AN INTELLIGENT MOBILE APPLICATION FOR CAR LICENSE PLATE DETECTION AND ANALYSIS USING MACHINE LEARNING ALGORITHM

Mengcheng Han[1] and Yu Sun[2]

[1]Santa Margarita Catholic High School, 22062 Antonio Pkwy,
Rancho Santa Margarita, CA 92688
[2]California State Polytechnic University,
Pomona, CA, 91768, Irvine, CA 92620

ABSTRACT

How did image recognition and object analysis function to bring convenience to people's lives? Within this question I bear in my mind, I started to explore and build this Automatic Car License Plate Detect and Analyze Project. Since cameras are being widely used for recording and analyzing vehicle information, it has been a great cost to buy such intelligent devices. Guided by recent research on machine learning approaches , we solve this financial problem by designing and implementing a mobile phone application that automatically utilizes the camera installed on the phone to analyze the information of the car license plate. Our design is built to provide users with an accessible way to analyze license plates in complex environments.

KEYWORDS

Machine learning, LPR, OCR libraries.

1. INTRODUCTION

The License Plate Recognition (LPR) System is an important study of public transportation in modern society [6][7]. The LPR System is used in many fields such as traffic law enforcement, toll road payment, community security, in the long decades of study and experiments. It benefits society by substituting a learning-based system to do the work used to be done by the human labor force. For example, a security guard would need to be present at a parking lot to record guest vehicles' information and time parked to tax. However, the LPR System automatically does those vehicle identifications that used to be done by humans, which have greatly increased the efficiency and accuracy of this job [8]. Although a few techniques have already been developed to fulfill these various situations, a unique camera is necessary to be installed in a fixed position to function properly. So we discovered that it is possible to implement a new LPR system that could adapt into mobile phone cameras based on an object-recognition model. This challenge is revolutionary in that we use another way to accomplish the same task of the traditional LPR System but without any expenditure (as long as a mobile phone with a camera is acquired to run the program).

As said above, an LPR camera is used in most situations for detection, which is a specialized camera that is designed to capture characters on the license of a car in motion [9]. A common feature of them is that they are bought as a unique device, and their price could be $100 to $800

David C. Wyld et al. (Eds): AIAP, IT, MoWiN, CCSEIT, CNSA, ICBB - 2022
pp. 75-82, 2022. CS & IT - CSCP 2022 DOI: 10.5121/csit.2022.120406

or more. Compared to a free phone application, these equipment are much more expensive for the task they are to complete - car license detection. On the other hand, the LPR System, though is used nationwide in many different forms, has the same basic technique - license detection, image fragmentation, and character recognition. However, the accuracy can be interrupted under extreme environments because the algorithm produces results based on the similarity of pixels. For example, the LPR camera might be blurred or covered in weather like rain or snow, and the results of detection are not desirable anymore for either government use or business. In fact, the first LPR System was implemented in the 20th century which means the same technology has been used for more than 40 years. Thus, another practical problem of the current LPR System contains a much more complex algorithm (probably the reason for using a specialized camera) than a trained machine learning model, which also solves the task efficiently with a simpler approach [3].

In this paper, we follow the same line of research by the LPR System using machine learning. Our approach is based on an automatic object-detection model YOLO-V3 to handle the tasks of detecting car licenses in pictures or videos. After training the machine learning model with datasets, it recognizes licenses that have different plat formats in nonuniform outdoor illumination. While the other two parts of the new LPR System in this project are also implemented using machine learning, we built a web server for the LPR System adapted into a phone application. Our goal is to abridge the process of car license recognition, and this step provides the user a simpler way by using a phone application rather than buying an LPR camera, which could be lavish. Mobility is an important feature of this project that the analysis can be done at any location with an internet connection, while the same result could only be resolved by taking pictures at a fixed camera position. Therefore, we believe that this proposed application can be a reformation to the original LPR System and bring benefits to the users.

