 64	52 × 52 × 64
7	conv	128	3 × 3 / 1	52 × 52 × 64	52 × 52 × 128
8	max		2 × 2 / 2	52 × 52 × 128	26 × 26 × 128
9	conv	256	3 × 3 / 1	26 × 26 × 128	26 × 26 × 256
10	max		2 × 2 / 2	26 × 26 × 256	13 × 13 × 256
11	conv	512	3 × 3 / 1	13 × 13 × 256	13 × 13 × 512
12	max		2 × 2 / 1	13 × 13 × 512	13 × 13 × 512
13	conv	1024	3 × 3 / 1	13 × 13 × 512	13 × 13 × 1024
14	conv	1024	3 × 3 / 1	13 × 13 × 1024	13 × 13 × 1024
15	conv	35	1 × 1 / 1	13 × 13 × 1024	13 × 13 × 35
16	detection				

Training: first, the weights from layers 1 to 14 were transfered from a pre-trained network[2]. Then, we refined the weights by adding samples of annotated FV and LP images. More precisely, training data is provided as a combination of two subsets: (i) full resolution images (1920×1080) with only annotated FVs, obtained with the weak labeling process described in Section III-A; and (ii) cropped regions depicting only the FVs, with their respective annotated LP provided in the SSIG database. Examples of input images used to train the proposed FV/LPD-NET are illustrated in Fig. 3. We did not mix validation, training and test sets in order to create set (ii).

License Plate Detection: we run the same network two times, or one for each of the first two layers in the cascade. The process is as follows:

- The first pass involves the whole image and looks only for FVs. Any LP found is discarded;
- The detected FVs are then cropped and fed to the same network, and only the output related to LP is used. If multiple LP are found, only the one with the highest probability is kept, since each vehicle is expected to present a single LP.

[2]Pre-trained YOLO models for Darknet framework are available at https://pjreddie.com/darknet/yolo/.

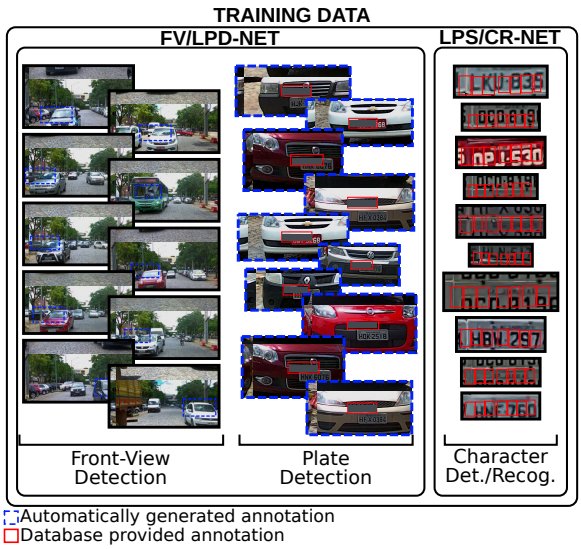

Fig. 3. Examples of labeled images used to train our two CNNs.

Note that if we have N license plates in the scene, the network is expected to run $N + 1$ times: one to detect the N FVs, and N to detect LPs (one for each FV). However, in several ALPR applications, such as parking or toll monitoring, there is only one vehicle at a time.

C. Character detection and recognition

The National Traffic Council (CONTRAN) in Brazil dictates how the LP format must be all over the country. Their specifications have very few exceptions (e.g. presidential and consular cars) that do not follow the 7 characters format. The vast majority of LPs share common features: uniform background color, fixed font size and LP dimensions, and 7 characters divided into two groups: a first group of 3 letters and a second group of 4 digits, as illustrated in Fig. 4. This is a nice characteristic for pattern recognition, since it has less variability than license plates found in other countries, making the character detection and recognition problems slightly easier. However, the Roman numerals and letters have some similarities between many characters, for instance, the pairs O/Q, 0/D, 1/I, 5/S, 2/Z, B/8, C/G, A/4 are often confused due to some distortion or occlusion [20]. In OCR applications for text, many of these confusions can be fixed by using adjacency information and semantic analysis, assuming that detected characters form valid words. This is harder for LPR, since semantic information is inexistent.

