VGU Machine learning project: license plate detection

Hoang-Tam Thai
CSE,Vietnamese-German University
Thu Dau Mot City, Vietnam
17387@student.vgu.edu.vn

Minh-Hoang Dinh
CSE,Vietnamese-German University
Thu Dau Mot City, Vietnam
17124@student.vgu.edu.vn

Minh-Triet Huynh
CSE,Vietnamese-German University
Thu Dau Mot City, Vietnam
17447@student.vgu.edu.vn

***Abstract*—Traffic and the control of traffic flow has always relied on big amount of data and data processing [1]. To aid this management of vehicle the process of vehicle registration categorization is very important. This project aims is to provide a cheap and easy to use program that can take an image or video and output the province the vehicle was originally registered. The result will be overlayed over the original image.**

I. Introduction

A. Problem Statement

Saigon is a diverse city with many people from all over Vietnam flocking to it every year, and with that comes a lot of traffic. This doesn't help that there are more than 1000 vehicles in Saigon registered everyday according to [2] and by Decembers 2021 the number of new car manufactured alone is more than 200 Millions [3]

This result in a diverse set of license plates can be found in the city and the surrounding area.

Resulting in the fact that Saigon has been ranked level 6, the highest level possible, for the severity of traffic jam in Vietnam [4], which means that there is a rising need to keep track of the vehicle that are operating within Saigon, the purpose of this is to create surveys or statistics for the relevant governmental bureaus or management bodies.

The traditional way of doing this is to look in the official records of the license plate registered in the city and deduct the actual statistic. However, there are many problems with this method. First, the official records are not always accurate in the sense that the number of bikes can move in and out of the city dynamically. Second, the official records are not always up to date.

This requires that a new, cheap, fast method to keep track of the vehicle in the city is needed.

II. Application

The project can be run in three mode:

- Real time mode
- website mode
- website with Real-ESRGAN mode [5]

In order to tackle the problem of vehicle registration detection on the road real time mode will be used to detect and recognize the licence plate on the fly. In the case where the licence plate is too far away or the quality of the image is too small then the website with Real-ESRGAN mode can then be use which will upscale the image before OCR recoginition.

III. Objective

The objective is to create something that is relatively cheap and easy to operate in real-time to keep track of the vehicle's origin in the city. For example the user can use their main camera of their phone to scan the license plate of the target vehicle, and it would display on screen where was the vehicle registered.

In addition the project would also capable of handling hard edge cases where the is too blurry or the license plate is too far from the camera which took the image by using Real-ESRGAN to upscale the image [5]

IV. System Design

The project pipeline includes the following stages:

- Training and Validation Data (Images and Labels) are stored on Google Drive.
- The model is trained on Google Colab with data source from Google Drive.
- The model is now ready, Images/Videos can be used as input for the trained detection model.
- The model detects license plates and crops them into frames
- Cropped frames can be upscaled with Real-ERSGAN first or directly go through EasyOCR to recognize license plates texts.
- Recognized texts are mapped to respective province of Vietnam.
- The output image can be shown either in real time or on web.

A. Data collection

The training of the data model get its training data from a google drive which now in total has 4000 images. The model, Yolov5, reads from this folder and produce three models:

