An ALPR System-based Deep Networks for the Detection and Recognition

Mouad Bensouilah, Mohamed Nadjib Zennir and Mokhtar Taffar
Department of Computer Science, Jijel University, BP 98, Ouled Aissa, 18000, Jijel, Algeria

Keywords: Computer Vision, Convolutional Neural Networks, Recurrent Neural Networks, GRU, LSTM, Object Detection and Recognition, OCR, License Plates Recognition.

Abstract: Automatic license plates reading (ALPR), from images or videos, is a research topic that is still relevant in the field of computer vision. In this article, we propose a new dataset and a robust ALPR system based on the YOLO object detector of literature. The trained Convolutional Neural Networks (CNN) allow us to extract features from license plates and label them through Recurrent Neural Networks (RNN) specialized character recognition. RNN are supported by GRU units instead of LSTM units that are generally used in the literature. The experiments results were conclusive reaching a recognition rate of 92%.

1 INTRODUCTION

In recent years, deep learning techniques have achieved good performances in the computer vision field particularly for tasks such as detecting objects and recognizing their class by offering different deep network models. These techniques have paved ways and allowed researchers to use powerful deep learning models to develop more performant algorithms and real systems like these used in the field of license plate recognition.

Automatic license plate reading (ALPR) is a common task these days. It is used for many applications such as transport, road safety, parking, etc. A large number of ALPR approaches (Liu et al., 2011)(Du et al., 2012)(Khan et al., 2017)(Cheang et al., 2017)(Laroca et al., 2018) have been proposed in the literature, some of which have been marketed on real world systems.

Most existing algorithms need segmentation as a pre-treatment for plate and/or character detection. Unfortunately, there is no robust segmentation technique for the variety of constraints to which fonts are subjected and which cause great variability in the appearance of characters such as: rotations, scale changes, blurring, lighting variations and noise (Špaňhel et al., 2017).

In this article we propose a recent approach as an alternative for the recognition of license plates without segmentation. It is based on the YOLO object detector (Redmon et al., 2016)(Redmon and Farhadi, 2017)(Redmon and Farhadi, 2018) and Recurrent Convolutional Neural Networks (CRNN)(Shi et al., 2016).

YOLO (Redmon et al., 2016), based on CNN, is an object detection algorithm, fast, and effective, and easy to integrate for real applications, with three versions of YOLO, YOLOv2 (Redmon and Farhadi, 2017) and YOLOv3 (Redmon and Farhadi, 2018). YOLOv3 is the last version which was optimized from the previous, with accuracy improved while maintaining speed and performance. We trained YOLOv3 with two backbones: Darknet-53 (Redmon and Farhadi, 2018) and MobileNets (Andrew G. Howard, 2017); on the ImageNet pre-training model. The test results show that the trained YOLOv3 have extremely high recall and precision.

CRNN proposed by (Shi et al., 2016) is a general framework for character recognition. We applied an improved CRNN based on CNN for extracting feature sequences of license plate images, and we used a 2-layer bidirectional Gated Recurrent Unit (GRU) (Cho et al., 2014) for labeling sequences. The original CRNN uses for labeling sequences a popular variant which is Long Short Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) having more complex structure than GRU. We used Connectionist Temporal Classifier (CTC) loss function proposed by (Graves et al., 2006) during training process. Test results show that our CRNN have high accuracy than other models tested on our dataset. It is important to point out that the Algerian license plates are far more complex than

Bensouilah, M., Zennir, M. and Taffar, M.
An ALPR System-based Deep Networks for the Detection and Recognition.
DOI: 10.5220/0010229202040211
In *Proceedings of the 10th International Conference on Pattern Recognition Applications and Methods (ICPRAM 2021)*, pages 204-211
ISBN: 978-989-758-486-2

others, for they have no standard model. Thus, many kinds of plate patterns are generated from different fonts, colors, and sizes.