In the two application scenarios in part 4, we demonstrate how the above solution is carried out and the accuracy of the algorithm we implemented. First, we design an experiment to test different machine learning models for their accuracy in detecting the license plate in the image [4]. In fact, Yolo-V3 is the model that provides a better result of detecting US license plates within 300 test images. while comparing to Yolo-V4 which has only 50% accuracy. Second, we did another experiment regarding different libraries that contain specific functions to recognize text in the image. The results indicate that Easy-OCR is the one that has the highest accuracy of 88.26% with 50 test images. Although the percentage isn't perfect among these experiments, it demonstrates our project could provide usefulness to the app users with its mobility and compatibility. Further modification to the algorithm and libraries is possible to be implemented to enhance the accuracy of the analysis.

To conclude, the rest of the paper is organized as follows: Section 2 gives concrete details on challenges within this project such as picking a specific OCR library, implementing camera function inside phone application, and fixing code compatibility. Section 3 focuses on the details of our methodology to solve the challenges mentioned in Section 2. The solution walks through the entire process of code implementation and compiled together into one LPR System; Section 4 presents the experimental data that was produced during the implementation, followed by the related document on this technology from different scholars; Section 6 gives the conclusion and possible future work that is yet to be done for this project.

2. CHALLENGES

In order to build the project, a few challenges have been identified as follows.

2.1. Picking A Specific Image Recognition and OCR Library is Difficult

Choosing a properly working and high accuracy OCR library is the core part of this type of project. For instance, there are specific issues within the Pytesseract Library that we could not implement the trained model on Google Colab. On another hand, we also tried KerasOCR Library which has low accuracy in recognizing the characters on the license plates. Having a high accuracy model is mandatory for this type of user experience. Thus, I picked EasyOCR Library as the final solution to this challenge. It is perfectly compatible with Google Colab and has 95% accuracy on overall detections. Moreover, the algorithm being used within this library is designed better than the previous two so much so that it took less processing time to do more analysis.

2.2. Installing and Implementing Camera Functionality for our App can Potentially Pose A Problem

Enabling the camera function is necessary for the mobile phone application to take pictures of car license plates. But getting permission to use the camera installed on the phone physically and testing this function on a virtual phone open in a computer is complicated. We specifically need to allow not only photos to be taken, but also access to the camera roll on most phones. As a result, the camera and mobile app are implemented with Flutter using Dart as the coding language as they have pre-built libraries for this solution. Also, setting up the camera in the virtual testing machine. requires some modification on the CPU of the computer through BIOS since it is tested on a virtual phone model instead of an actual phone. So I switched the setting of the CPU on my laptop and finally got the camera to work.

2.3. Due to the Multi-Faceted Nature of the Project, Integrating Multiple Libraries in Python can Lead to General Problems

It is difficult to adapt different libraries into this image-recognition project, especially using python and dart as main languages because I am more confident coding with java [11]. For instance, libraries that contain machine learning algorithms such as Pytesseract update their syntax periodically. So it is hard for programmers like me to implement functions with those syntaxes changing over time. By looking up relative resources, a variety of syntax errors are solved through debugging. It takes time and patience to go through each line of code to find out those “off by one” errors. Moreover, libraries' syntax changing over time also slows the process development of this project. With 6 months of effort in debugging, I was able to run through the whole project without any syntax error.

3. SOLUTION

This project is built based on YOLO V3 (You Only Look Once), which is a specific machine learning algorithm that is very efficient at analyzing pictures in real-time [10]. We used this advantage to train its function of recognizing license plates based on its function of detecting cars in the pictures. On the other hand, the phone application functions to bring the entire project together as it allows for ease of access to the research and program we have created. “What’s My License”, the APP written for this project, is implemented via Android Studio using the programming language Dart.

Once the object is detected from the picture taken from the mobile application, the program will send the image data through a web-based Python flask server to develop a machine-learning model [12]. Then, by utilizing the cropping function to separate the character-contained license plate from the rest of the image, we used tools implemented in Easy-OCR(Object Character

Recognition) library to analyze the picture. This library looks into the similarity of characters that need to be recognized in the picture and returns texts that are successfully detected and the confidence. Finally, the program would send the license plate information back to the user's phone through the Python Flask Server and display it on a separate application page.

