

CAN THO UNIVERSITY
COLLEGE OF INFORMATION AND COMMUNICATION TECHNOLOGY



SPECIALIZED TOPIC IN
INFORMATION TECHNOLOGY
(HIGH-QUALITY PROGRAM)

BUILDING MODEL TO RECOGNIZE
VIETNAMESE LICENSE PLATES BASED ON
YOLOV5 MODEL

Student: Ho Xuan Phuong Dong

Student ID: B1910628

Class: 2019-2023 (K45)

Advisor: Dr. Tran Cong An

Can Tho, 11/2022

CAN THO UNIVERSITY
COLLEGE OF INFORMATION AND COMMUNICATION TECHNOLOGY
DEPARTMENT OF INFORMATION TECHNOLOGY



SPECIALIZED TOPIC IN
INFORMATION TECHNOLOGY
(HIGH-QUALITY PROGRAM)

BUILDING MODEL TO RECOGNIZE
VIETNAMESE LICENSE PLATES BASED ON
YOLOV5 MODEL

Student: Ho Xuan Phuong Dong

Student ID: B1910628

Class: 2019-2023 (K45)

Advisor: Dr. Tran Cong An

Can Tho, 11/2022

***Topic:** Building model to recognize Vietnamese license plates based on YOLOv5 model*

ACKNOWLEDGMENTS

First of all, I would like to express my gratitude to Dr. Tran Cong An for suggesting the topic as well as guiding me to complete this topic. He spent time and created favorable conditions in all aspects so that I could complete the topic.

Many thanks to my friends who helped me about technical issue when I had problems on my topic.

Because there are many limitations in my knowledge, my topic will certainly be difficult to avoid mistakes. I look forward to receiving comments and suggestions from teacher and classmates to help me improve the topic.

Thank you sincerely!

Can Tho, November 18, 2022
Student

Ho Xuan Phuong Dong

TABLE OF CONTENT

ACKNOWLEDGMENTS.....	i
TABLE OF CONTENT	ii
LIST OF FIGURES.....	iv
LIST OF TABLES	iv
LIST OF ABBREVIATIONS	iv
ABSTRACT	v
TÓM TẮT	vi
INTRODUCTION.....	1
Statement of the problem.....	1
The purpose of the topic	1
Object and Scope	1
General approach	1
Fundamental topic outline	2
CONTENT	3
CHAPTER 1. REQUIREMENTS SPECIFICATION	3
CHAPTER 2. LITERATURE REVIEW.....	4
2.1. Machine Learning	4
2.2. Computer Vision.....	4
2.3. YOLO	5
2.3.1. History of YOLO	5
2.3.2. YOLO Network Architecture.....	5
2.3.3. The principle of operation of the YOLO network	6
2.4. YOLOv5	6
2.4.1. Definition	6
2.4.2. Versions of YOLOv5	8
2.4.3. The distinct advantages of the YOLOv5 model	8
2.5. Related terms	9
2.5.1. Bounding box.....	9
2.5.2. Intersection on Union.....	9
2.5.3. Anchor box.....	10
2.5.4. Non-max suppression.....	11
2.5.5. Precision.....	12
2.5.6. Recall	12

2.5.7. Mean Average Precision	12
2.5.8. Calculate Precision and Recall.....	12
CHAPTER 3. IMPLEMENTATION	14
3.1. Training Model	14
3.1.1. Preparing dataset	14
3.1.2. Labelling dataset	14
3.1.3. Training model	15
3.1.3.1. Algorithm settings	15
3.1.3.2. Training Results	17
3.2. Building Algorithm to arrange recognized objects in the correct order	20
CHAPTER 4. TESTING AND EVALUATION	25
4.1. Testing	25
4.2. Evaluation	27
CONCLUSION	28
Result and Discussion	28
Future Works	28
REFERENCES	29
APPENDICES	30

LIST OF FIGURES

Figure 1. Example of detected object sorting algorithm	3
Figure 2. YOLO Network Architecture [3]	6
Figure 3. YOLOv5 Network Architecture [4]	7
Figure 4. Versions of YOLOv5 [5]	8
Figure 5. Formula calculate IoU [6]	10
Figure 6. Defining the anchor box for an object [3]	11
Figure 7. The directory structure of YOLOv5	14
Figure 8. .txt file of labelled image	15
Figure 9. .yaml file	16
Figure 10. Structure of .zip file	16
Figure 11. Validation result of training model	19
Figure 12. Confusion matrix of the training model	20
Figure 13. Determine coordinates of object	21
Figure 14. Remove overlapped object	21
Figure 15. Find the Y-average of Ymin and Ymax	22
Figure 16. Determine the plate has 1 or 2 rows	22
Figure 17. Split origin array into 2 arrays	23
Figure 18. Sort array by ascending order of X-axis	23
Figure 19. Merge 2 array after being sorted	24
Figure 20. Testing result 1	25
Figure 21. Testing result 2	25
Figure 22. Testing result 3	26
Figure 23. Testing result 4 with algorithm	27

LIST OF TABLES

Table 1. Model training results	18
---------------------------------------	----

LIST OF ABBREVIATIONS

No.	Abbreviation	Origin word
1	YOLO	You Only Look Once
2	CNN	Convolutional Neural Network
3	2D	2-Dimension

ABSTRACT

Today, with the development of technology and science, machine learning is applied more and more in many aspects of life from healthcare, education, tourism, security, sports,... Image processing and recognition is an area of computer vision. Machine learning is applied to solve dangerous jobs or used to reduce the amount of work that humans have to handle. With the application of machine learning, work can be processed faster and more accurately when compared to the same work done by human effort. Take for example security, with machine learning being able to automatically detect intruders and issue alerts without always having someone in person.

Machine learning is also applied in the field of traffic, such as detecting speeding or recording traffic violations. Usually, violations will be recorded through the control plate - the license plate of the vehicle. Security issues in parking areas are also ensured through the registration of license plates. Within the scope of this topic, I would like to present on the topic of identifying objects that are alphanumeric characters appearing on Vietnamese license plates and algorithms to arrange objects according to the correct position of appearance.

TÓM TẮT

Ngày nay, với sự phát triển của công nghệ và khoa học, máy học được ứng dụng ngày càng nhiều vào trong nhiều khía cạnh của cuộc sống từ y tế, giáo dục, du lịch, an ninh, thể thao,... Xử lý và nhận dạng ảnh là một lĩnh vực thuộc về thị giác máy tính. Máy học được ứng dụng để giải quyết những công việc mang tính nguy hiểm hay được ứng dụng để giảm bớt khối lượng công việc mà con người phải xử lý. Với việc ứng dụng máy học, công việc có thể được xử lý nhanh và chính xác hơn khi so với cùng một công việc được thực hiện bởi sức người. Ví dụ như vấn đề về an ninh, với máy học có thể giúp tự động phát hiện người lạ xâm nhập và phát cảnh báo mà không cần lúc nào cũng phải luôn có người nào đó trực trực tiếp.

Máy học cũng được ứng dụng vào trong lĩnh vực giao thông, như việc phát hiện vượt quá tốc độ hay ghi nhận các phương tiện vi phạm giao thông. Thông thường các trường hợp vi phạm sẽ được ghi nhận thông qua biển kiểm soát – biển số xe của phương tiện. Vấn đề an ninh ở các khu trông giữ xe cũng được đảm bảo hơn thông qua ghi nhận biển số xe. Trong phạm vi đề tài này, em xin trình bày về đề tài nhận dạng các đối tượng là các ký tự chữ và số xuất hiện trên biển số xe Việt Nam và thuật toán để sắp xếp các đối tượng theo đúng vị trí xuất hiện.

INTRODUCTION

Statement of the problem

With the development of socio-economic lead to the explosion of means of transport, causing difficulties in the management and handling of violations for people. The automatic recognition of license plates is not a new problem in the field of traffic and security.