The remainder of the paper is organized as follows. Section 2 briefly reviews related license plate recognition literature. Section 3 describes our proposed method. Section 4 introduces publicly our new ALP dataset, and then we describe in Section 5 the training process on our dataset. The next section presents experimental setups and discusses experimentations and achieved results. Finally, Section 7 concludes the paper.

2 RELATED WORK

In this section, we review recent work that uses deep learning approaches in the context of ALPR. We discuss work related to the two phases: plate detection and plate content recognition.

2.1 Plate Detection

Plate detection is an important phase in the ALPR system. Numerous research works have proposed models that specifically use convolutional neural networks (CNN). Li et al. (Li et al., 2018) trained a CNN capable of detecting the license plate based on the characters in an image. The detected characters are grouped in text regions and provided as initial candidates. Then, false positive plates are removed by a plate/non-plate CNN classifier. This approach takes more than two seconds to process a single image when running on a NVIDIA Tesla GPU K40c.

Xie et al. (Xie et al., 2018) proposed a YOLO-based model called MD-YOLO to detect license plates rotating angle in addition to coordinates and its trust value. Before MD-YOLO, Xie et al. applied a CNN to determine the region of interest in an image, assuming that some distance inevitably exists between two license plates. This approach was executed in real time on three public datasets. Hsu et al. (Hsu et al., 2017) proposed custom models of YOLO and YOLO 2.0 specifically for the detection of license plates. These YOLO's custom models worked better and were able to process 54 FPS on a "GeForce GTX TITAN X" GPU.

2.2 Plate Recognition

After detection, there is an equally important phase for an ALPR system, which is reading or recognizing the content of the plate. For this last phase, there are many papers proposing different approaches. Selmi et al. (Selmi et al., 2017), for example, proposed a convolutional neural network (CNN) with 37 classes. It contains four convolutional layers, three pooling layers, one dropout layer and two fully connected layers. This model is based on a pre-treatment for the extraction of candidate characters, and then all candidates are resized to 32×32 pixels in gray levels to feed the model.

On another hand, Li and Shen (Li and Shen, 2016) took the gray-level image of the detected license plate and divided it into regions or sub-images of 24×24, then switched all image regions to a convolutional neural network (CNN) 9-layers with 36 classes. After the sub-image recognition, the authors eliminated the false positives with a BRNN with the LSTM units.

Wu et al. (Wu et al., 2018), however, proposed a DenseNet based model (Huang et al., 2017), with 68 classes for Chinese license plates. DenseNet is a convolutional neural network (CNN) with dense blocks, containing three types of blocks: convoluted block, dense block and transition block. In their proposed model, there are a convoluted block, three dense blocks and two transition blocks to process a gray-level image of 136×136 pixels and recognize the license plate. Yet, Spanhel et al. (Špaňhel et al., 2017) proposed a convolutional neural network (CNN) with eight output branches recognizing characters on respective positions with 36 classes. It takes a color image of 200×40 pixels as an input.

3 PROPOSED APPROACH

In ALPR systems field, the goal is to build a system that, from an image, detects license plates and effectively recognizes the existence of writing and characters in the image. Our approach addresses the problem of reading license plates in two phases: plate detection followed by plate contents recognition. Figure 1 depicts the overall architecture of our approach.

3.1 Plate Detection

In this first phase, we use the YOLO (Redmon et al., 2016)(Redmon and Farhadi, 2017)(Redmon and Farhadi, 2018) object detector from literature. It is a CNN capable of detecting objects in real time and thus achieving interesting results in terms of compromise (speed/precision) on datasets of the detection of published objects, such as Pascal VOC (Everingham et al., 2010) and Microsoft COCO (Lin et al., 2014).

