Vehicle License Plate Recognition

Aman Jain, Jatin Gupta, Somya Khandelwal, Surinder Kaur*

Information Technology Bharati Vidyapeeth's College Of Engineering, New Delhi, India
Emails: aman.jain744@gmail.com; jatingupta5jan@gmail.com; somyakhandelwal27@gmail.com; kaur.surinder@bharaividyapeeth.edu

* Correspondence: aman.jain744@gmail.com

Abstract

One of the most significant parts of integrating computer technologies into intelligent transportation systems (ITS) is vehicle license plate recognition (VLPR). In most cases, however, to recognize a license plate successfully, the location of the license plate is to be determined first. Vehicle License Plate Recognition systems are used by law enforcement agencies, traffic management agencies, control agencies, and various government and non-government agencies. VLPR is used in various commercial applications, including electronic toll collecting, personal security, visitor management systems, parking management, and other corporate applications. As a result, calculating the correct positioning of a license plate from a vehicle image is an essential stage of a VLPR system, which substantially impacts the recognition rate and speed of the entire system. In the fields of intelligent transportation systems and image recognition, VLPR is a popular topic. In this research paper, we address the problem of license plate detection using a You Only Look Once (YOLO)-PyTorch deep learning architecture. In this research, we use YOLO version 5 to recognize a single class in an image dataset.

Keywords: IOU; ITS; LPR; VLPR; YOLO v5; YOLO v5s

1. Introduction

Vehicle License Plate Recognition aims to detect the presence of a license plate on a vehicle in a concise amount of time and with high accuracy. The license plate of a vehicle is usually referred to as a "number plate." It's a metal plate fitted to a car that carries its official registration number imprinted on it. The vehicle can be identified using the number plate placed on both the front and rear sides. Over the last few decades, License Plate Recognition (LPR) has been a methodology utilized in various Intelligent Transportation Systems (ITS). It replaces the efforts of humans in recognizing the license plate. This proposed LPR system is primarily used in a car park with a static camera rather than a high-speed vehicle plate recognition setting. In a traditional LPR system, the initial step is to collect a vehicle image with the license plate using a camera. After that, several image processing techniques are used to convert the digital image into usable information. License Plate (LP) detection, character segmentation [9], and character recognition are the three steps of most LPR systems. Because non-observance of LP would lead to failure in the subsequent phases., the early steps require better accuracy or near-perfection, and this paper focuses on the detection part only. To reduce processing time and prevent false positives, many approaches scan the car first, then the license plate. Even though LPR has been discussed extensively in the literature, many studies are still unreliable in the actual world.

DOI: https://doi.org/10.54216/FPA.040102
Received February 02, 2021 Accepted June 05, 2021

Why is it necessary to register?

• A legally purchased motor vehicle must be registered with the local authority or the government. This helps prevent theft of that vehicle and identify the crimes committed using that vehicle. By registering the vehicle, a record is created, which gives the vehicle its unique identity.

Contents of Number Plates

• Number plate comes in various colors and numeric or alphanumeric characters. These colors signify the authority of the owner of that vehicle. The numeric or alphanumeric character combination varies from country to country, and it is the record that gives the vehicle its unique identity.

2. Literature Survey

Character segmentation and character classification steps were used in previous works on License Plate recognition:

Various handcrafted algorithms, including projections, connectedness, and contour-based picture components, are commonly used in character segmentation. Because it uses a binary picture or intermediate representation as input, noise, low resolution, blur, and deformations in the input image significantly impact character segmentation quality[9].

Character classification typically utilizes one of the optical character recognition (OCR) methods - adopted for LP character sets. Because classification comes after character segmentation, the quality of end-to-end recognition is highly influenced by the segmentation method used. End-to-end Convolutional Neural Networks (CNNs) [12]based solutions were presented to handle the problem of character segmentation, accepting the entire LP picture as input and producing the output character sequence[6].

The segmentation-free model is based on connectionist temporal classification (CTC) loss and variable-length sequence decoding. To generate character classes probability, it leverages handcrafted features LBP based on a binarized picture as CNN input[7][12]

. The input sequence for the bi-directional LongShort Term Memory (LSTM) [5]based decoder is created by applying the sliding window technique to all input image positions. CTC loss is employed for pre-segmentation-free end-to-end training since the decoder output and target character sequence lengths are different[7].

The sliding window method was substituted by CNN output spatial splitting to the RNN input sequence ("sliding window" over feature map instead of input) in this model[5][8].