Fig. 4. Brazilian LP layout: 3 letters followed by 4 digits.

The YOLO and FAST-YOLO networks have fixed input and output aspect ratio and granularity. Their aspect ratio is $1:1$, and yet they both present a good detection performance on portrait and landscape images, as demonstrated in [14]. However the Brazilian LPs present an aspect ratio of approximately $3:1$, being too wider for these networks as our tests indicated. Therefore, we changed the network parameters in order to match the LPs aspect ratio.

Another difference is related to the network input and output granularity. We noted that the 13×13 YOLO output was not dense enough to capture seven big objects (characters) horizontally spread side by side. In order to amend this issue, we almost triplicated the horizontal granularity, letting the final network output as 30×10. This slight decrease in vertical size output did not affect the network performance, since the LPs have no vertical sequence of objects to detect.

For the input, we chose to use 240×80 images, which is approximately two times the average plate size on the database, and reduces the chance of losing important information. Just for the record, we also tried different input and output sizes, such as $144 \times 48 \rightarrow 18 \times 6$, $192 \times 64 \rightarrow 24 \times 8$ and $288 \times 96 \rightarrow 36 \times 12$. The first 2 networks performed worse than the chosen one, and the later performed equally but is more complex.

Using a smaller input than the original proposal (240×80 against 416×416) implied in some architecture modifications. First we need to cut down the number of max pooling layers from five to three, in order to keep the fine output granularity by avoiding many dimensionality reductions. Second, to maintain the network depth equals to FAST-YOLO and yet being allowed to use as much transfer learning as possible, we used the first eleven layers of YOLO network, stopping on the twelfth layer, since it contains the fourth max pooling in that network. If we applied the same idea of capturing all layers before the fourth max pooling to FAST-YOLO, we would end up using just seven layers, reducing the network depth. Lastly, four more layers were added and trained from scratch to improve non-linearity.

The final architecture of the proposed network is provided in Table II. The training procedure was similar to FV/LPD-NET presented on Section III-B, and some training samples are shown on the right of Fig. 3.

D. Heuristics

Based on the fact that Brazilian LP as formed by three letters followed by four numbers, we used two heuristic rules to filter the results produced by the LPS/CR-NET: i) if more than seven characters are detected, only the seven most probable are kept. ii) the first three characters are assumed to be letters, and the following four digits. This assumption is used to swap letters by numbers and vice-versa, depending on the character position. In summary, if a letter is recognized in the LP block related to digits, it is swapped by the digit that presented the largest occurrence in the confusion matrix obtained with the training data. A similar process is applied when a digit is

TABLE II
LICENSE PLATE SEGMENTATION AND CHARACTER RECOGNITION NETWORK (LPS/CR-NET): ALL LAYERS FROM 1 TO 11 WERE TRANSFERED FROM YOLO-VOC NETWORK. THE INPUT IMAGE IS A 240×80 COLORED LP PATCH, AND THE 30×10 OUTPUT ENSURES ENOUGH HORIZONTAL GRANULARITY FOR THE 7 CHARACTERS.

nº	Layer	Filters	Size	Input	Output
1	conv	32	3 × 3 / 1	240 × 80 × 3	240 × 80 × 32
2	max		2 × 2 / 2	240 × 80 × 32	120 × 40 × 32
3	conv	64	3 × 3 / 1	120 × 40 × 32	120 × 40 × 64
4	max		2 × 2 / 2	120 × 40 × 64	60 × 20 × 64
5	conv	128	3 × 3 / 1	60 × 20 × 64	60 × 20 × 128
6	conv	64	1 × 1 / 1	60 × 20 × 128	60 × 20 × 64
7	conv	128	3 × 3 / 1	60 × 20 × 64	60 × 20 × 128
8	max		2 × 2 / 2	60 × 20 × 128	30 × 10 × 128
9	conv	256	3 × 3 / 1	30 × 10 × 128	30 × 10 × 256
10	conv	128	1 × 1 / 1	30 × 10 × 256	30 × 10 × 128
11	conv	256	3 × 3 / 1	30 × 10 × 128	30 × 10 × 256
12	conv	512	3 × 3 / 1	30 × 10 × 256	30 × 10 × 512
13	conv	256	1 × 1 / 1	30 × 10 × 512	30 × 10 × 256
14	conv	512	3 × 3 / 1	30 × 10 × 256	30 × 10 × 512
15	conv	80	1 × 1 / 1	30 × 10 × 512	30 × 10 × 80
16	detection				