YOLO uses a single CNN that applies to the entire image. This network divides the image into regions and predicts the bounding boxs and probabilities for

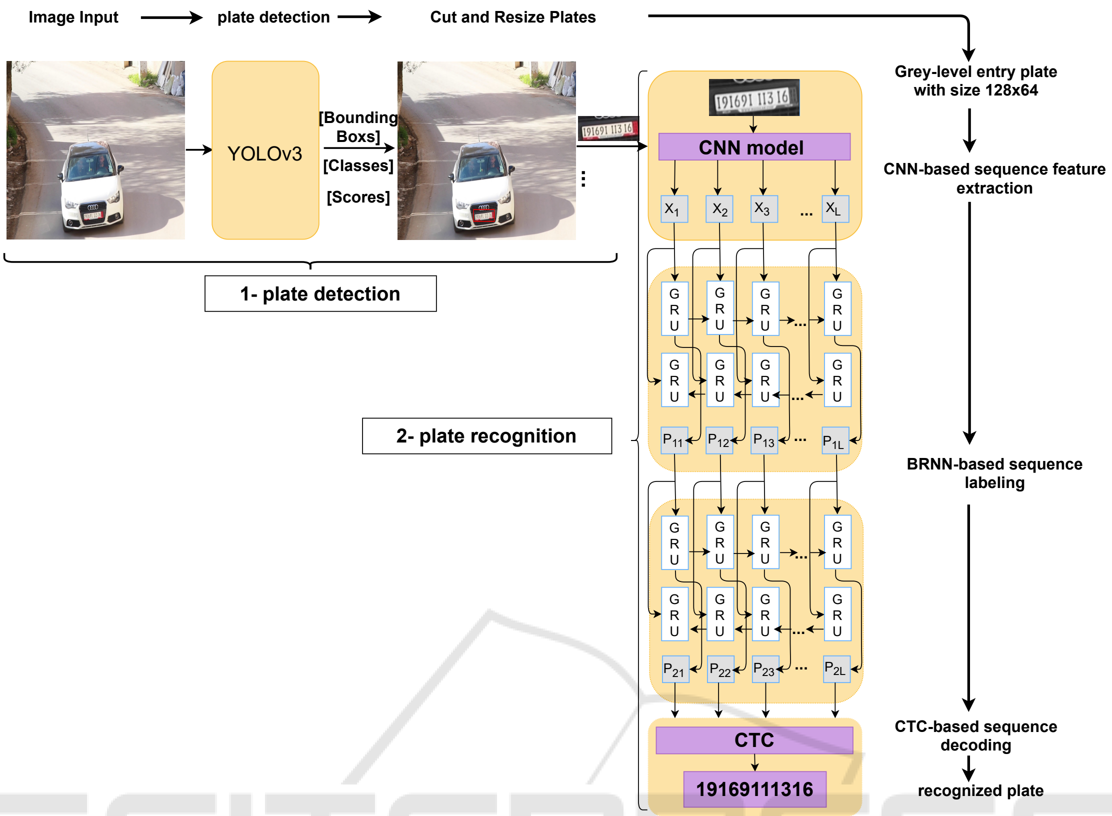


Figure 1: General scheme of our approach of license plates recognition.

each. These bounding boxs are weighted by predicted probabilities. YOLOv3 (Redmon and Farhadi, 2018) is an update to previous versions of YOLO (Redmon et al., 2016)(Redmon and Farhadi, 2017); proposed by Redmon and Farhadi in 2018, it is a more advanced version than the previous one that is more accurate and faster.

YOLOv3 (Redmon and Farhadi, 2018) predicts the boxes at 3 different scales. It extracts the features of these scales using a concept similar to that of pyramid networks. It uses "k-means clustering" to determine the history of encompassing areas. The system uses 9 clusters and 3 arbitrary scales, and then divides these clusters evenly between scales.

In this paper, we compare the Darknet-53 feature extractor (backbone) that is proposed for YOLOv3 (Redmon and Farhadi, 2018) to another named MobileNets (Andrew G. Howard, 2017). MobileNets was proposed by Howard et al. of Google Inc. in 2017, and it is faster and provides good results. We assign to the input the images of 416×416 pixels and use the MobileNet-YOLOv3 implementation (Yang, 2019).