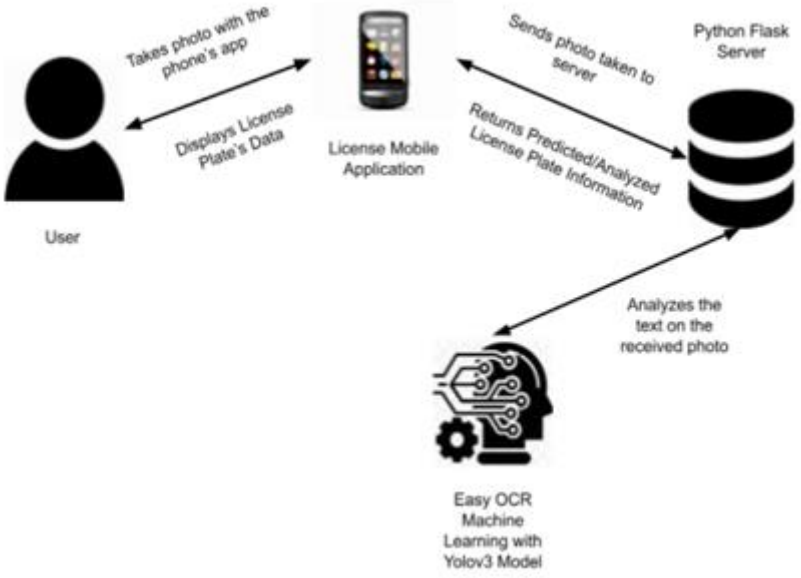

Figure 1. Overview of the APP

The APP that users download and install from google PlayStation / Apple App store is a Flatter application written in the programming language DART. By using dart, we can customize the theme colors, UI pages design, and App logos to produce a more efficient phone APP. The app "What's my license" uses the "Camera" tool to access the camera function on the phone to take desired images. It is adapted to mobile platforms to have easier access and experience of using this project in some real cases.

The user can take a picture of the license plate by clicking on the button, and the image data would be sent back to this project's local storage through a Web-Request of Python Flask Server. The Python Flask Server is a library that allows functions to turn locally running code into a server application that is accessible via HTTP request [13]. Thus, the image taken by the application is sent to the server through our web API, then the picture can be analyzed by the machine learning model by just calling the HTTP request on the server [5]. Once the analysis is done, the Python Flask Server would send a string response that contains the license number being detected back to the mobile application. After the whole process is done, the app would display the information on a separate screen.

As for how the analysis is actually done, that is done in the Python back-end code that is called through the Flask HTTP request. To begin, the original model is trained using 100s of photos of license plates. This data set is trained on the YOLO-V3 machine learning library/model for object recognition. We chose this specific library because its feature set is useful for license plate detection as it has some partial training with vehicle image detection already. We also utilized Tensorflow while training the model. Both of these in tandem allowed us to fully train the model and isolate license plates within any image. We then saved those weights to be utilized later on. (Something like ... now that this code has isolated the license plate, we move on to extract the text from it). Furthermore, Easy-OCR is another library we use for character/text recognition

after the actual license plate is cropped out of the rest of the image. The function, which is implemented for this project, matches similarities of possible characters in the picture and returns the results in string variable: the model can complete this task by taking in a specific training data set that contains pictures already analyzed and determine new data results by the machine learning algorithm.

To conclude, this LP-Detecting project was achieved by a mobile phone application that can take image data. By implementing a Python Flask Server, this application transfers images to be analyzed by a license-recognition HTTP request and displays the detected information back to the users.

4. EXPERIMENTS

4.1. Experiment 1

Data (Accuracy) Yolov3/Yolov4
Training on 100 images (US) - 61.22%
Training on 300 images (US) - 81.54%
Training on one type of license plate but detecting multiple - 54.70%
Training on multiple state license plates and European license plates - 72.31%

Experiment 1 is designed to test different machine learning models' efficiency in detecting different types of license plates. By implementing a generator using Python, the model can process a large number of test images. The accuracy of the analysis out of this experiment is more accurate than doing tests with single or multiple images manually. On the other hand, Yolov3 provides us the best result of an 81.54% accuracy in the training on 300 images that contain US license plates. In fact, images that failed to be analyzed by Yolov3 mostly have low quality because the learning algorithm cannot function properly while the picture is too blurry.