recognized in the LP block related to letters. The specific swaps are given by

- Swap rules for the first 3 positions (letters): $5 \Rightarrow S$, $7 \Rightarrow Z$, $1 \Rightarrow I$, $8 \Rightarrow B$, $2 \Rightarrow Z$, $4 \Rightarrow A$ and $6 \Rightarrow G$;
- Swap rules for the last 4 positions (numbers): $Q \Rightarrow 0$, $D \Rightarrow 0$, $Z \Rightarrow 7$, $S \Rightarrow 5$, $J \Rightarrow 1$, $I \Rightarrow 1$, $A \Rightarrow 4$ and $B \Rightarrow 8$.

This last heuristic is only applied when the network outputs exactly seven characters. Otherwise, the LP is wrong anyway and correctly recognized characters can be swapped, worsening the result.

IV. EXPERIMENTAL RESULTS

In this section we separate the evaluation into two steps. First, we performed individual assessments in each ALPR subtask to verify their performance and set values for fixed parameters. Second, we evaluated the full ALPR system and compared to a state-of-the-art system.

To train our LPD network, the following parametrization were used: 10k iterations on Stochastic Gradient Descent algorithm, with mini-batch size of 64, and learning rate of 10^{-3} for the first $1k$ iterations, and 10^{-4} after them. The validation set was used to select thresholds to filter the outputs. Basically, all segments found by adopting a threshold λ, were submitted to a Non-Maximal Suppression algorithm in order to filter out overlapped detections.

In Fig. 5 we show the LPD precision and recall curves considering different IoU acceptance values (from 0.4 to 0.7). The curves were extracted using a fixed FV threshold $\lambda_{fv} = 0.2$, and varying the LP threshold λ_{pl} uniformly in the range $[0, 1]$. As can be observed, our method is not much sensitive for threshold variations, except for very high or very low values. In fact, if λ_{pl} assumes any value between 0.1 and 0.5, there is no significant difference ($\rho = 0.05$) in the results.

It can also be observed that our method achieved very high recall and precision rates for IoU up to 0.6. For instance, considering an IoU threshold of 0.5 (that is used in similar problems, such as pedestrian detection [27]), it reached 99.51% recall and 95.07% precision rates in the validation set, considering $\lambda_{fv} = 0.2$ and $\lambda_{pl} = 0.5$. When analyzing the precision errors, we noted an issue in the test database named *Track101*. There is a sequence of 15 frames in that test (almost 4% of the validation set) with a second car totally visible in the scene, including its license plate, but without any annotation. This represents the majority part of the precision error.

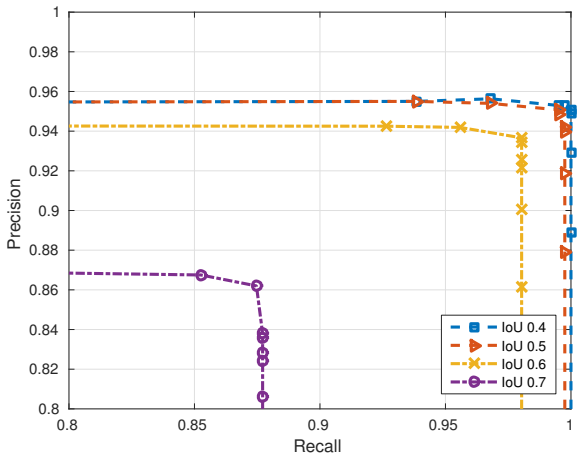

Fig. 5. License Plate Detection: precision and recall curves for IoUs ranging from 0.4 to 0.7.