As the paper An Approach for Building an Intelligent Parking Support System [1], this paper use cascade of Boosting with Haar-like features and Support Vector Machines to identify the license plate and the characters appearing on it.

In this topic, I use object detection algorithm YOLOv5 model to build the model to recognize letters and numbers in the Vietnamese license plates.

The purpose of the topic

The purpose of the topic is with:

- Building a test application for the trained model to recognize Vietnamese license plate.
- The test application can accurately identify license plates with 1 line and 2 lines.

Object and Scope

The following is the objects that the topic is aimed at:

- YOLOv5 model.
- 10 numbers (0 – 9) and 20 letters in motorbike and car Vietnamese license plate.
- Android application.
- Algorithm to arrange recognized objects in the correct order.

The scope of the topic is building a test application for the trained model to recognize Vietnamese license plate.

General approach

Research to problem-solving theory:

- Learning about YOLOv5 model
- Training the model with letters and numbers appear in motorbike and car Vietnamese license plate

Topic: Building model to recognize Vietnamese license plates based on YOLOv5 model

- Building test application for the model in smart phone

Fundamental topic outline

The remainder of this topic is structured as follows:

CONTENT

Chapter 1. Requirement Specification

Chapter 2. Literature Review

Chapter 3. Implementation

Chapter 4. Testing and Evaluation

CONCLUSION

1. Result and Discussion

2. Future Works

CONTENT

CHAPTER 1. REQUIREMENTS SPECIFICATION

Image-based object recognition is the main area that the YOLOv5 model has focused on. Using this model it is possible to identify the position as well as the class of the object defined on that image.

The problem is that YOLOv5 does not sort the detected objects on an input image in a particular order. That directly affects the returned results when identifying a specific number plate and the returned results are the characters appearing on that number plate in the correct order specified from left to right and from top to bottom.

Therefore, after training the model to detect the number plate characters, it is necessary to build an algorithm that helps to arrange the recognizable objects in a certain order to get can return an exact result of the input plate number.

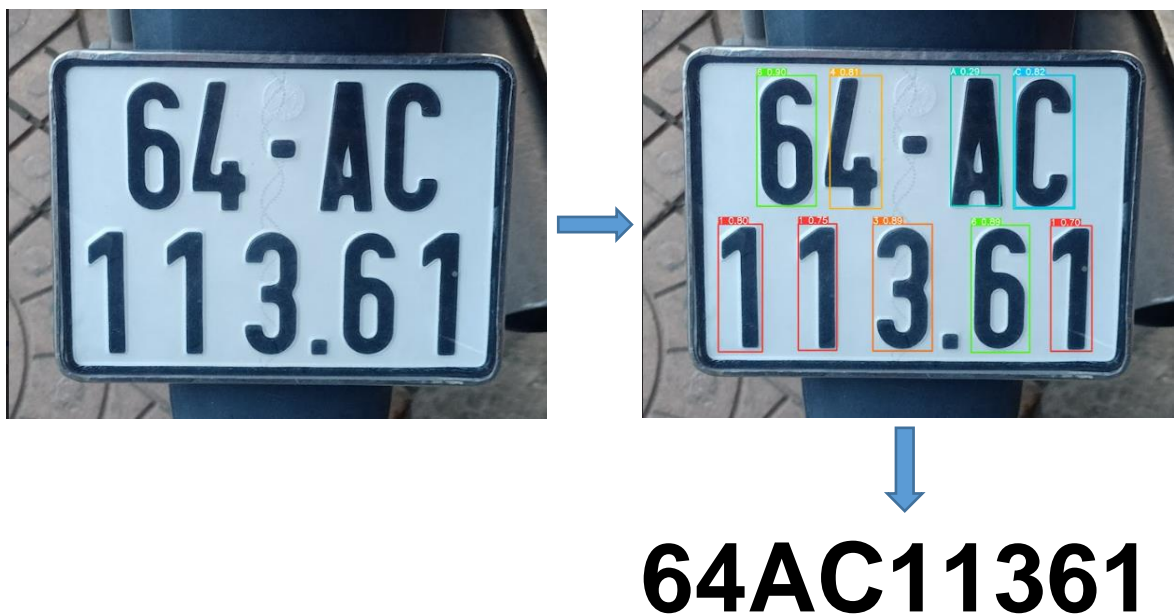


Figure 1. Example of detected object sorting algorithm

CHAPTER 2. LITERATURE REVIEW

2.1. Machine Learning

Machine learning is an area of Artificial Intelligence that deals with the study and construction of algorithms, or techniques that allow systems to learn automatically from the dataset to solve specific problems. For example, if we have some picture of a chair, or a table, and we “teach” computer that these picture is about a the chair or table, then give a picture that has never been seen, it can still guess whether it is a chair or a table. In Computer Vision, Machine Learning is the final step after processing the input image into data. The returned result is the class (label) of the input image. Such as physical image recognition, license plates, traffic signs, etc.

In general, a Machine Learning application has the following basic steps:

- Collect data (number plates, faces, cats and dogs, ...)
- Extract necessary information from data
- Training to create a model to use
- Programming the application using the trained model

2.2. Computer Vision

Object recognition is a general term to describe a set of computer vision tasks that are related to the identification of objects in digital images.

Image classification is concerned with predicting the class of an object in an image. Object localization refers to determining the position of one or more objects in an image and drawing a bounding box around that object. Object detection combines the above two tasks and performs for one or more objects in the image. We can distinguish between these three basic computer vision tasks by their input and output as follows:

- Image classification: predict the label of an object in an image
 - Input: an image with an object, such as a photograph
 - Output: class label
- Object Localization: Determines the presence of objects in the image and indicates their position using the bounding box
 - Input: an image with one or more objects, such as a photograph
 - Output: one or more bounding boxes defined by negative coordinates, width and height

- Object detection: locate the presence of objects in the bounding box and the labels of objects within an image.
 - Input: an image with one or more objects, such as a photograph
 - Output: one or more bounding boxes and labels for each bounding box

The difference in image classification models compared to Object recognition is that the image classification model has a loss function based only on the error between the forecast label and the actual label while Object detection evaluates based on the error between the forecast label and forecast frame error compared to reality.

2.3. YOLO

2.3.1. History of YOLO

YOLO (You Only Look Once) is an algorithm of the CNN (Convolutional Neural Network) network model for object detection and classification. YOLO is created by combining convolutional layers and connected layers. Where the convolutional layers will extract the features of the image, and the full-connected layer will predict the probability and coordinates of the object. YOLO may not be the best algorithm but it is one of the fastest in the class of object detection models. It can achieve almost real-time speed without losing too much accuracy compared to other models. As its name suggests, YOLO offers the advantage that it only uses the information of the entire image once and relies on guessing the entire object box containing the objects. [2]

YOLO is an algorithm used in object detection, so the goal of the YOLO model is not only to predict the label for the object, but also to determine the location of the object. Therefore, YOLO can detect many objects with different labels in an image instead of only classifying a single label for an image. Up to now, YOLO has released different versions and improved over the previous version: YOLO (2015), YOLOv2 (or YOLO9000 – 2016), YOLOv3 (2018), YOLOv4 (April 2020), YOLOv5 (May 2020), MT-YOLOv6 (2022 – however, MT-YOLOv6 is not part of the official YOLO series), and YOLOv7 (July 2022).

2.3.2. YOLO Network Architecture

The YOLO network architecture includes: the base network is a convolution network that performs feature extraction. The back part is the Extra Layers applied to detect objects on the feature map of the base network. YOLO's base network uses mainly convolutional layers and fully connected layers. YOLO architectures are also quite diverse and can be customized into versions for many different types of input.