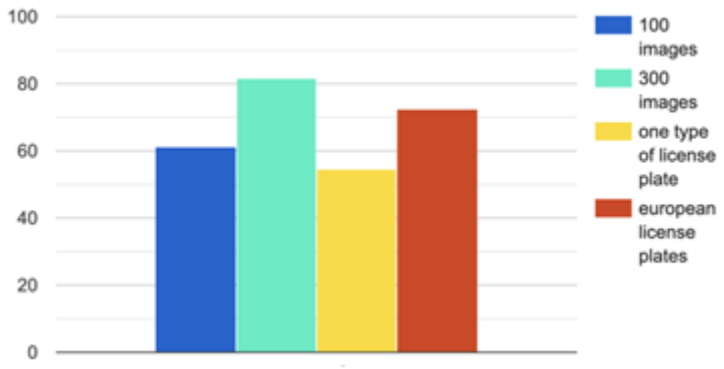

Figure 2. Result of experiment 1

4.2. Experiment 2

Experiment 2: Which machine Learning Model is most efficient for text recognition?
Data
TensorFlow Keras - 82.33%
Pytesseract - 79.01%
EasyOCR - 88.26%

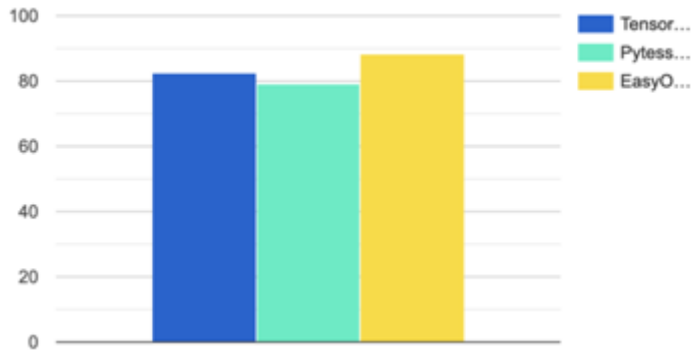

Figure 3. Result of experiment 2

Experiment 2 is to test several machine learning models running text-recognition algorithms on images that are already detected to have a license plate. These results do not account for highlighting more than required, but simply the accuracy of the text read on the screen. Out of the three libraries being tested, EasyOCR has the highest percentage among the other two, which is also the reason we choose to use this library in our project. Although it recognizes text with 88.26% accuracy, special signs and other language characters are not included.

Based on both experiments we have done above, we believe our solution has both stable performance and high accuracy.

5. RELATED WORK

This research paper implements a similar project to ours as it is focused on License Plate Detection. Its functionality is adopted to detect characters and license plates from different angles and different regions around the world. The researchers implemented modifications to the YOLOv2 networks in the paper, which is similar to the YOLO V3 model being trained in our project. However, theirs is used for locating the license plate in various scenarios. The testing data being used in the paper is focused on extreme viewing angles and regional designs, while our project emphasizes the mobility of algorithms utilized through a mobile application. Most users take the picture by phone camera that would most likely have a front-facing view of the license plate. Overall, both of the models use some sort of YOLO machine learning implementation but are focused on different goals.

In this research paper, the author concentrated on improving the processing speed and detection accuracy of their license plate recognition model [1]. They used an image downscaling method to achieve their goal as well as a method to efficiently limit what parts of the image had to be processed. Their modifications to this as well as the classifier resulted in a significantly faster and more accurate model than most standard implementations. One of the approaches of this research paper is similar to one of the python libraries in my project - the Pytesseract library. They both degrade the quality of the picture to a level that could accelerate the detection without sacrificing accuracy. This downgrade of the photo also is utilized to decrease any noise that may unnecessarily confuse the model. Ultimately, the focus of this research paper does not directly correlate to the goal of my project in that our work is focused more so on implementing machine learning via a mobile application, not simply trying to find the fastest prediction possible.

The article discusses an automatic LP-detect system using a Convolution Neural Network to train the model [2]. The system provides results with a great percentage of accuracy under complex environments and distorts perspectives. There are similarities between the Deep Learning System

implemented by the author and mine in that both of our goals are to analyze the license plates image into useful information. In addition, “What’s My License” is also tested for performance under extreme prerequisites such as low pixel, customized license plate. Perhaps all of these projects did a great job on data-collecting and analyzing, but our project is adapted to a mobile application to enhance its usefulness. Installing the APP grants people an easier way for LP detecting purposes in real-life scenarios instead of running source code on PC.