3.2 Plate Recognition

For the second phase, we rely on a type of CRNN network proposed by (Shi et al., 2016). CRNN is a hybrid neural network model whose architecture is specially designed to recognize sequence-like objects in images. It is a combination of a CNN and an RNN.

To extract the sequence of features from an input image, the CRNN model accommodates a basic convoluted neural network (CNN) by keeping convolution layers and max-polling layers by eliminating the fully connected layer. The CNN release is the entry of the recurrent neural networks (RNN) built to make labeling for each frame of the sequence of features.

3.2.1 CNN for the Extraction of Features

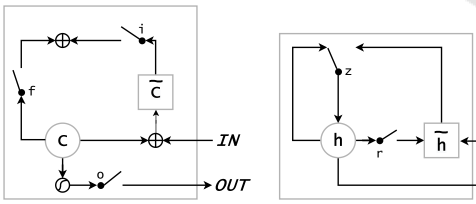
Each image contains features that set it apart from the rest of the images. To extract these features, we developed a model inspired by the CRNN model (Shi et al., 2016) and CNN model it's the VGG-12. The VGG-12 (Simonyan and Zisserman, 2014) consists of 12 layers (convolution and max-polling). It takes raw image inputs and produces robust feature maps that contain high-level descriptions of input images. In other words, it builds representations that bring out the properties of objects learned from the images of inputs. Because a CNN requires input images to be scaled to a fixed size to meet its fixed input size, we fix the input images to the size of 128×64 pixels.

3.2.2 RNN for Sequence Labelling

A bidirectional recurrent neural network is built above convolutional layers, such as recurrent layers. RNN has a strong ability to capture contextual information in a sequence (Williams and Zipser, 1995). In our work, we replace the LSTM (Hochreiter and Schmidhuber, 1997) units with GRU units (Cho et al., 2014) in both BRNN layers. GRU was introduced in 2014 by Cho et al. (Cho et al., 2014) to allow each recurrent unit to adaptably capture dependencies at different time scales (Chung et al., 2014). It is similar to the LSTM unit (e.g., Figure 2), but easier to calculate and implement (Fu et al., 2016). The GRU has blocking units that modulate the flow of information within the unit, however, without having separate memory cells (Chung et al., 2014), represented by the following equations (1): Initially, for $t = 0$, the output vector is $h_0 = 0$.

$$\begin{aligned} z_t &= \sigma(W_z x_t + U_z h_{t-1}) \\ r_t &= \sigma(W_r x_t + U_r h_{t-1}) \\ h_t &= (1 - z_t) h_{t-1} + z_t \tilde{h}_t \\ \tilde{h}_t &= \tanh(W_h x_t + U_h (r_t \odot h_{t-1})) \end{aligned} \tag{1}$$

Where x_t,h_t respectively, are the input and output vectors; z_t,r_t represent the update and reset gates vectors with U, W two parameter matrices.



(a) Long Short-Term Memory (b) Gated Recurrent Unit

Figure 2: Illustration of (a) LSTM et (b) GRU (Shi et al., 2016).

In addition to the fact that GRU units perform better than LSTM, we have augmented the output vector classes of GRU from 256 to 512 to improve the recognition rate of our model. Table 1 shows the configuration of our CRNN model.

3.2.3 Layer Transcription

Transcription is a process of converting the higher frame predictions made by the coding module into a label sequence. Mathematically, the transcription procedure consists of finding the label sequence with the highest probability conditioned by predictions about pre-frames. To accomplish this task, we use the Connectionist Temporal Classification (CTC) proposed by Graves et al. (Graves et al., 2006). The CTC is based on a procedure inspired by the ”forward-backward” algorithm, without segmenting the input sequence before training.

4 TRAINING

We train YOLOv3 with a single class and mini-batches (16 images per batch) using our dataset (3408 images of different sizes), with the input image sized to $416 \times 416 \times 3$.We trained the CRNN with eleven classes (numbers from 0 to 9, plus white special character). Thus, we trained the model with our dataset (2179 Algerian license plates) to mini-batches (32 license plates per batch), where the input image are sized to $128 \times 64 \times 1$. We used an Adam optimizer and a learning rate parameter fixed to $lr = 0.001$ during the training step.