- small
- medium
- large

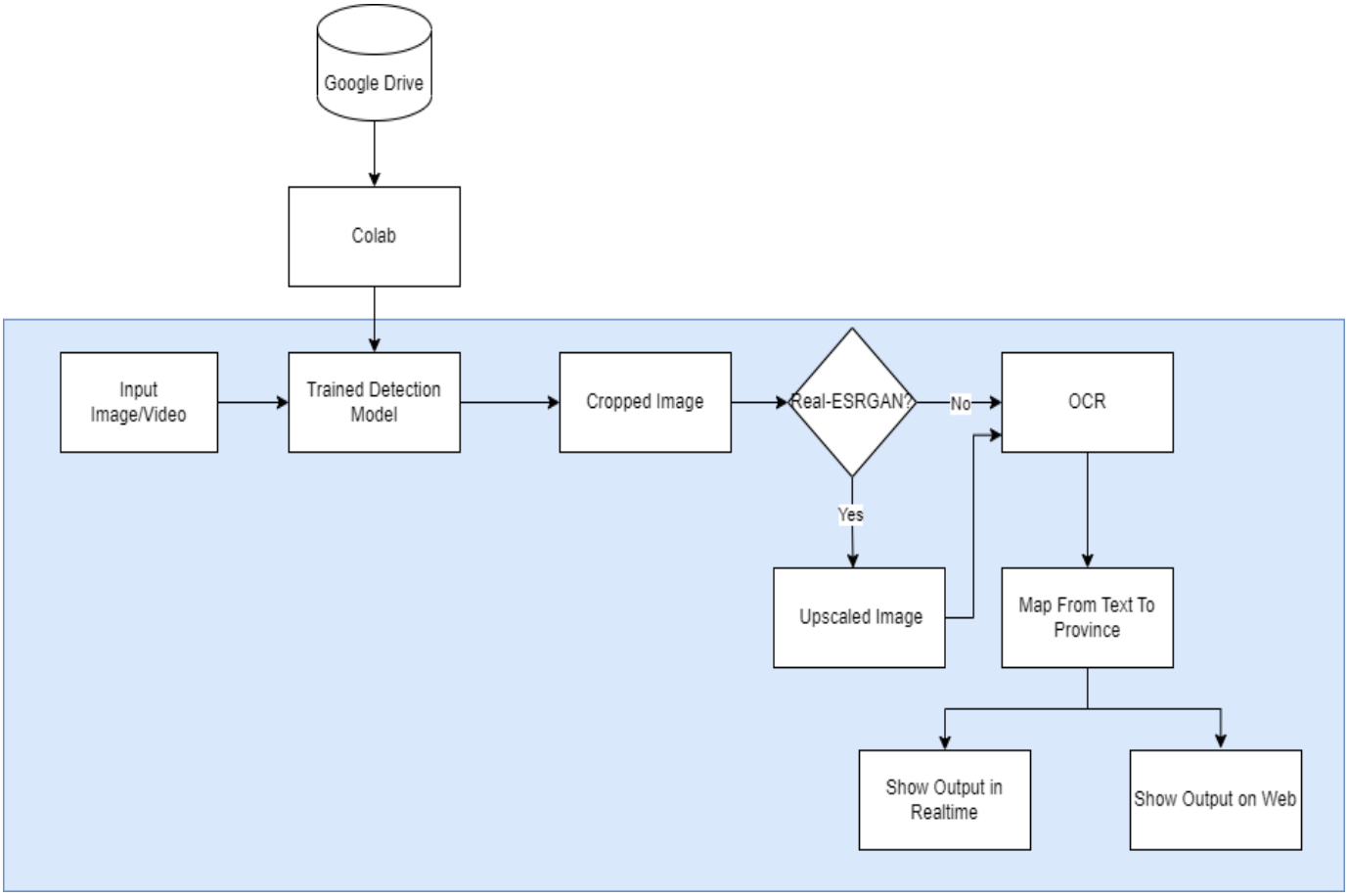

Fig. 1: Project's pipeline.

V. Challenges

A. Python-tesseract as OCR

One alternative to EasyOCR is Python-tesseract. Python-tesseract is an optical character recognition (OCR) tool for python. It will recognize and "read" the text embedded in images.

Python-tesseract can only run on CPU, while EasyOCR can run on either CPU or GPU. OCR runs considerably faster on GPU than CPU, so Python-tesseract's execute time is much longer than EasyOCR.

Furthermore, Python-tesseract is more suitable to regconize text in a paper-like document.