There are two types of license plate segmentation algorithms available: projection-based and connected component (CC)-based. Because the lettering and backgrounds in a license plate have visibly different nine hues, they should have opposite binary values in the binary picture. For character segmentation, projection-based methods use the histograms of vertical and horizontal pixel projections[9]. The top and bottom limits of the characters are determined by projecting the binary license plate image horizontally, then vertically to separate each character. The rotation of the license plate might easily affect this type of procedure. Based on 4 or 8 neighborhood connectivity, CC-based algorithms identify connected pixels of the binary license plate into components. Characters can be extracted from rotating license plates using CC-based approaches. However, they cannot be segmented effectively if they are connected together or torn apart.

3. Proposed Solution

The issue at hand can be resolved by utilizing various object detection methods, where we must train our model using photos of cars and other vehicles with license plates, but in this research work, we are focused on using YOLO v5s[1]. YOLO v5s are used to detect the presence of a number plate in a given image along with its location and confidence of being a number plate in the detected location[1]. We used Google Colab for this experiment because of the free availability of GPUs.

The proposed solution, along with character recognition, is likely to be integrated with a web application. The backend will display the contents to the user after the camera takes the image of the vehicle.

DOI: https://doi.org/10.54216/FPA.040102

Received February 02, 2021 Accepted June 05, 2021

3.1. YOLO

YOLO v5 is a real-time, single-stage object detection model that builds on YOLO v2 [2]with several improvements. The detection is done by applying 1 x 1 detection kernels on feature maps of three different sizes at three different places in the network.
The YOLO v5 network is made up of three primary components.
1) Backbone - A convolutional neural network that accumulates and produces visual features at various granularities is known as the backbone.
2) Neck - A set of layers that integrate and mix picture characteristics before passing them on to prediction.
3) Head - Takes box and class prediction steps while consuming features from the neck.
YOLO v5 is a PyTorch implementation rather than a fork from the original Darknet. YOLO v5 [3] introduces a new PyTorch training and deployment framework that improves state-of-the-art for object detectors.
YOLO v5[3] feeds training data through a data loader with each training batch, which augments data online. Scaling, color space adjustments, and mosaic augmentation are the three types of augmentations performed by the data loader. Mosaic data augmentation is the most unique of them, combining four photos into four random-ratio tiles.
YOLO v5[3] provides four versions, namely
• YOLO v5s, which is a miniature version
• YOLO v5m, which is a medium version[3]
• YOLO v5l, which is a large version
• YOLO v5xF, which is an extra-large version

3.2. Mean Average Precision

Precision is a metric that asks, "How often does your model estimate correctly?" The recall is a metric that asks, "Has your model guessed every time it should have?"
Precision is a metric that gauges how accurate your forecasts are. In other words, the percentage of your predictions that are correct[10].

Recall is a metric that assesses how well you remember all of the positives.

Some mathematical definitions related to Precision and Recall :

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

TP = True positive
TN = True negative
FP = False positive
FN = False negative

The average of AP is mAP (mean average precision). We compute the AP for each class and average it in some situations. However, in some situations, they are interchangeable. There is no distinction between AP and mAP in the COCO context.

3.3 Intersection over Union (IoU)

Intersection over Union is a statistic for determining how accurate an item detector is on a given dataset. Object detection problems, such as the popular PASCAL VOC challenge, frequently employ this assessment metric[11].

IOU = Area of Overlap / Area of Union

Or

IOU = Overlapping region / Combined Region

DOI: https://doi.org/10.54216/FPA.040102

Received February 02, 2021 Accepted June 05, 2021

Intersection over Union (IoU) is a statistic that compares our predicted bounding box to the ground truth bounding box. The concept is to compare the ratio of the overlapped area of the two boxes to the total combined area of the two boxes.

4. Dataset

Dataset for this experiment was taken from various sources. It consists of images of cars with number plates of different countries[4]. They are Indian, Tunisian, United Kingdom, and Arabian number plates in the dataset. Images are taken from different angles and distances to make the dataset more flexible. Some images contained more than one number plate and were annotated accordingly[4]. Dataset consists of 800 images. It is bifurcated in the ratio of 70:20:10 for train, test, and validation, respectively. All the images were annotated manually using the roboflow web app. The single class named "plate" was used in annotation[4]. Bounding boxes are created in the roboflow app. Roboflow web app stores the bounding box coordinates, i.e., number plate coordinates, in a text file; for YOLO v5, it also creates a YAML file, i.e., data configuration file, which consists of the address of the train, test, and validation data and the number of class that the data is divided into along with the class name[10].