For the license plate segmentation and character recognition network, no parameter tunning was needed since, according to the heuristic described in Section III-D, only the seven most probable detections will remain. In this way, our network achieved more than 99% of recall and precision (considering and IoU of 0.5) to segment the characters, with just a few position mistakes. We compared our LPS to the method presented in [18][3], and the F-measure (which combines precision and recall) for different IoU acceptance thresholds are shown in Fig. 6. In particular, for an IoU of 0.5, our network achieved and F-measure of 99.82% against 41.96% of [18], showing that it successfully performed segmentation on the majority of the characters.

Regarding the proposed full ALPR pipeline, we compared our results with Sighthound[4] [2]. On one hand, our method was tuned specifically for Brazilian plates, while Sighthound was devised for American and European license plates. On the

[3]Their work was tested on a newer and larger database which is not publicly available yet, but the authors kindly provided their results using the SSIG Database.

[4]State-of-the-art commercial system, which has a cloud service available at https://www.sighthound.com/products/cloud.. The results presented here were collected on June, 2017.

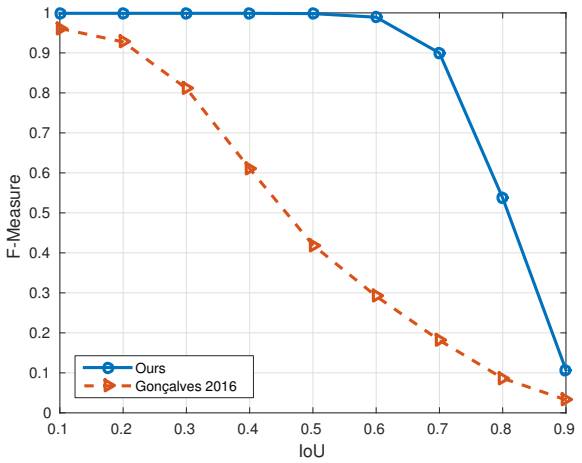

Fig. 6. Character Detection performance: F-Measure comparison with the segmentation method presented in [18].

other, Sighthound was trained using much larger and private datasets than ours, which is very important for DL techniques.

The final results are presented in Table III. Our system successfully recognized 63.18% of the license plates on the test set, resulting in a relative improvement of 13.9% when compared to Sighthound. Also, it was capable of correctly identifying at least 5 characters in 97.39% of the LPs, which might be helpful to prune candidate plates based on additional information, such as vehicle manufacturer/model. Some examples of ALPR results produced by our method are presented in Figure 7.

TABLE III
FINAL OUTPUT EVALUATION. ACCURACY CONSIDERING: FULLY CORRECTED LPS, AT LEAST 6 CORRECT CHARACTERS AND AT LEAST 5 CORRECT CHARACTERS.

ALPR	All correct	$\geq$ 6-characters	$\geq$ 5-characters
Sighthound	55.47%	80.47%	87.94%
Ours	**63.18%**	**90.55%**	**97.39%**
Ours (only letters)	63.43%	–	–
Ours (only numbers)	93.03%	–	–

Another important consideration is regarding the recognition of numbers and letters, which is shown in the second half of Table III. Our system was able to correctly recover all four the numbers in a LP in 93.03% of the test set cases. However, only in 63.43% all three letters were correctly recognized. Hence, the main accuracy bottleneck of the proposed approach is letter recognition, which might be explained by the database unbalance between characters: there are a lot more number samples than letter samples. Moreover, in letter classes, there are some characters that appear much more often than others due the country zone where the database were recorded. For example, the least frequent digit is '2', with a little more than 200 samples, while the least frequent letter is 'V', with 16 samples.

Our final evaluation is related to execution time. In Table IV we show the average time needed for each network to process an input, and the total time for the whole system (assuming that a single vehicle is being processed). Our ALPR system runs at 76 FPS using a high-end GPU, and it can achieve around 9 FPS with a cheaper mid-end GPU, which is also feasible in several applications such as parking and toll monitoring systems.

TABLE IV
EXECUTION TIME: NETWORKS FEED FORWARD TIMES FOR HIGH AND MID-END GPUS.