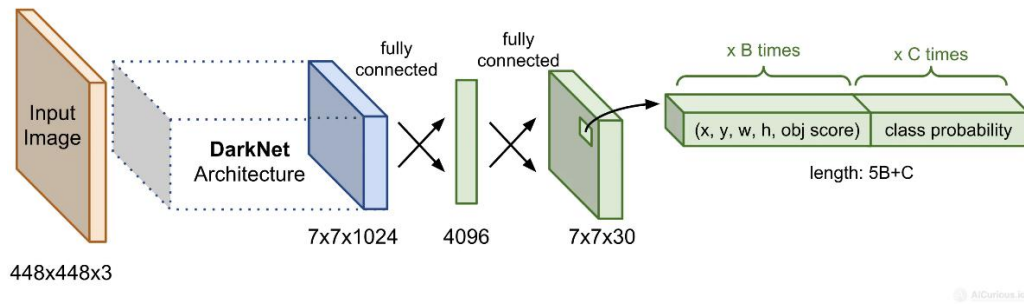


Figure 2. YOLO Network Architecture [3]

The DarkNet Architecture component called the base network is used for feature extraction. The output of the base network is a 7x7x1024 feature map that will be used as input for the Extra layers that predict the object's label and bounding box coordinates. The images, when fed into the model, will be scaled to the same size as the input shape of the model and then collected into a batch for training.

2.3.3. The principle of operation of the YOLO network

The input of the YOLO model will be an image, the model will recognize that image to see if any objects are detected. If so, the model then determines the coordinates of the object in the image. The input image is divided into SxS cells, usually 3x3, 7x7, 9x9, etc. The subdivision affects the object detection of the model [2]. Grid cells predict all bounding boxes and give each cell a confidence score to determine the accuracy of each prediction.

Each cell contains a set of information that the model must predict. The model will predict the probability that the object is in the bounding boxes around each of these sub-cells. The bounding boxes with a high probability will be kept and used for the task of determining the position of the object in the image.

2.4. YOLOv5

2.4.1. Definition

YOLOv5 is the deep learning based architecture used in this study. This model is considered an extension of YOLOv3's PyTorch and is a marketing strategy by Ultralytics to promote the popularity of the YOLO family of object detection models. YOLOv5 algorithm basically also inherits the basic methods of YOLO models, but applies some algorithms to help detect objects quickly, optimize parallel operations to speed up recognition, reduce the model training time optimally. Therefore, it is much faster than the predecessor YOLO models.

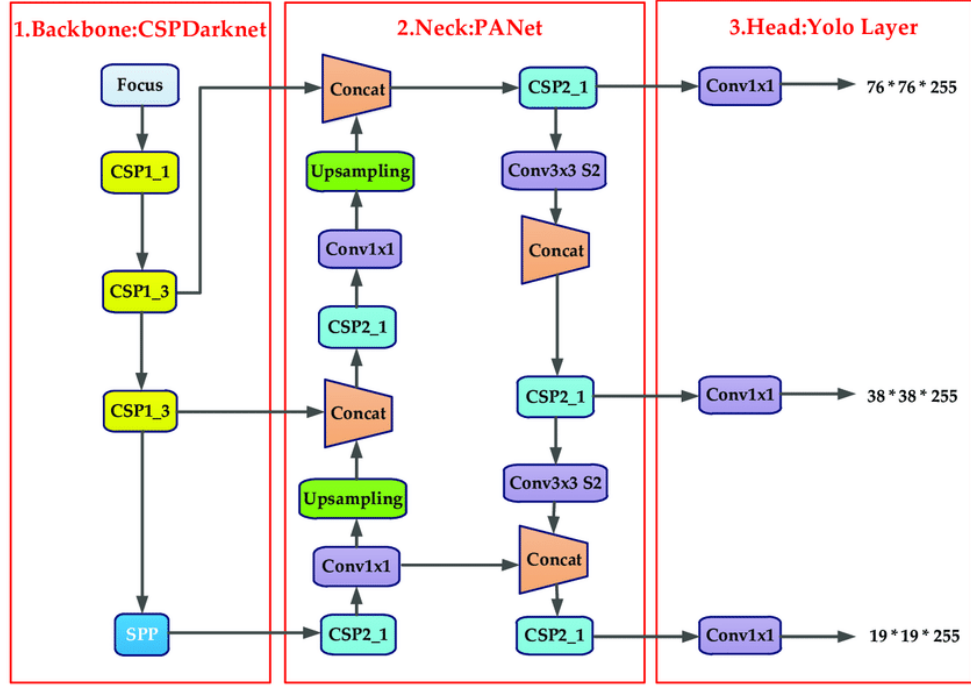


Figure 3. YOLOv5 Network Architecture [4]

The network architecture of YOLOv5 shown in Figure 3 consists of three parts: Backbone part, Neck part and Head part.

YOLOv5 uses Cross Stage Partial Network (CSPNet) into Darknet, creating CSPDarknet as Backbone to extract features from images. CSPNet solves the problem of repetitive gradient information in large-scale backbones by including gradient changes in feature maps, reducing model parameters and floating-point per second (FLOPS) operations, ensuring speed inference and accuracy, while reducing the model size. In recognition problems, the speed of detection and classification as well as accuracy is very important, as it will affect the output of the problem. Besides, the small model size also affects the inference efficiency on resource-constrained devices.

Besides, YOLOv5 also applies Path Aggregation Network (PANet) as the Neck part to promote information flow. PANet helps improve the propagation of low-level features by using a Feature Pyramid Network (FPN) topology with enhanced bottom-up paths. At the same time, adaptive feature aggregation, which connects the feature grid to all feature levels, is used to ensure that information makes sense from each feature level to the next subnet. PANet improves the use of precise localization signals in the lower layers, which can greatly improve the location accuracy of the object.

The Head part of YOLOv5, named YOLO layer, generates 3 different dimensions of the feature map (18 x 18, 36 x 36, 72 x 72) to achieve multi-scale prediction, allowing the model to process handle objects.

2.4.2. Versions of YOLOv5

YOLOv5 offers 4 versions with different network architectures:

Yolov5-s: small version

Yolov5-m: medium version

Yolov5-l: large version

Yolov5-x: extra-large version

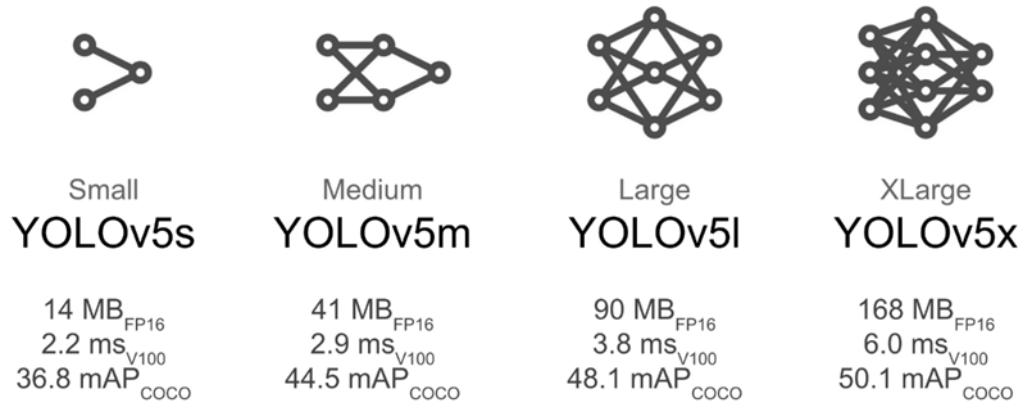


Figure 4. Versions of YOLOv5 [5]

The difference between versions of YOLOv5 is the trade-off between model size and model computation time. The lightweight model version like YOLOv5s is only 14 MB in size, but the accuracy is not as high as the larger versions. On the other hand, the YOLOv5x version of 168 MB is the heaviest model version in terms of size with the highest accuracy of the versions.

2.4.3. The distinct advantages of the YOLOv5 model

YOLOv5 benefits from the established PyTorch ecosystem: support and implementation are both simplified since YOLOv5 was originally developed in PyTorch. The repeatability of YOLOv5 may also become easier for the larger research community because YOLOv5 is a well-known research model.