5 DATASET

We collected a set of image and video data that we captured in the municipality of Draria in Algeria using a fixed camera. Then we increased our dataset by public images from websites that can be accessed on the Internet (such as ”google images” [1], ”facebook marketplace” [2] and “Ouedkniss” [3]). We have published our dataset on the GitHub repository (LPA Dataset, 2019).The vehicle objects of our dataset exhibit a wide variability of viewpoints and lighting. Thus, we have built up a learning set of 2408 images, and a test set of 1000 images used to train and test YOLO detector. Figure 3 presents some samples.

For the license plates recognition, we collect the plates extracted during the detection step on our dataset. The latter contains a learning set of $1,775$ plates, and a test set of 404 plates. They were carefully and manually annotated with license plate char-

[1]https://www.google.com/imghp

[2]https://www.facebook.com/

[3]https://www.ouedkniss.com/

Table 1: The configuration of our CRNN.

Example Type of layer	Parameters
Entry	Grey-level image with size 128×64
Convolution Max pooling	filters: 64, kernel: 3×3, s: 1, p: 1 kernel: 2×2, s: 2
Convolution Max pooling	filters: 128, kernel: 3×3, s: 1, p: 1 kernel: 2×2, s: 2
Convolution	filters: 256, kernel: 3×3, s: 1, p: 1
Convolution Max pooling	filters: 256, kernel: 3×3, s: 1, p: 1 kernel: 1×2, s: 2
Convolution Batch normalization	filters: 512, kernel: 3×3, s: 1, p: 1
Convolution Batch normalization Max pooling	filters: 512, kernel: 3×3, s: 1, p: 1 kernel: 1×2, s: 2
Convolution	filters: 512, kernel: 2×2, s: 1, p: 1
BGRU BGRU	units: 512 units: 512
Fully Connected Softmax	units: 11 classes: 11

acters to allow an accurate assessment of optical character recognition. Some samples shown in Figure 4.

Figure 3: Image samples used to train the YOLO detector.

Figure 4: Samples of license plates used to train the CRNN.

6 EXPERIMENT AND RESULTS

In this section, we first describe a set of measures commonly used for evaluating deep learning-based systems. Then, we present our configurations and the experimental considerations that we have imposed by setting the benchmarks and parameters that have allowed us to achieve satisfactory performance and high rates of detection and recognition of Algerian license plates.

6.1 Popular Metrics

First we denote by TP, FP, TN, FN respectively true-positive, false-positive, true-negative and false-negative.

Precision/Recall. It is a matter of using accuracy or error rate for the detection of objects. These measures are summarized in the following equations (2):

$$Precision = \frac{TP}{TP+FP}, \quad Recall = \frac{TP}{TP+FN} \tag{2}$$

IoU: Intersection over Union measures the overlap between two borders. It is given as equation (3):

$$IoU = \frac{Area\ of\ Overlap}{Area\ of\ Union} \tag{3}$$

WER/CER. The performance of character recognition models can be measured by word error rate (WER) and character error rate (CER). WER is the report of reading errors calculated at the word level.

Table 2: Comparison of the performance of YOLOv3 and MobileNet-YOLOv3.

Detection model	Average detection speed (sec/Image)	Recall	Precision	AP^{50}
YOLOv3	0.050	**0.970**	**0.990**	**0.969**
MobileNet-YOLOv3	**0.032**	0.960	0.980	0.954

Table 3: Comparison of the performance of CRNN, Wu-DenseNet, and our model.

Recognition Model	Input Image Size	WAR(%)	CER(%)
CRNN (Shi et al., 2016)	$128 \times 32 \times 1$	train: **0.06**, test: 15.63	train: **0.01**, test: 2.46
Wu-DenseNet (Wu et al., 2018)	$136 \times 36 \times 1$	train: **0.06**, test: 28.29	train: **0.01**, test: 4.33
Our Model(BLSTM)	$128 \times 64 \times 1$	train: **0.06**, test: 9.18	train: **0.01**, test: 1.08
Our Model(BGRU)	$128 \times 64 \times 1$	train: **0.06**, test: **7.94**	train: **0.01**, test: **0.99**

The CER measures the distance of Levenshtein standardized by the length of the word ground-truth.