Network	Time (FPS) High-end GPU (NVIDIA TITAN X)	Mid-end GPU (GeForce GT 750M)
FV/LPD-NET	5.4ms (185)	47.2ms (21)
LPS/CR-NET	2.2ms (448)	20.1ms (47)
Total (2× FV/LPD-NET + LPS/CR-NET)	**13.0ms (76)**	**114.5ms (9)**

V. CONCLUSION

In this paper we proposed an end-to-end ALPR system for Brazilian license plates based on Deep Learning. Two YOLO-based CNN networks were created: the first (FV/LPD-NET) detects car frontal-views and license plates operating in a cascaded mode, and the second (LPS/CR-NET) detects and recognizes characters within a cropped license plate.

To overcome the lack of car bounding boxes in the available database, a simple method to generate a large set of weakly annotated car bounding boxes from a few manually annotated ones was also introduced in the paper.

The FV/LPD-NET, which is an adaptation of the FAST-YOLO network, achieved 99.51% of recall coupled with a high precision (95.07%) in the LPD task, given an IoU threshold of 0.5. For LPS, we proposed a new network architecture designed to be fast and precise. This network (LPS/CR-NET) shares similar aspect ratio to LPs, and is as deep as FAST-YOLO, but faster to execute. It was able to recover 99.84% of the characters considering an IoU of 0.5, outperforming the baseline by a large margin.

Considering the full ALPR system running end-to-end, 63.18% of the LPs on the database test set were correctly recognized (all seven characters), even with a highly unbalanced training database of characters (in particular, letters). When considering partial matches (at least five characters correctly recognized in a given LP), the proposed method presented an accuracy of 97.39%.

For future work, we intend to improve our ALPR system by firstly using a more solid method for character recognition. A good recognizer for letters can significantly boost the final accuracy. We also want to try different CNN models like Spatial Transforming Networks [22] and Parts-Based Networks [28], [29]. Both can be applied to LPS, being the former particularly interesting because it can capture rotated patterns, potentially improving generalization. The later can be used to detect

HJN2081 PUG7401 00J2834 HIF8352 00K5510 GMF2862 HAY5852

Fig. 7. Examples of ALPR produced by the proposed method.

objects inside objects, for instance a LP attached to a car or a character in the LP, or to explore the expected spatial distribution of characters within a plate for LPS.

Finally, we want to include car manufacturer/model recognition in the ALPR pipeline. For this, the above mentioned parts-based models can be of great help.

ACKNOWLEDGMENT

The authors would like to thank the funding agencies CAPES and CNPq, as well as NVIDIA Corporation for the donation of the Titan X Pascal GPU used for this research.

REFERENCES

[1] G. R. Gonçalves, S. P. G. da Silva, D. Menotti, and W. R. Schwartz, "Benchmark for license plate character segmentation," *Journal of Electronic Imaging*, vol. 25, no. 5, pp. 1–5, 2016.

[2] S. Z. Masood, G. Shu, A. Dehghan, and E. G. Ortiz, "License Plate Detection and Recognition Using Deeply Learned Convolutional Neural Networks," 2017. [Online]. Available: http://arxiv.org/abs/1703.07330

[3] V. Jain, Z. Sasindran, A. Rajagopal, S. Biswas, H. S. Bharadwaj, and K. R. Ramakrishnan, "Deep automatic license plate recognition system," in *Proceedings of the Tenth Indian Conference on Computer Vision, Graphics and Image Processing - ICVGIP '16*. New York, New York, USA: ACM Press, 2016, pp. 1–8.

[4] C.-N. E. Anagnostopoulos, "License Plate Recognition: A Brief Tutorial," *IEEE Intelligent Transportation Systems Magazine*, vol. 6, no. 1, pp. 59–67, 2014.

[5] J. W. Hsieh, L. C. Chen, and D. Y. Chen, "Symmetrical SURF and Its applications to vehicle detection and vehicle make and model recognition," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 1, pp. 6–20, 2014.

[6] P. R.F, C.-C. G, W. R. Schwartz, and M. D, "Brazilian License Plate Detection Using Histogram of Oriented Gradients and Sliding Windows," *International Journal of Computer Science and Information Technology*, vol. 5, no. 6, pp. 39–52, 2013.