YOLOv5 has useful components such as hyperparameters, data augmentation techniques and comes with many documents and tutorials that make it easy to refer to this model.

YOLOv5 is small in size, especially the smallest version of YOLOv5 is YOLOv5s which is 14 megabytes in size. This means that YOLOv5 can be deployed on embedded devices more easily.

2.5. Related terms

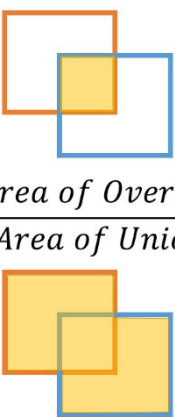
2.5.1. Bounding box

Bounding box is the smallest rectangle that can be drawn around an object to determine its position in the image. This section encloses the entire object to help determine the object's position. There will be 2 types of bounding box:

- Ground-truth bounding box: actual bounding box, is the actual part surrounding the predefined object.
- Predicted bounding box: predicted bounding box, is the part of the predicted bounding box where the position of the object is determined by the algorithm.

2.5.2. Intersection on Union

Intersection over Union (IoU), also known as Jaccard Index, is a method used to measure the accuracy of object recognition models on a particular data set. This measurement can evaluate different models such as RCNN, Fast-RCNN, Faster-RCNN or YOLO. This method helps us to decide if the predicted bounding box gives good results by calculating the intersection between the combination of the actual bounding box and the predicted bounding box. To apply IoU to evaluate object recognition algorithms, we need ground-truth bounding boxes and predicted bounding boxes.



The diagram shows two overlapping rectangles. The top rectangle is orange with a blue border, and the bottom rectangle is blue with an orange border. The intersection of the two rectangles is shaded yellow. Below the rectangles, a legend indicates that the blue line represents the 'Prediction' and the orange line represents the 'Ground-truth'.

$$\text{Intersection over Union (IoU)} = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$

Figure 5. Formula calculate IoU [6]

In there:

- The area of overlap: the area of the intersection between the predicted bounding box and the ground-truth bounding box.
- The area of union: the area of the union between the predicted bounding box and the ground-truth bounding box.
- IoU is calculated by dividing the area of overlap by the area of union, in other words, IoU is the quotient of the area of overlap divided by the area of union.
- If the IoU output is greater than 0.5, it can be seen that the accuracy rate is quite good.

2.5.3. Anchor box

In the convolutional neural network model CNN, to find the bounding box for the object, the algorithm will need the anchor boxes as the basis of estimation. These anchor boxes will be predefined and will surround the object with relative precision. Later, the regression bounding box algorithm will refine the anchor box to create a predicted bounding box for the object.

Each object in the training image is distributed about an anchor box. In case there are 2 or more anchor boxes surrounding the object, we will determine the anchor box that has the highest IoU with the ground truth bounding box.

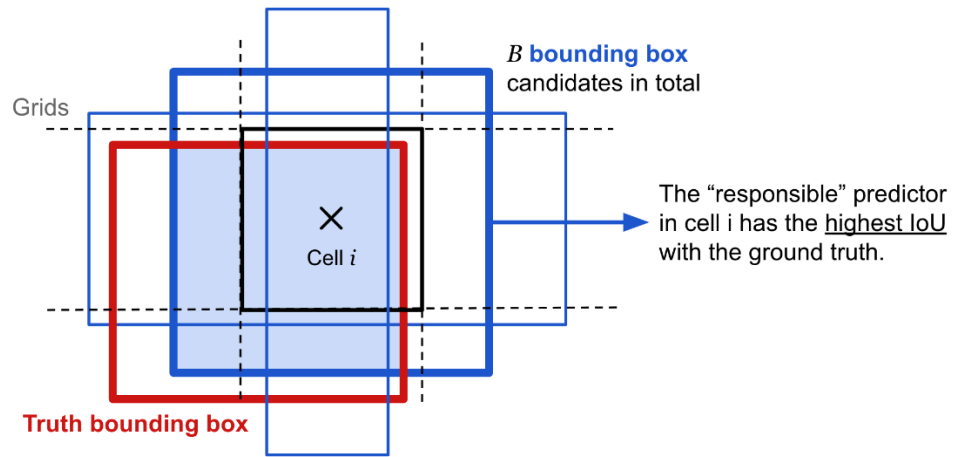


Figure 6. Defining the anchor box for an object [3]

From Cell i we can identify 3 green-bordered anchor boxes as shown in the figure. All three of these anchor boxes intersect with the bounding box of the object. However, only the anchor box with the thickest blue border is selected as the anchor box for the object because it has the highest IoU compared to the ground truth bounding box.

2.5.4. Non-max suppression

One of the most common problems with object detection algorithms is that instead of detecting an object just once, the algorithms can detect the object multiple times. For each object in the image there should be only one bounding box. The non-Max suppression technique solves this problem by selecting the most appropriate bounding box for the object. The way this technique works is as follows:

- The technique will first remove all boxes with probability less than or equal to the predefined threshold. This is to ensure that grids that contain no objects or have a low probability of object occurrence are not shown bounding boxes.
- Then the algorithm will choose the bounding box with the highest probability.
- If there are many bounding boxes with the same highest probability of occurrence of the object, we will remove the IoU, the bounding box with the IoU index lower than the threshold will be discarded. At the end of step three, we will get the bounding box with the best object recognition ratio.

During the test phase, the system will produce many different bounding boxes with different prediction probabilities and different IoU ratios, so the Non-max suppression algorithm helps to eliminate the bounding boxes with low prediction rates. and keep only the last 1 bounding box with the highest prediction rate.

2.5.5. Precision

Precision is the probability that the predicted bounding boxes match the ground-truth bounding boxes, also known as the positive predictor value. The accuracy score ranges from 0 to 1, the higher the accuracy indicates that most of the detected objects match the actual objects.

2.5.6. Recall

Recall use to measure the probability that actual objects are detected correctly. Recall ranges from 0 to 1, where a high recall score means that most of the underlying actual objects were detected.

2.5.7. Mean Average Precision

Precision and Recall are always between 0 and 1. Therefore, Average Precision (AP) is also between 0 and 1. Mean Average Precision (mAP) is the average value of AP. In some cases, the AP for each layer is calculated and averaged to get the mAP. However, they can be the same in different contexts. The mAP mean accuracy score was calculated by averaging AP across all classes and/or total IoU thresholds, depending on detection constraints.

2.5.8. Calculate Precision and Recall

TP (True Positive): the model correctly predicts the class of the object and the frame containing the object predicted by the model (predicted bounding box) and the frame containing the real object (ground-truth bounding box) have a large IoU value more than 0.5

TN (True Negative): the model predicts no objects and actually none

FP (False Positive): the model predicts the wrong class of the object or the model correctly predicts the class of the object, but the frame containing the object predicted by the model and the frame containing the object actually has an IoU value less than 0.5

FN (False Negative): the object is present in the image but the model cannot recognize the object

Formula to calculate Precision:

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

Topic: Building model to recognize Vietnamese license plates based on YOLOv5 model

Formula to calculate Recall:

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

CHAPTER 3. IMPLEMENTATION

3.1. Training Model

3.1.1. Preparing dataset

Collect images of 10 digits on Vietnamese license plate: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.

Collect images of 20 letters on Vietnamese license plate: A, B, C, D, E, F, G, H, K, L, M, N, P, S, T, U, V, X, Y, Z.

Images be collected thanks to Kaggle [7].

All classes are collected based on actual images of license plates of Vietnamese motorbikes and cars, with black characters on a white background.

The dataset about license plates will be divided into 2 groups:

- Train: 1825 images
- Valid: 459 images

Each image contain some classes of the dataset of 10 digits and 20 letters, so the number of images in each class is not equal.

The directory structure of YOLOv5 is different from previous YOLO versions.

Images and labels are divided into 2 separate folders.