Below is a demonstration comparing Python-tesseract and EasyOCR:

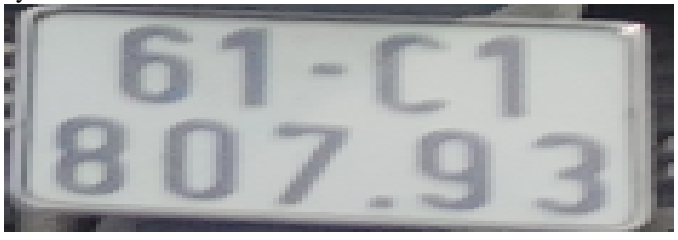

Fig. 2: Comparison between the two OCR methods
Python-tesseract: {61 1807.93 EasyOCR: 61CT 807.93

B. Upscale cropped license plates images with Real-ESRGAN

To further improve the success of recognizing license plates numbers/characters with OCR, we have tried to upscale the license plates frame with Real-ESRGAN. Real-ESRGAN aims at developing Practical Algorithms for General Image/Video Restoration.[5]

Validation image is processed by the model, then the model detects and crops the license plates. Each cropped license plates are then upscaled with Real-ESRGAN before EasyOCR recognizes text.

Below is a demonstration of Real-ESRGAN: The first picture is before upscale with Real-ESRGAN, the second picture is after upscale with Real-ESRGAN.

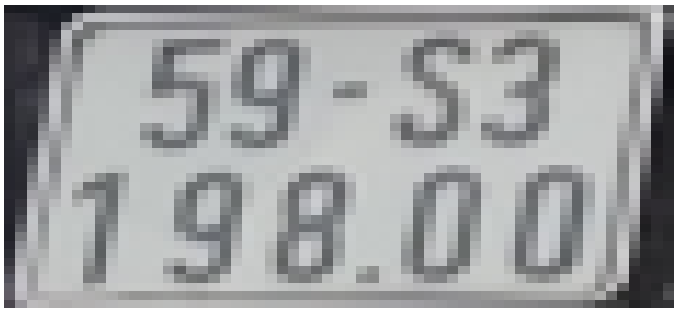

Fig. 3: Before Upscale

EasyOCR's Result: 59-53 7198.00

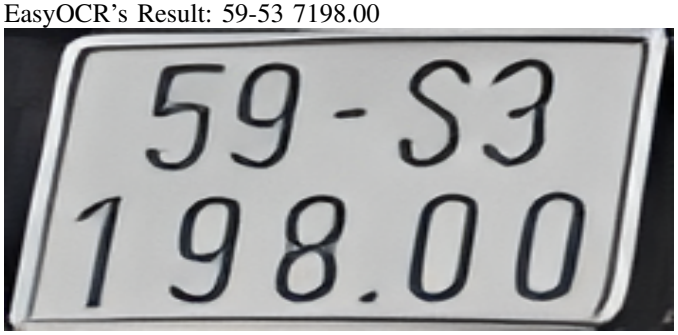

Fig. 4: After Upscale

EasyOCR's Result: 59-S3 198.00

VI. Data collection and Processing

A. Tools

1) labelImg: The software we use to label images is labelImg. It can be downloaded as a standalone software running on Windows/Linux or it can be downloaded via pip. The command to download via pip is:

```
pip install labelImg
```

We choose labelImg because of three main reasons. First is because it uses Python, which is the language chosen for the whole project, so no other installation is required. The second reason is that it is easy to install, intuitive to use, and it is lightweight. The final reason which is also an important one is that it can convert different label format to Yolov5 format. We can select an image, and click the label format button to switch between the format, and then we can save the label. The example is showed in part VI-C1.

2) pyautogui: This is a Python library that can control the cursor to click at specific place on the screen. This library is used to automate the process of changing label format (from xml to txt) for data received from Kaggle. More detail here VI-C1.

B. Data source

Since every machine learning model performance relies heavily on the data that it has been trained on, generally the more the better. However most licence plate is stored in the data bank of the Ministry of transportation so we need to obtain our own source of data.