6. CONCLUSION

In this project, we designed a mobile application incorporating multiple services through Google-Colab to achieve the very same goal of the LPR System, which is already implied in many fields [14]. The application we proposed can be split into three parts: analysis, server, and application. First, detection and recognition are done in the analysis part; The trained machine learning model Yolov3 scan through the image that contains license plate and crops the image into a rectangle to separate the license plate from complex pixels; The function implemented by tools in the EasyOCR library is then used to recognize the text on the license plate and return the information as a string variable. Furthermore, we set up a Python-Flask server to allow the users to upload their own images and get results from the mobile application through HTTP requests. Last but not least, we use android studio and the programming language DART to implement a phone app that is more customizable than other applications, so the other two parts of this project can be fused into one without any compatibility error [15]. Throughout the entire development process, we also did a lot of experiments to ensure our approach is efficient; For example, we test different machine learning models to see which of them have a higher accuracy of detection within hundreds of sample pictures. The results indicate the effectiveness of our design that provides not only high accuracy analysis but also higher mobility than the original LPR System.

There are possible limitations in this phone application that is yet to be solved with more advanced technology, such as accuracy of the analysis could flat for a low-quality picture; there must be a stable internet connection for the transportation of information between the host and server; the function we currently developed can only analyze photos, so optimization on file compatibility like videos would provide more convenience to users.

To enhance the accuracy of analysis, we can try to implement other OCR libraries since different algorithms within each of them can produce different results. Moreover, by optimizing the camera function in the phone application to upload video files, users can have a better experience while using our application.

REFERENCES

[1] Z. Selmi, M. Ben Halima and A. M. Alimi, "Deep Learning System for Automatic License Plate Detection and Recognition," 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), 2017, pp. 1132-1138, doi: 10.1109/ICDAR.2017.187.

[2] Y. Yuan, W. Zou, Y. Zhao, X. Wang, X. Hu and N. Komodakis, "A Robust and Efficient Approach to License Plate Detection," in IEEE Transactions on Image Processing, vol. 26, no. 3, pp. 1102-1114, March 2017, doi: 10.1109/TIP.2016.2631901.

[3] Goodfellow, Ian, Yoshua Bengio, and Aaron Courville. "Machine learning basics." Deep learning 1.7 (2016): 98-164.

[4] Jordan, Michael I., and Tom M. Mitchell. "Machine learning: Trends, perspectives, and prospects." Science 349.6245 (2015): 255-260.

[5] Zhang, Xian-Da. "Machine learning." A Matrix Algebra Approach to Artificial Intelligence. Springer, Singapore, 2020. 223-440.

[6] Vuchic, Vukan R. "Urban public transportation systems." University of Pennsylvania, Philadelphia, PA, USA 5 (2002): 2532-2558.

[7] Pucher, John. "Public transportation." (2004).
[8] Ozbay, Serkan, and Ergun Ercelebi. "Automatic vehicle identification by plate recognition." World Academy of Science, Engineering and Technology 9.41 (2005): 222-225.
[9] Niknam, Mehdi, and Parimala Thulasiraman. "LPR: A bio-inspired intelligent learning path recommendation system based on meaningful learning theory." Education and Information Technologies 25.5 (2020): 3797-3819.
[10] Redmon, Joseph, et al. "You only look once: Unified, real-time object detection." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.
[11] Van Rossum, Guido, and Fred L. Drake Jr. Python tutorial. Vol. 620. Amsterdam: Centrum voor Wiskundeen Informatica, 1995.
[12] De Smedt, Tom, and Walter Daelemans. "Pattern for python." The Journal of Machine Learning Research 13.1 (2012): 2063-2067.
[13] Lutz, Mark. Programming python. " O'Reilly Media, Inc.", 2001.
[14] Niknam, Mehdi, and Parimala Thulasiraman. "LPR: A bio-inspired intelligent learning path recommendation system based on meaningful learning theory." Education and Information Technologies 25.5 (2020): 3797-3819.
[15] Godefroid, Patrice, Nils Klarlund, and Koushik Sen. "DART: Directed automated random testing." Proceedings of the 2005 ACM SIGPLAN conference on Programming language design and implementation. 2005.