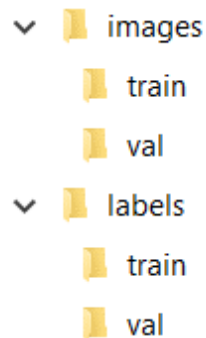
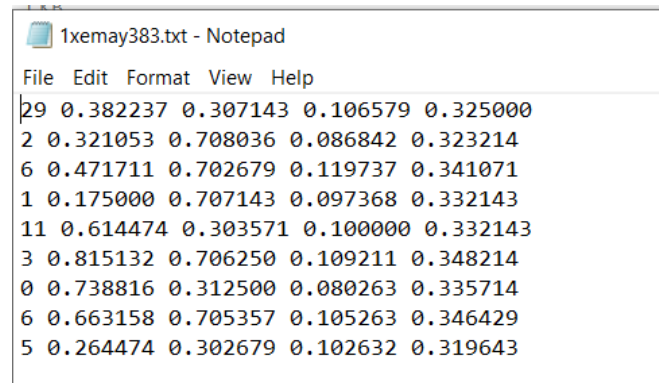


Figure 7. The directory structure of YOLOv5

3.1.2. Labelling dataset

After allocating the images to the images folder, I used the Labeling [8] tool to label for each image in dataset. The information file after labeling will be in .txt format:



```
1xemay383.txt - Notepad
File Edit Format View Help
29 0.382237 0.307143 0.106579 0.325000
2 0.321053 0.708036 0.086842 0.323214
6 0.471711 0.702679 0.119737 0.341071
1 0.175000 0.707143 0.097368 0.332143
11 0.614474 0.303571 0.100000 0.332143
3 0.815132 0.706250 0.109211 0.348214
0 0.738816 0.312500 0.080263 0.335714
6 0.663158 0.705357 0.105263 0.346429
5 0.264474 0.302679 0.102632 0.319643
```

Figure 8. .txt file of labelled image

In there:

Each row corresponds to a labeled object. Each row has a structure:

$\langle \text{class} \rangle \langle x \rangle \langle y \rangle \langle \text{width} \rangle \langle \text{height} \rangle$

- The first column $\langle \text{class} \rangle$ is the ordinal number of the class (from 0 to 29 corresponds to 30 classes)
- The remaining columns mark the coordinates of the bounding boxes that we have labeled. The values of the columns range from 0 to 1 in the format $\langle x \rangle \langle y \rangle \langle \text{width} \rangle \langle \text{height} \rangle$ and are calculated as follows:
 - $\langle x \rangle = \langle \text{coordinates of the center of the object in the x direction} \rangle / \langle \text{width of the shape} \rangle$
 - $\langle y \rangle = \langle \text{coordinates of the center of the object in the x direction} \rangle / \langle \text{height of the figure} \rangle$
 - $\langle \text{width} \rangle = \langle \text{actual width of the frame} \rangle / \langle \text{width of the image} \rangle$
 - $\langle \text{height} \rangle = \langle \text{actual height of the frame} \rangle / \langle \text{height of the image} \rangle$

3.1.3. Training model

3.1.3.1. Algorithm settings

Step 1: Create a file describing the data used for training the model: create a file named `datanumplate.yaml` with the content:

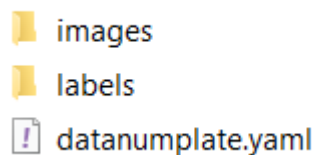
```
! datanumplate.yaml X
D: > CTU_Project > NLCN > newdataset > datanumplate > ! datanumplate.yaml
1  train: datanumplate/images/train
2  val: datanumplate/images/train
3
4
5  nc: 30
6  names: [
7    '1','2','3','4','5','6','7','8','9',
8    'A','B','C','D','E','F','G','H','K',
9    'L','M','N','P','S','T','U','V','X',
10   'Y','Z','0'
11 ]
12
```

Figure 9. .yaml file

In there:

- train: path to the image directory used to train.
- val: path to the image directory used for validating.
- nc: the total number of classes is 30.
- Array of names: array containing the list of 30 class names in the correct order.

Step 2: After have the .yaml file, compress the two images and labels folders that have been processed from the data set preparation step with the .yaml file according to the structure:



```
images
labels
! datanumplate.yaml
```

Figure 10. Structure of .zip file

Step 3: Upload the newly created .zip file to Google Drive and conduct training.

Step 4: Create a notebook with .ipynb format on Google Colab, open the file and make a connection to the Google Colab server.

Step 5: Mount to Google Drive to save the process with the command:

```
from google.colab import drive
drive.mount('/content/drive')
```


Step 6: Clone YOLOv5 from github and install necessary libraries:

```
!git clone https://github.com/ultralytics/yolov5
%cd yolov5
%pip install -qr requirements.txt
%pip install -q roboflow
import torch
import os
```

Step 7: Extract the compressed dataset into yolov5 folder

Step 8: In the yolov5 directory, train the model with the command:

```
!python train.py --img <size> --batch <size> --epochs
<epochs> --data datanumplate/data.yaml --weights
<yolov5*.pt> --cache
```

In there:

- img <size>: specify the input image size (Example: --img 416).
- batch <size>: batch size – how many images to train at one time (Example: --batch 64, here batch size is 64, ie 64 images will be randomly selected for each iteration. in the dataset, repeat until there are no more pictures to end an epoch).
- epochs <epochs>: the number of iterations when all data is included (Example: --epochs 500, training will complete at the end of 500 epochs).
- data datanumplate/data.yaml: set the path to the data.yaml file we just uploaded.
- weights <yolov5*.pt>: specify the type of YOLOv5 model that we use to train (Example: --weights yolov5s.pt)
- cache: use cache to store images for faster model training, with -cache without parameters defaults to using RAM to cache images.

3.1.3.2. Training Results

In this topic, I use YOLOv5s model to training the model to recognize Vietnamese license plates with 10 digits and 20 letters in black characters in white background form.

- The model is trained on Google Colab server with 2 cores, 12.68GB RAM, Tesla T4 GPU.
- Training model with epochs of 500.
- Batch size is 64.
- Image size is 416.

- The model stopped training early as no improvement observed in last 100 epochs. Best results observed at epoch 152, best model saved as best.pt.
- The model completed the training in about 2 hours.

Table 1. Model training results

STT	Classes	Images	Instances	Precision	Recall	mAP@.50
1	1	459	605	0.992	0.982	0.986
2	2	459	319	0.993	0.991	0.995
3	3	459	251	0.994	1	0.995
4	4	459	265	0.986	0.996	0.992
5	5	459	574	0.985	0.993	0.989
6	6	459	271	0.983	0.989	0.99
7	7	459	268	0.981	0.989	0.99
8	8	459	207	0.963	0.99	0.983
9	9	459	407	0.994	0.988	0.991
10	A	459	34	1	0.949	0.991
11	B	459	41	0.951	0.956	0.944
12	C	459	36	0.99	1	0.995
13	D	459	17	0.92	0.941	0.936
14	E	459	13	0.975	1	0.995
15	F	459	65	0.988	1	0.995
16	G	459	26	0.99	1	0.995
17	H	459	33	0.926	1	0.98
18	K	459	16	0.898	0.938	0.986
19	L	459	28	0.987	1	0.995
20	M	459	8	1	0.786	0.909
21	N	459	13	0.95	0.923	0.99
22	P	459	29	0.998	1	0.995
23	S	459	31	0.986	1	0.995
24	T	459	18	0.98	1	0.995
25	U	459	21	0.986	1	0.995
26	V	459	10	0.885	1	0.995
27	X	459	16	0.981	1	0.995
28	Y	459	6	0.913	0.833	0.972
29	Z	459	4	0.91	1	0.995
30	0	459	286	0.994	0.997	0.992
All		459	3918	0.969	0.975	0.985

The results show that the mAP @ .50 value of the whole image with the IOU = 50% threshold is 0.985. The total number of identifiable objects in those 459 valid images was 3918.