[7] Y. Yuan, W. Zou, Y. Zhao, X. Wang, X. Hu, and N. Komodakis, "A Robust and Efficient Approach to License Plate Detection," *IEEE Transactions on Image Processing*, vol. 26, no. 3, pp. 1102–1114, mar 2017.

[8] J. Matas, O. Chum, M. Urban, and T. Pajdla, "Robust Wide Baseline Stereo from Maximally Stable Extremal Regions," in *Procedings of the British Machine Vision Conference 2002*. British Machine Vision Association, 2002, pp. 36.1–36.10.

[9] H. Cho, M. Sung, and B. Jun, "Canny Text Detector: Fast and Robust Scene Text Localization Algorithm," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, jun 2016, pp. 3566–3573.

[10] M.-C. Sung, B. Jun, H. Cho, and D. Kim, "Scene text detection with robust character candidate extraction method," in *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*. IEEE, aug 2015, pp. 426–430.

[11] Q. Ye and D. Doermann, "Text detection and recognition in imagery: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 7, pp. 1480–1500, July 2015.

[12] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A Fast Learning Algorithm for Deep Belief Nets," *Neural Computation*, no. 7, pp. 1527–1554, jul 2006.

[13] 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, jun 2017.

[14] 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)*. IEEE, jun 2016, pp. 779–788.

[15] 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)*, vol. 4, no. 3. IEEE, jun 2016, pp. 770–778.

[16] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Proceedings of the 27th International Conference on Neural Information Processing Systems*, ser. NIPS'14. Cambridge, MA, USA: MIT Press, 2014, pp. 3320–3328.

[17] D. Lee, S. Yoon, J. Lee, and D. S. Park, "Real-Time License Plate Detection Based on Faster R-CNN," *KIPS Transactions on Software and Data Engineering*, vol. 5, no. 11, pp. 511–520, nov 2016.

[18] G. R. Gonçalves, D. Menotti, and W. R. Schwartz, "License plate recognition based on temporal redundancy," in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, Nov 2016, pp. 2577–2582.

[19] M. Hill, "OpenALPR," https://github.com/openalpr/openalpr.

[20] 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, no. 2, pp. 311–325, Feb 2013.

[21] C. Gou, K. Wang, Y. Yao, and Z. Li, "Vehicle License Plate Recognition Based on Extremal Regions and Restricted Boltzmann Machines," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 4, pp. 1096–1107, apr 2016.

[22] M. Jaderberg, K. Simonyan, A. Zisserman, and k. kavukcuoglu, "Spatial transformer networks," in *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds. Curran Associates, Inc., 2015, pp. 2017–2025. [Online]. Available: http://papers.nips.cc/paper/5854-spatial-transformer-networks.pdf

[23] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, Jun. 2010.

[24] Z. Zhu, D. Liang, S. Zhang, X. Huang, B. Li, and S. Hu, "Traffic-sign detection and classification in the wild," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016, pp. 2110–2118.

[25] A. Psyllos, C. Anagnostopoulos, and E. Kayafas, "Vehicle model recognition from frontal view image measurements," *Computer Standards & Interfaces*, vol. 33, no. 2, pp. 142–151, feb 2011.

[26] H. Yang, L. Zhai, Z. Liu, L. Li, Y. Luo, Y. Wang, H. Lai, and M. Guan, "An Efficient Method for Vehicle Model Identification via Logo Recognition," in *2013 International Conference on Computational and Information Sciences*. IEEE, jun 2013, pp. 1080–1083.

[27] M. Enzweiler and D. M. Gavrila, "Monocular pedestrian detection: Survey and experiments," *IEEE transactions on pattern analysis and machine intelligence*, vol. 31, no. 12, pp. 2179–2195, 2009.

[28] N. Zhang, J. Donahue, R. Girshick, and T. Darrell, "Part-based r-cnns for fine-grained category detection," in *European conference on computer vision*, 2014, pp. 834–849.

[29] H. Zhang, T. Xu, M. Elhoseiny, X. Huang, S. Zhang, A. Elgammal, and D. Metaxas, "SPDA-CNN: Unifying Semantic Part Detection and Abstraction for Fine-Grained Recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, jun 2016, pp. 1143–1152.