5. Inference and Results

We have used YOLO v5s[3], a miniature version of YOLO v5, for detecting number plates in images. It can be implemented in real cameras for plate detection as YOLO v5 provides the option of detection from video streams. Here are the statistics about YOLO v5s. When evaluating the accuracy of any model, we must consider many parameters such as mAP (mean Accuracy Precision), recall, and precision. We can also study parameters such as the object [1]loss and classification loss. Parameters like object[1] loss and categorization loss can also be investigated.

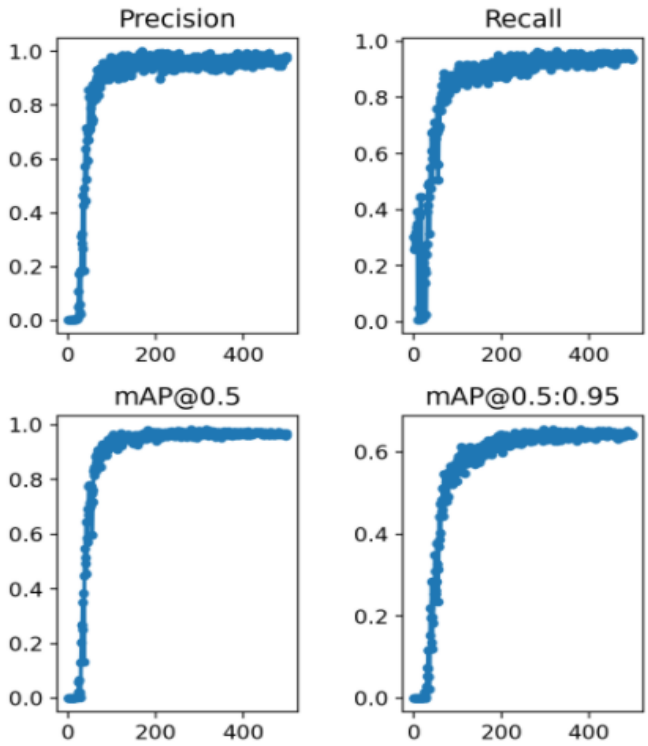

Figure 1: Plots of precision, recall and mean average precision (mAP) over the training epochs for the training and validation set.

DOI: https://doi.org/10.54216/FPA.040102

Received February 02, 2021 Accepted June 05, 2021

The values of mAp, precision parameter, and recall for Yolo v5s are 0.984, 0.953, and 0.988, respectively.

On running the trained model on an image, the output contains a new image generated with a bounding box(if the object is detected) with the class name, i.e., plate, along with its class confidence value.