Average Accuracy (mAP) is 0.985 against bounding box IoU threshold of 0.50, Recall is 0.975 and Precision is 0.969 as shown below:

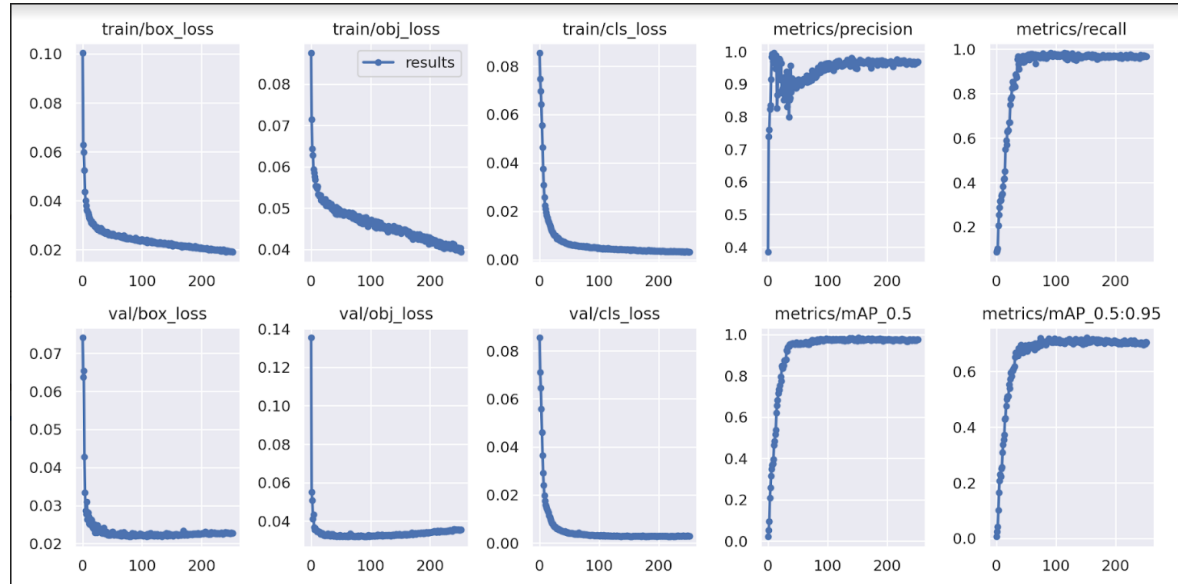


Figure 11. Validation result of training model

box_loss: is the average loss of the bounding envelope, the smaller the ratio, the more accurate it is.

obj_loss: is the average loss of target recognition, the smaller the ratio, the more accurate it is.

cls_loss: is the average value of classification loss, the smaller the ratio, the more accurate it is.

precision: the higher the ratio, the more accurate it is.

recall: the higher the ratio, the more accurate it is.

Through each epoch loop, the ratio of loss functions tends to decrease and the ratio of precision and recall tends to increase, showing that the trained model is getting better and better with each training.

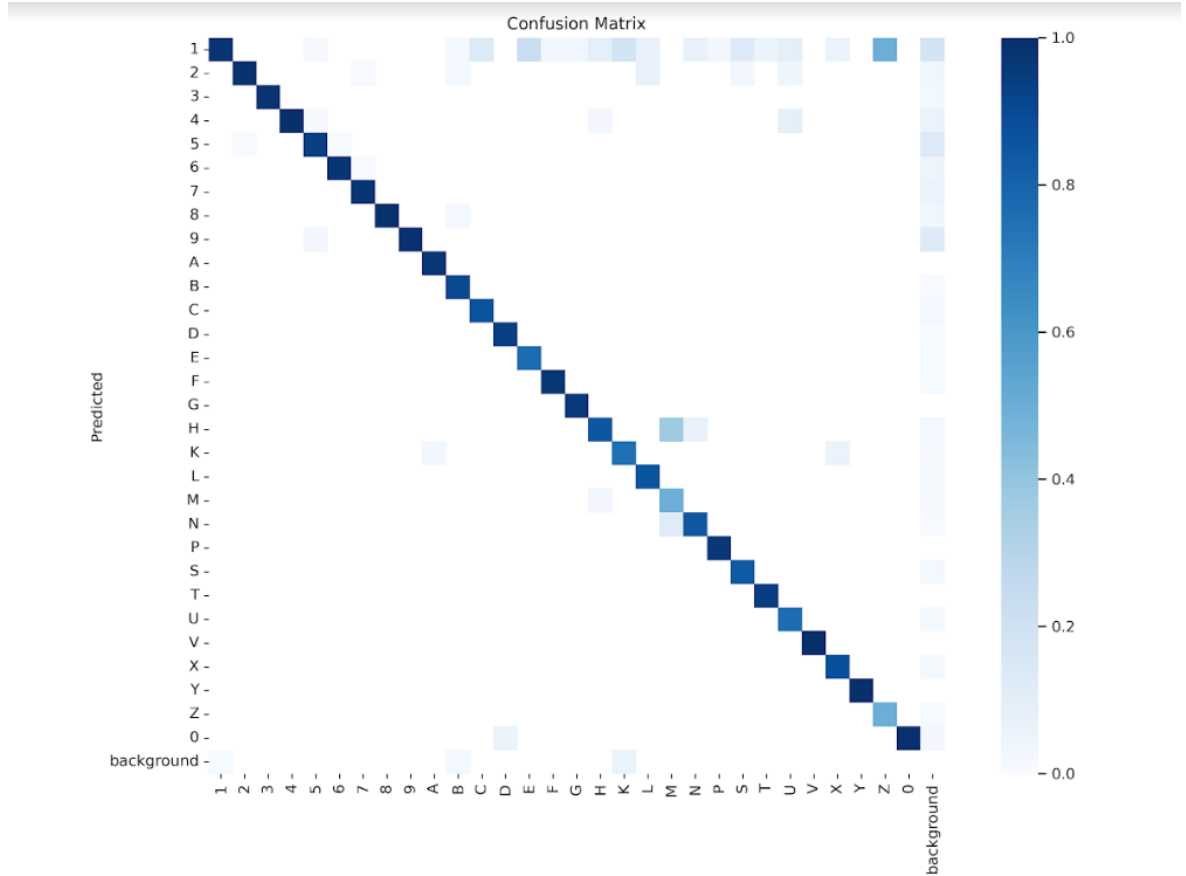


Figure 12. Confusion matrix of the training model

3.2. Building Algorithm to arrange recognized objects in the correct order

The characters on the license plate after being recognized are usually individual objects, not arranged in the correct order in which they appear on the sign (from left to right, from top to bottom). In order to be able to more easily manage the number plate, from the number plate image it needs to be converted to text so that information can be easily stored. Therefore, I would like to propose an algorithm to arrange the recognized objects from the input image into a text string.

Each object, after being recognized and located by the YOLOv5 model in the image, will have attributes such as class name, recognition scale, and position of edges (left, top, right, bottom) to determine define the bounding box. With the algorithm I proposed, I will rely on the position of the predicted bounding box of the objects to arrange the objects in order.

Step 1. I will find the coordinates of the objects in an array. If we consider the image as a 2D array, with the top left corner having coordinates (0, 0), the horizontal axis is X and the vertical axis is Y. The coordinate (X, Y) is considered as the center point

of the object. To be able to determine the coordinates (X, Y) of an object on the image, we can find it based on the position of the edges of the bounding box.