6.2 Experiment Setups

We implemented our approach using both Keras and Tensorflow libraries with Adam optimization. The training and test run using a PC with Intel Core Xeon CPU and 16 GB GPU (NVIDIA Tesla P100-PCIE).

6.3 Results

Table 2 shows the results obtained by the two models, tested on our dataset (LPA Dataset, 2019), after 22,123 training steps. Compared to YOLOv3, MobileNet-YOLOv3's detection speed is faster, and it always meets real-time requirements. While in terms of Average Precision (AP) performance (Everingham et al., 2010), YOLOv3 is better than MobileNet-YOLOv3, which means that the license plates detected by YOLOv3 are closer to the ground truth.

The evaluation criterion for recognition is the accuracy of the license plate, which means that the recognition is correct when all the characters of a license plate are correctly recognized. To this purpose we used two metrics (CER, WAR). Table 3 shows the results obtained by our model, tested on our dataset (LPA Dataset, 2019), as well by CRNN (Shi et al., 2016) and Wu-DenseNet (Wu et al., 2018). All three models were obtained after 22,188 training steps. Compared to both models, our model is more efficient for Algerian license plates (CER-0.99%, WAR-7.94%).

Table 4 shows the results obtained by our model tested on our dataset (LPA Dataset, 2019) and other results on the Chinese Dataset-1 that is featured in Wu-DenseNet (Wu et al., 2018). Chinese Dataset-1 (Wang et al., 2017) contains 203,774 Chinese license plates for training and 9,986 for testing. Chinese license plates generally come according to a standard model and are identical. On the other hand, Algerian license plates do not have a standard model, causing the existence of many different models (fonts, colors, design) of license plates, which has, therefore, accentuates the difficulty to recognize the Algerian license plates (e.g., Figure 5).

Figure 5: Comparisons between Chinese license plates (top), and Algerian license plates (bottom).

Figure 6 presents some examples of MobileNet-YOLOv3 detection and CRNN recognition result.

7 CONCLUSION

In this article, we proposed an end-to-end license plate recognition system based on the YOLO detector and CRNN. For the detection step, we optimized some training parameters of YOLOv3 and MobileNet-YOLOv3, and then we trained a license plate detection model on datasets with high and uniform intra-class variability of plate patterns.

We carried out comparative experiments of YOLOv3 and MobileNet-YOLOv3, and the results show that MobileNet-YOLOv3 works better on detection speed while YOLOv3 works better on detection accuracy. In the recognition part, we designed and trained our CRNN model which is an improved RNN. We also carried out comparative experiments of our CRNN with other models, and the results show that our system performs better for Algerian license plates. Experimental results show that the proposed system has achieved top performance in terms of recognition speed and recognition accuracy, which

Table 4: Comparison of performance on Chinese Dataset and our Algerian Dataset.

Recognition Model	Dataset	Input Image Size	WAR(%)	CER(%)
Wu-DenseNet (Wu et al., 2018)	Chinoises (Wang et al., 2017)	$136 \times 36 \times 1$	0.01	0.001
	Algériennes (LPA Dataset, 2019)		28.29	4.33
Our Model	Algériennes (LPA Dataset, 2019)	$128 \times 64 \times 1$	7.94	0.99

Figure 6: Samples of MobileNet-YOLOv3 detection and CRNN recognition result.

can fully meet the needs of practical applications. As perspective, we plan to enrich our dataset by exploring a new ways to improve the recognition of Algerian license plates.

REFERENCES

Andrew G. Howard, Menglong Zhu, B. C. D. K. W. W. T. W. M. A. H. A. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv*.