The solution to this problem is that we obtain our data from three main source:

- Kaggle: 433
- various Vietnamese machine learning forums: 3692
- Picture taken on the road: 17
- Picture taken from various websites: 20

Total: 4162, of which 4000 are used for training, 148 are used for validation, and the remaining are used for testing. link to google drive folder of the images: source

C. Data processing

1) Kaggle data format: The first step of the pipeline is to detect all of the license plate located on the image using Yolov5. Initially we only manage to obtain 433 images from Kaggle which is in xml format, however Yolov5 only accept data in a txt format. In order to convert these 433 images labelImg has an option to do so however it exist as a GUI element and does not expose any CLI options that we could use to run on the terminal. Therefore to automate the task we wrote a python script using Pyautogui library to go and click the relevant buttons to transform the bounding boxes to the correct format.

```
while (true):
    keyboard.press('d')
    pyautogui.click(x=57, y=519)
    keyboard.press('␣')
```

2) Vietnamese data collection: The kaggle data mentioned above came with labelling however most of the data is taken in the US and not Vietnam, mostly license plate of cars and not motorbikes. Even then most of the US license plates are one row plates and not two rows, for example:

Fig. 5: US licence plate

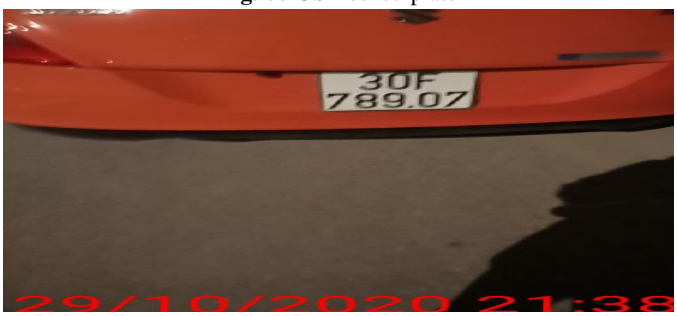

Fig. 6: Vietnamese licence plate

In which the above is an example of a US car and below of a Vietnamese car, even though some Vietnamese car can also have some one row variation the majority of the vehicles in Vietnam most of which are motorbikes have two rows of

characters.

Hence that is why we want not only one row licence plate but also two rows to train the model to detect Vietnamese car. In addition Vietnamese plate has colors to indicate the different types of vehicle registration, whereas license plate in the US does not.

Both of these factors above show that the need for localize data in Vietnam grows more significant. That is why we have gathered from various forum or months pictures of cars and motorbikes in Vietnam.

we have divided the work of labelling to three members of our team and managed to label 932 images.

So with the new labelled images we train an intermediate model to recognize the next 300 images and then sort out the false positive images using a bash script, see more in VI-C3. the process on first with 300 images each. Then by the time we have managed to train a model with a data set of 1700 images the model is then used to train the remaining 2300 images.

During the process many images that are detected by the model during the early stages has many false positive cases, for example:

Fig. 7: False positive

Members of our team will go in and and remove the bounding boxes that are not actually licence plate.

3) False positive: Since we know that most of the pictures in the Vietnamese training data set is from parking lot, meaning each picture should have exactly one licence plate only. So we create a simple bash script that just goes through every file that has been recognized by the intermediate Yolov5 model and copy all of the files that has more than 1 label or no label at all.

What the script does is to separate the incorrect label images and their corresponding label into one folder, which makes it easier for our team member to then go over each one and correct them by hand.

The script source code can be found: here

VII. Experiments

A. Evaluation Metrics

The metrics used to evaluate (validate) the differences between different models are:

- The average and the best mAP (mean Average Precision) of a model
- The number of license detected in an image
- the number of license falsely detected in an image
- Average time to detect licenses in an image

The metrics above are chosen because it give a general idea on how well the model perform. It shows the accuracy, and the efficiency of the model.

B. Benchmark Data

The validation data is categorized into 2 group: simple and hard. Each image in simple group contains a small amount of license plates and they are usually easy to detect. Meanwhile each image in hard group contains a large amount of license plates and they are usually hard to detect.

There are a total of 23 images, in which 18 images are in simple category, and 5 images are in hard category.

C. Results

The result is acquired from the result.csv file after the model has been trained. The mAP values in the table below is taken from metrics/mAP_0.5 data.
Every model has 4000 train images and 148 validation images.