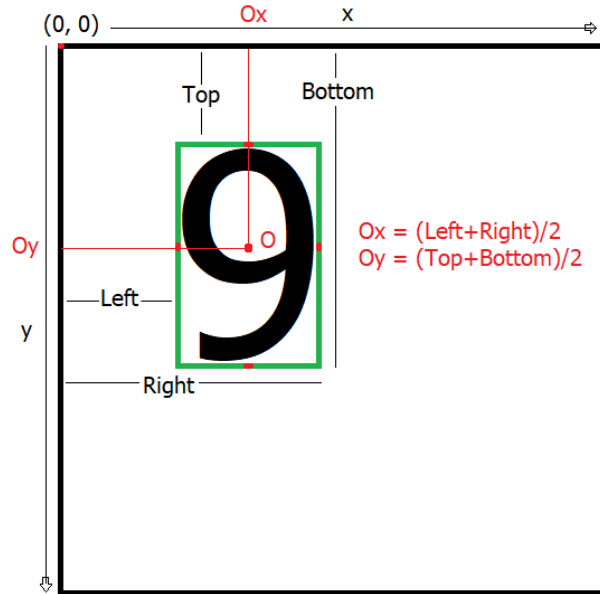


Figure 13. Determine coordinates of object

The X coordinate can be specified by $(Left + Right) / 2$, and the Y coordinate is calculated by $(Top + Bottom) / 2$. After finding 2 points X and Y, the coordinates of those 2 points are the center of the object, calculated from the top left origin (0, 0). The point O(O_x , O_y) in **Figure 13** is the coordinates of the center of the object, being calculated from Location (left, top, right, bottom) of the bounding box.

Step 2. After finding the X coordinates and Y coordinates of each detected object, I will store each object with the properties as: class name, location (left, top, right, bottom), recognition ratio, along with the X coordinate and Y just found into a 1-dimensional array. Objects may appear to be overlapping, so between two objects with the center of coordinates close to each other, the object with a lower ratio will be removed from the array.

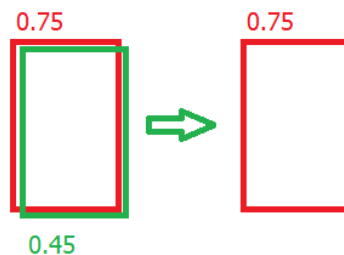


Figure 14. Remove overlapped object

To determine if any objects are overlapping, I will rely on the X and Y coordinates of each object to determine. If there is an object whose Y-axis coordinates are between top and bottom edge, and the X-axis coordinates are between left and right edge of another object, the recognition rate will be checked. The recognition rate of object is higher then will keep that object and delete the remaining object. The result of this step is the array with objects is not overlapping.

Step 3. I will find the midpoint of the object with the smallest Y axis and the object with the largest Y axis. Iterate over each element in the array to find Ymin and Ymax, then find the Yaverage by $(Ymin + Ymax)/2$.

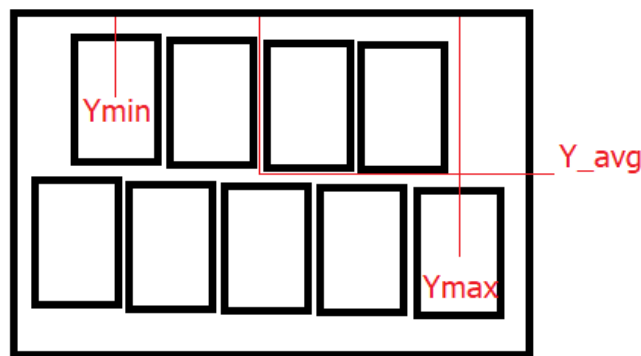


Figure 15. Find the Y-average of Ymin and Ymax

This Yaverage point is considered as the dividing line between two lines.

Step 4. The license plate in Vietnam will usually be in the form of 1 or 2 rows and the input number plate is not always neat and straight, to be able to determine the number of rows on the input number plate, I will go through the array to see if there are any objects whose Y-axis coordinates are greater than the bottom edge of the object whose coordinates are Ymin. If there is an object with Y-axis $>$ the bottom edge of the object has Ymin, the number plate will be 2 rows.

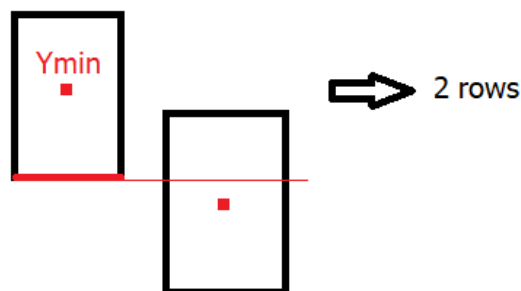


Figure 16. Determine the plate has 1 or 2 rows

Step 5. After **step 4**, I determined that the input is 1 or 2 lines. In this step, I will according this number of lines to split the array we had in **step 2** into 1 or 2 arrays.

- If we have 2 lines, we will have 2 arrays. According the Yaverage in **step 3**, if the objects with Y-axis is smaller than Yaverage, these objects will be included in array 1. If not, these objects will be included in array 2 because this Y-axis is greater than Yaverage.
- If we have 1 lines, we just have an array with all objects similar in **step 2**.

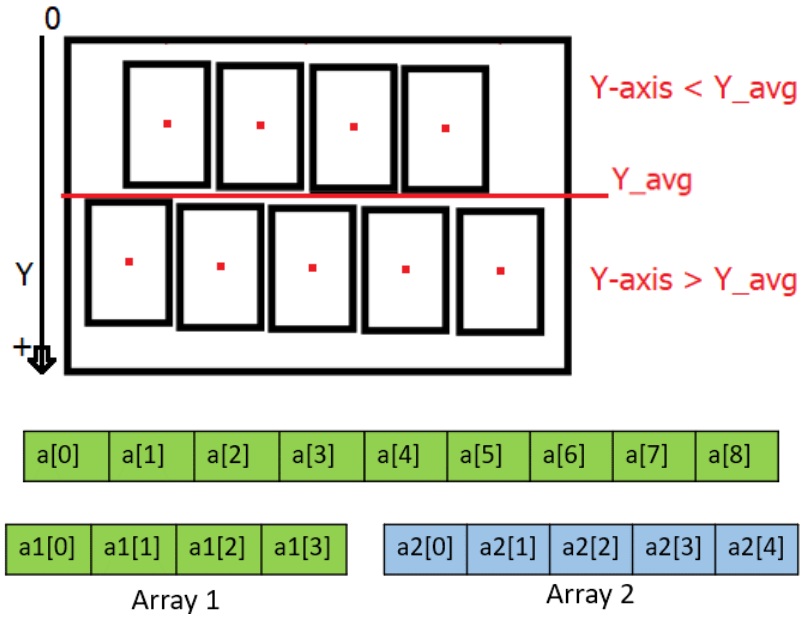


Figure 17. Split origin array into 2 arrays

Step 6. For each array in **step 5**, I will proceed to sort the objects in the array based on the X-axis of each object. The new order will be sorted by ascending order of X-axis of each object, from small to large, representing the order from left to right. The result of this step will be the array(s) with the order of the objects from left to right as same as in the license plate.

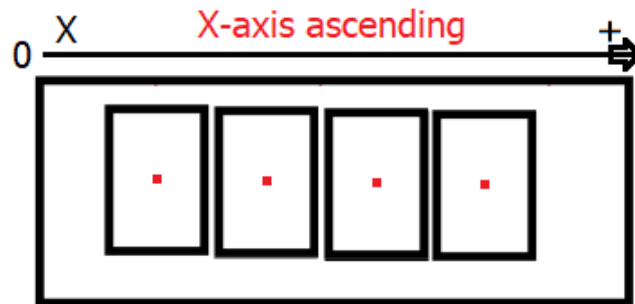


Figure 18. Sort array by ascending order of X-axis

Step 7. After we have 2 arrays representing 2 rows, and the order of elements in the arrays represents the order of objects from left to right of each array. In this step I will proceed to merge 2 arrays to create a complete sequence of characters. the elements of array 1 will be at the beginning, then the elements of array 2. The result of this step is the completed array with order of objects will be stored from left to right, from top to bottom as same as in the license plate.