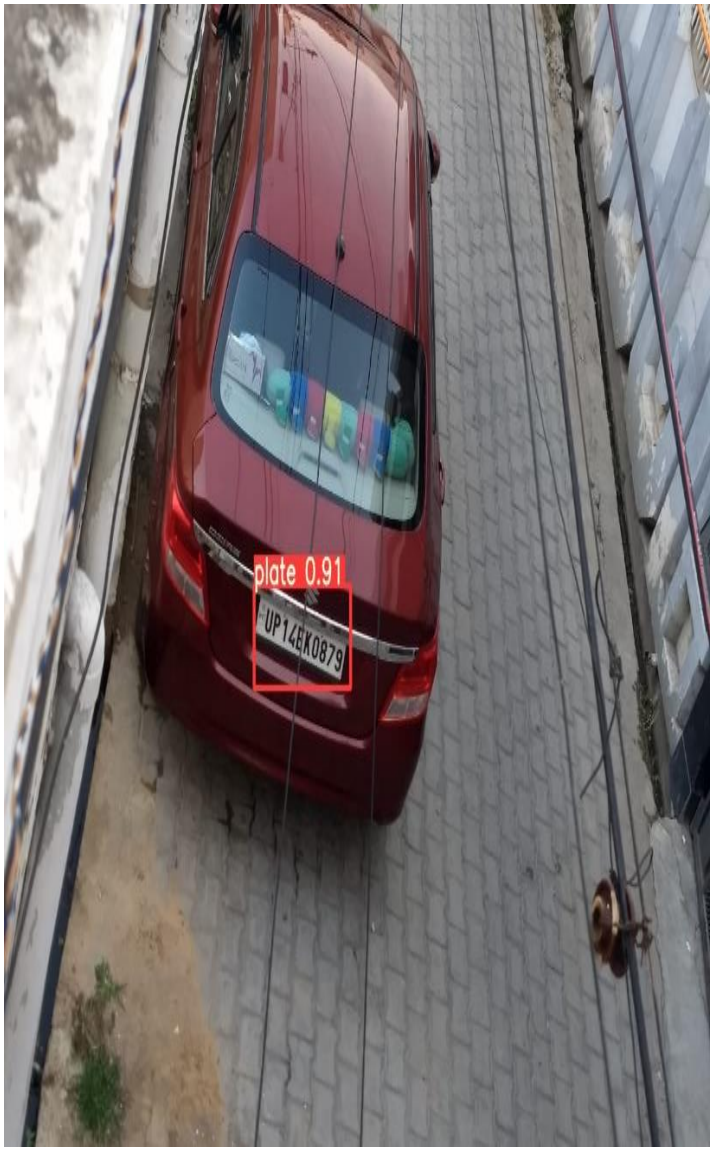

Figure 2: The output of running model on a test image

DOI: https://doi.org/10.54216/FPA.040102
Received February 02, 2021 Accepted June 05, 2021

6. Conclusion

This research work will significantly benefit the future of object detection, especially vehicle detection. It uses the latest version of YOLO, i.e., version 5, and the miniature version for number plate detection. YOLO v5, in a traditional sense, does not bring any novel architectures or losses, or techniques to the table. However, it vastly improves the speed with which individuals may integrate YOLO into their existing pipelines. It is written in PyTorch (python), while the older version was in C, making it more friendly to people and companies. It includes a simple approach to defining experiments utilizing modular config files, quick inference, mixed-precision training, improved data augmentation approaches, etc. There is yet to be a research paper released for YOLO v5. The outcome of this research work was pretty good, with the mAP@0.5 values of 0.984 on the dataset of 800 images split into 70:20:10 ratio for train, test, and validation.

6. Future Work

This work can further be extended to recognize the characters of the detected number plate by the proposed solution and implementation in a web app connected to a database of car owners to make it more versatile and beneficial for the country's judicial system.

An increase in the number of images in the dataset can result in better detection accuracy. Furthermore, if the number plates of a particular country are used, the model will work better. Dataset can be changed to video format further to test the detection rate and accuracy of the model.

Hyperparameter tuning can be done with more precision to generate better results. YOLO v5 also gives the option of custom architectures and anchors, which can be explored more.

References

[1] Hui, J., 2018. Real-time Object Detection with YOLO, YOLOv2 and now YOLOv3. [online] medium.com. Available at: [Accessed 3 April 2021].

[2] Redmon, J., Divvala, S., Girshick, R. and Farhadi, A., n.d. You Only Look Once: Unified, Real-Time Object Detection. [online] Pjreddie.com. Available at: [Accessed 5 April 2021].

[3] Chablani, M., 2017. YOLO — You only look once, real time object detection explained. [online] Medium. Available at: [Accessed 4 April 2021].

[4] C. N. E. Anagnostopoulos, I. E. Anagnostopoulos, I. D. Psoroulas, V. Loumos, and E. Kayafas, "License Plate Recognition From Still Images and Video Sequences: A Survey," IEEE Transactions on Intelligent Transportation Systems, vol. 9, no. 3, pp. 377–391, Sep. 2008.

[5] H. Li and C. Shen, "Reading Car License Plates Using Deep Convolutional Neural Networks and LSTMs," arXiv:1601.05610 [cs], Jan. 2016, arXiv: 1601.05610.

[6] A. Graves, Supervised Sequence Labelling with Recurrent Neural Networks, 2012th ed. Heidelberg ; New York: Springer, Feb. 2012. 2, 3

[7] A. Graves, S. Fernandez, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in Proceedings of the 23rd international conference on Machine learning. ACM, 2006, pp. 369–376.

[8] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," Neural Computation, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[9] T. K. Cheang, Y. S. Chong, and Y. H. Tay, "Segmentation free Vehicle License Plate Recognition using ConvNetRNN," arXiv:1701.06439 [cs], Jan. 2017, arXiv: 1701.06439.

[10] 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, ser. ICVGIP '16. New York, NY, USA: ACM, 2016, pp. 6:1–6:8.

[11] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu, "Spatial Transformer Networks," arXiv:1506.02025 [cs], June. 2015, arXiv: 1506.02025.

DOI: https://doi.org/10.54216/FPA.040102

Received February 02, 2021 Accepted June 05, 2021

[12] H. Li, P. Wang, and C. Shen, "Towards End-to-End Car License Plates Detection and Recognition with Deep Neural Networks," ArXiv e-prints, Sep. 2017.

DOI: https://doi.org/10.54216/FPA.040102

Received February 02, 2021 Accepted June 05, 2021