	Average mAP	Best mAP
Small - Yolov5s	0.7452764	0.78473
Medium - Yolov5m	0.7263648	0.76881
Large - Yolov5l	0.7714684	0.80372

1) Average and best mAP by model's weight:
2) Simple Category: Refer to table I
3) Hard Category: Refer to table II

D. Summary

- Model Small-Yolov5s struggles with images where license plates are hard to detect. It is suitable to run on low-end hardware and detection image is simple.
- Model Medium-Yolov5m has low false detection but also low license plates detection percentage. It is suitable when false detection of license plates is not tolerable.
- Model Large-Yolov5l sometimes struggles with images where license plates are easy to detect. It is suitable to run on high-end hardware and detection image is complex.

VIII. Conclusion

The project has achieved its two main goal: license plate detection and then text recognition the license plate to map the number to the province.

The entire pipeline works using Yolov5 to detect and crop out the lisence plate from the original image. Then it will go through an upscaling model Real-ESRGAN to better enhance the image (optional) and finally the image will be display on the monitor of the device that originally captured the image.

The project app operates in three main mode:

- Real time mode
- Website mode
- Website with Real-ESRGAN [5] mode

Finally the final result is to overlay the Province of the vehicle over the orginal image with the bounding box of the image included.

Index	Licenses in image	Model: Small		Model: Medium		Model: Large	
		Detected Licenses	Falsely Detected Licenses	Detected Licenses	Falsely Detected Licenses	Detected Licenses	Falsely Detected Licenses
1	1	1	0	1	0	1	0
2	1	1	0	1	0	1	0
3	1	1	0	1	0	1	0
4	1	1	0	0	0	0	0
5	1	1	0	1	0	1	0
6	1	1	0	1	0	1	0
7	1	1	0	1	0	1	0
8	2	1	0	1	0	1	0
9	1	1	0	1	0	1	0
10	1	1	0	1	0	1	0
11	1	1	0	1	0	1	0
12	2	2	0	2	0	2	0
13	1	1	0	1	0	1	0
14	2	2	1	1	1	2	1
15	1	1	0	1	0	1	0
16	4	4	0	3	0	4	0
17	1	1	0	1	0	1	0
18	1	1	0	1	0	1	0
Total	28	23	1	20	1	22	1

TABLE I: benchmark table for simple validation images

Index	Licenses in image	Model: Small		Model: Medium		Model: Large	
		Detected Licenses	Falsely Detected Licenses	Detected Licenses	Falsely Detected Licenses	Detected Licenses	Falsely Detected Licenses
1	3	3	0	3	0	3	0
2	4	4	1	3	0	4	2
3	5	5	2	4	0	5	0
4	8	4	0	3	0	4	0
5	8	3	0	4	0	4	1
Total	28	19	3	17	0	20	3

TABLE II: Benchmark table for hard validation case

REFERENCES

[1] A. I. Torre-Bastida, J. Del Ser, I. Laña, M. Ilardia, M. N. Bilbao, and S. Campos-Cordobés, "Big data for transportation and mobility: recent advances, trends and challenges," *IET Intelligent Transport Systems*, vol. 12, no. 8, pp. 742–755, 2018. [Online]. Available: https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/iet-its.2018.5188

[2] "Tp.hcm: Moi ngay hon 1.000 phuong tien dang ky moi." [Online]. Available: https://www.baogiaothong.vn/tphcmmoingayhon1000phuongtiendangkymoid568127.html

[3] "Vietnam motor vehicle sales: Passenger cars, 2005 – 2022 | ceic data." [Online]. Available: https://www.ceicdata.com/en/indicator/vietnam/motor-vehicle-sales-passenger-cars

[4] "Qua tai giao thong o tphcm da o nguong nguy hiem." [Online]. Available: https://tienphong.vn/quataigiaothongotphcmdaonguongnguyhiempost1473821.tpo

[5] "xinntao/real-esrgan: Real-esrgan aims at developing practical algorithms for general image/video restoration." [Online]. Available: https://github.com/xinntao/Real-ESRGAN#online-inference