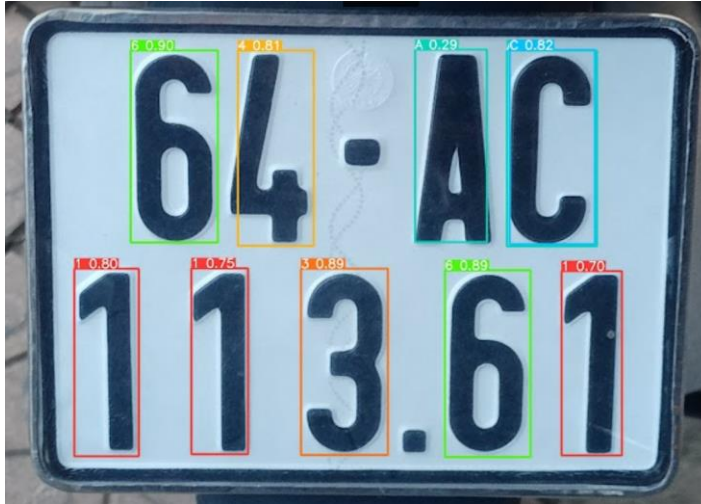


Figure 19. Merge 2 array after being sorted

CHAPTER 4. TESTING AND EVALUATION

4.1. Testing

The testing results of trained model:



6 – 0.90
4 – 0.81
A – 0.29
C – 0.82
1 – 0.80
1 – 0.75
3 – 0.89
6 – 0.89
1 – 0.70

Figure 20. Testing result 1



1 – 0.83
B – 0.80 (wrong)
F – 0.42 (wrong)
1 – 0.68
0
0 – 0.89
0 – 0.88
0 – 0.88
0 – 0.84

Figure 21. Testing result 2



Figure 22. Testing result 3

The results of testing application with algorithm to arrange recognized objects in the correct order:

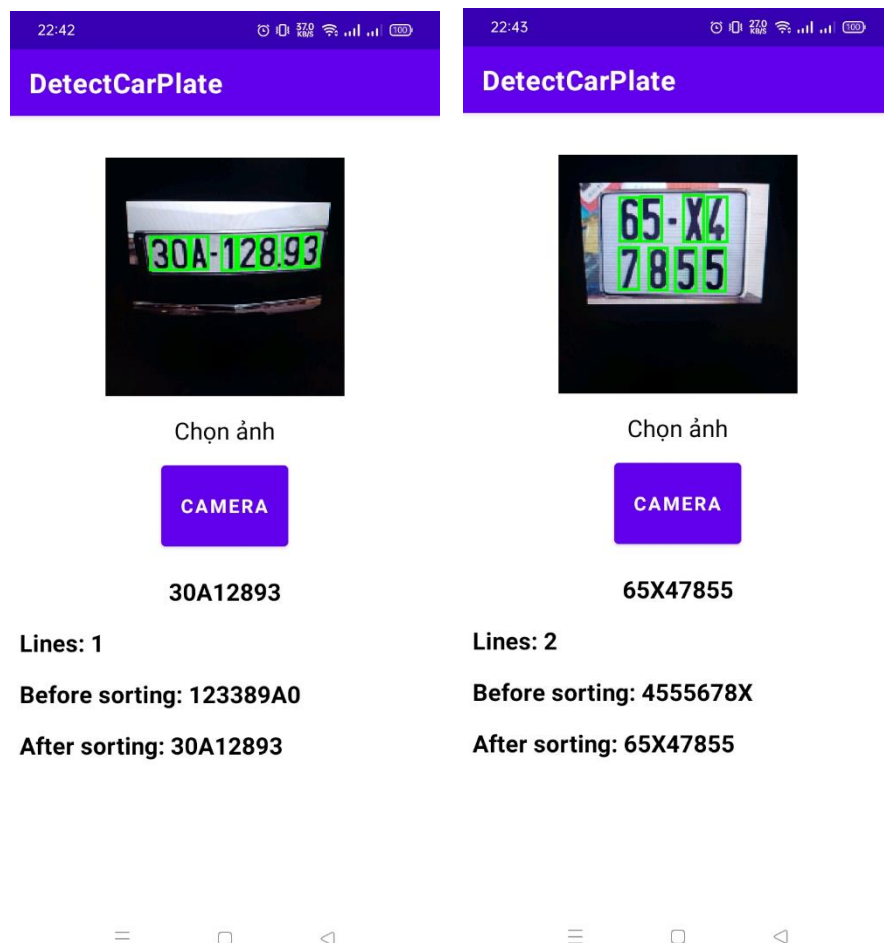




Figure 23. Testing result 4 with algorithm

4.2. Evaluation

After several tests, the model still receives errors in some classes when the sign image is not clear with the characters. The model is still confused between a number of classes with similar characteristics (number 8 and letter B; letter G, letter C and number 6; number 4 with letter A; letter T and number 7 ...).

The algorithm to arrange recognized objects in the correct order works well in most cases, but if the input license plate image is too skewed, it will lead to incorrect identification of the number plate row and lead to incorrect results.

CONCLUSION

Result and Discussion

Through this topic, I can learn more about deep learning depending on the process to study about YOLOv5 model.

This topic also help me to improve my own algorithmic thinking in order to solve the problem I meet when studying in this topic.

Future Works

The recognition rate of the model in some classes with features similarly is still have confusion (as number 8 with letter B; letter G, letter C with number 6; number 4 with letter A; letter T with number 7...) . The model was trained with black characters on a white background on most license plates in Vietnam, but with state license plates with blue or red plates or white characters remained untrained. In the future, it is possible to collect more images as well as find ways to improve the model to give more accurate identification results.

The model is tested on the test application based on the mobile-android platform. In the future, it can be improved so that the model can be applied to the computer system, applied to real-time identification as well as the application of the recognition model into parking systems through camera, contributing to the management becomes easier and more secure.

REFERENCES

- [1] Nguyen Thai-Nghe and Nguyen Chi-Ngon. 2014. An Approach for Building an Intelligent Parking Support System. In Proceedings of the Fifth Symposium on Information and Communication Technology (SoICT '14). ACM, New York, NY, USA. pp. 192-201. ISBN: 978-1-4503-2930-9. [Accessed Oct 24 2022].
- [2] Tìm hiểu về YOLO trong bài toán real-time object detection [Online]. Available: <https://viblo.asia/p/tim-hieu-ve-yolo-trong-bai-toan-real-time-object-detection-yMnKMdvr57P> [Accessed Oct 24 2022].
- [3] YOLO You Only Look Once [Online]. Available: <https://phamdinhhkhanh.github.io/2020/03/09/DarknetAlgorithm.html> [Accessed Oct 24 2022].
- [4] The network architecture of YOLOv5. [Online]. Available: https://www.researchgate.net/figure/The-network-architecture-of-YOLOv5-1-Backbone-CSPDarknet-for-feature-extraction-2_fig1_358553872 [Accessed Oct 24 2022].
- [5] Series YOLO: #5 YOLOv5: Nhanh, gọn, nhẹ và dễ sử dụng. – DevAI [Online]. Available: <https://devai.info/2021/09/01/series-yolo-5-yolov5-nhanh-gon-nhe-va-de-su-dung/> [Accessed Oct 24 2022].
- [6] Global Wheat Detection [Online]. Available: <https://www.kaggle.com/competitions/global-wheat-detection/overview/evaluation> [Accessed Oct 24 2022].
- [7] <https://www.kaggle.com/datasets> [Accessed Oct 24 2022].
- [8] <https://github.com/heartexlabs/labelImg> [Accessed Oct 24 2022].

Topic: Building model to recognize Vietnamese license plates based on YOLOv5 model

APPENDICES

Google Drive <https://drive.google.com/drive/folders/1Hnd-fulSQQlps1Fu4UMSs-MyekOTdSec>

Source Code <https://github.com/hxpdong/DetectCarPlate>