Cheang, T. K., Chong, Y. S., and Tay, Y. H. (2017). Segmentation-free vehicle license plate recognition using convnet-rnn. *arXiv preprint arXiv:1701.06439*.

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Chung, J., Gulcehre, C., Cho, K., and Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Du, S., Ibrahim, M., Shehata, M., and Badawy, W. (2012). Automatic license plate recognition (alpr): A state-of-the-art review. *IEEE Transactions on circuits and systems for video technology*, 23(2):311–325.

Everingham, M., Van Gool, L., Williams, C. K., Winn, J., and Zisserman, A. (2010). The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338.

Fu, R., Zhang, Z., and Li, L. (2016). Using lstm and gru neural network methods for traffic flow prediction. In *2016 31st Youth Academic Annual Conference of Chi-*

nese Association of Automation (YAC), pages 324–328. IEEE.

Graves, A., Fernández, S., Gomez, F., and Schmidhuber, J. (2006). Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Hsu, G.-S., Ambikapathi, A., Chung, S.-L., and Su, C.-P. (2017). Robust license plate detection in the wild. In *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6. IEEE.

Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. (2017). Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708.

Khan, M. A., Sharif, M., Javed, M. Y., Akram, T., Yasmin, M., and Saba, T. (2017). License number plate recognition system using entropy-based features selection approach with svm. *IET Image Processing*, 12(2):200–209.

Laroca, R., Severo, E., Zanlorensi, L. A., Oliveira, L. S., Gonçalves, G. R., Schwartz, W. R., and Menotti, D. (2018). A robust real-time automatic license plate recognition based on the yolo detector. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–10. IEEE.

Li, H. and Shen, C. (2016). Reading car license plates using deep convolutional neural networks and lstms. *arXiv preprint arXiv:1601.05610*.

Li, H., Wang, P., You, M., and Shen, C. (2018). Reading car license plates using deep neural networks. *Image and Vision Computing*, 72:14–23.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer.

Liu, G., Ma, Z., Du, Z., and Wen, C. (2011). The calculation method of road travel time based on license plate recognition technology. In *Advances in information technology and education*, pages 385–389. Springer.

LPA Dataset, L. P. o. A. D. (2019). *License Plates of Algeria Dataset*. https://github.com/mouad12345/License_Plates_of_Algeria_Dataset.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788.

Redmon, J. and Farhadi, A. (2017). Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271.

Redmon, J. and Farhadi, A. (2018). Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*.

Selmi, Z., Halima, M. B., and Alimi, A. M. (2017). Deep learning system for automatic license plate detection and recognition. In *2017 14th IAPR international conference on document analysis and recognition (ICDAR)*, volume 1, pages 1132–1138. IEEE.

Shi, B., Bai, X., and Yao, C. (2016). An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE transactions on pattern analysis and machine intelligence*, 39(11):2298–2304.

Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Špaňhel, J., Sochor, J., Juránek, R., Herout, A., Maršík, L., and Zemčík, P. (2017). Holistic recognition of low quality license plates by cnn using track annotated data. In *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6. IEEE.

Wang, X., Man, Z., You, M., and Shen, C. (2017). Adversarial generation of training examples: applications to moving vehicle license plate recognition. *arXiv preprint arXiv:1707.03124*.

Williams, R. J. and Zipser, D. (1995). Gradient-based learning algorithms for recurrent. *Backpropagation: Theory, architectures, and applications*, 433.

Wu, C., Xu, S., Song, G., and Zhang, S. (2018). How many labeled license plates are needed? In *Chinese Conference on Pattern Recognition and Computer Vision (PRCV)*, pages 334–346. Springer.

Xie, L., Ahmad, T., Jin, L., Liu, Y., and Zhang, S. (2018). A new cnn-based method for multi-directional car license plate detection. *IEEE Transactions on Intelligent Transportation Systems*, 19(2):507–517.

Yang, A. (2018 (accessed February 29, 2019)). *A Keras implementation of Yolov3*. https://github.com/Adamdad/keras-YOLOv3-mobilenet.