# Money detection with YOLOv5

Vu Tien Danh - 15873
15873@student.vgu.edu.vn

Vo Nhu Duc Nghia - 16299
16299@student.vgu.edu.vn

Le Duc Minh - 16669
16669@student.vgu.edun.vn

# Chapter 1

# Introduction

## 1.1 Problem statement

Deep learning has developed and become closer to modern technology. Indeed, technology companies such as Google, Baidu or Facebook have been applying deep learning techniques to their products, which shows the usefulness and necessity of applying deep learning techniques into everyday life. Technologies that have integrated with deep learning include speech recognition, virtual assistants (Apple's Siri, Cortana's Microsoft, Amazon's Alexa or Baidu's DeepVoice), automatic translation (Google Translate[1]), facial recognition (Facebook's Deepface or lipsync[2] techonologies that learn the movement of the human lips while speaking). Clearly, the application of deep learning technology in life increases the variety and efficiency of products as well as proves the usefulness and necessity of deep learning technology.

Object detection is more focused and developed recently. Its successor is facial detection, people could witness its development via the rise of application of photo editing software, photo filters on social networks (Facebook, Instagram, Snapchat). The integration of deep learning technologies makes the recognition more powerful and superior. Policies to choose artificial intelligence development as the spearhead for science and technology development and long-term orientation to maintain stability and continuous development for those policies have been proposed[3]. However, the study of computer vision has not been widely developed in Vietnam, our research aims to delve into the phenomenon and explore its potential in the application of our daily life.

Object recognition is a broad problem with many approaches and solutions for the same problem. We found cash identification to be an interesting and potential problem. Indeed, when making a sale, it is inevitable to neglect or forget existing thoughts to make transactions, which can lead to loss in the cash exchange process, with support of cash recognition technology, cash recognition will assist in the calculation, recording or recognize the monetary value of money.

[1]https://en.wikipedia.org/wiki/Google$_{N}eural_{M}achine_{T}ranslation$

[2]https://www.lipsync.ai/

[3]https://sti.vista.gov.vn/tw/Lists/TaiLieuKHCN/Attachments/300474/47989-469-151744-1-10-20200513.pdf

## 1.2 Application

In everyday society, object recognition offers several benefits such as verifying whether an object is fake or real, or as entertainment applications. Our research is to determine the value of Vietnamese banknotes, through which, we aim for practical applications such as identifying counterfeit money, converting money to bills from Vietnam to other countries, furthermore, it could be proposed to be integrated into payment machines in retail stores.

## 1.3 Objective

Research artificial intelligence techniques, deep learning technology and its application in real life situation. In details, this research uses YOLOv5 for detecting money with many denominations.

Researching and describing the money recognition system. At the same time, evaluate the performance of the method by experiments on real data sets. Furthermore, applying deep learning techniques to money recognition for integration into automated trading systems.

# Chapter 2

# Methodology

## 2.1 Data Collection and Processing

Data collection and processing is an important part of model training. In order to achieve the expected results, the volume of data needs to be maximized. We use Vietnamese banknotes of all denominations as the dataset.

Our dataset is collected from various sources. The data source is collected from many different sources. Most of the data sources are taken from taking pictures with a phone. At the same time, we also use other programs to assist with image capture such as cutting frames from the videos we filmed the banknotes. Finally, a small portion of it is taken from online sources.

## 2.2 System architecture

### 2.2.1 You Only Look Once (YOLO)

You Only Look Once (YOLO) is renowned for its ability to identify objects. Combining with deep learning algorithms, YOLO becomes a popular algorithm that has gone viral. In detail, YOLO uses Convolution Neural Network as one of its main components for detecting objects.

**Object Detection** According to whether they are one- or two-stage, whether anchor frames are used, and the labeling techniques used, object detection algorithms may generally be divided into two main classes[1].

**Two-stage approach** In a two-stage object detection process, the first step creates region or object proposals, which are then categorized and detected in the second stage utilizing bounding boxes. Prominent algorithms in this category are R-CNN, Fast R-CNN, and Faster R-CNN.

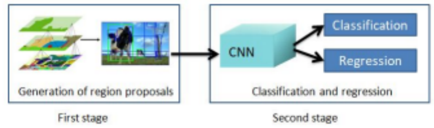



Figure 2.1: Architecture of two-stages approach[1]

**One-stage approach** One feed-forward fully convolutional network, seen in one-stage detectors, is responsible for both the object classification and the bounding boxes. In other words, the one-stage approach uses regular and dense sampling with consideration for locations, scales, and aspect ratio to perform classification and regression in a single step. Single Shot Muiltibox Detector (SSMD) and YOLO are well-known algorithms for this class.

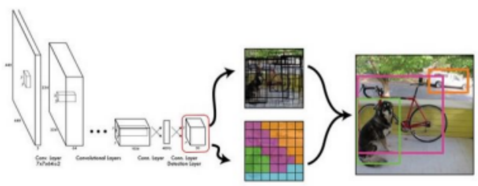


Figure 2.2: Architecture of one-stage approach[2]

**YOLOv5 Algorithm** Since the initial YOLOv1 was released in 2015, it has become overly well-liked in the computer vision world. Multiple YOLOv2, YOLOv3, YOLOv4, and YOLOv5 versions have since been released, but by various authors. With each update, YOLO improves and delivers tangible results, not to mention enormous generation-by-generation growth in batch normalization, high-resolution grading, and optimization. computation in classification and recognition during training. YOLOv5 has gained many good features such as flexible control of model size, application of Hardswish activation function and data enhancement[3].

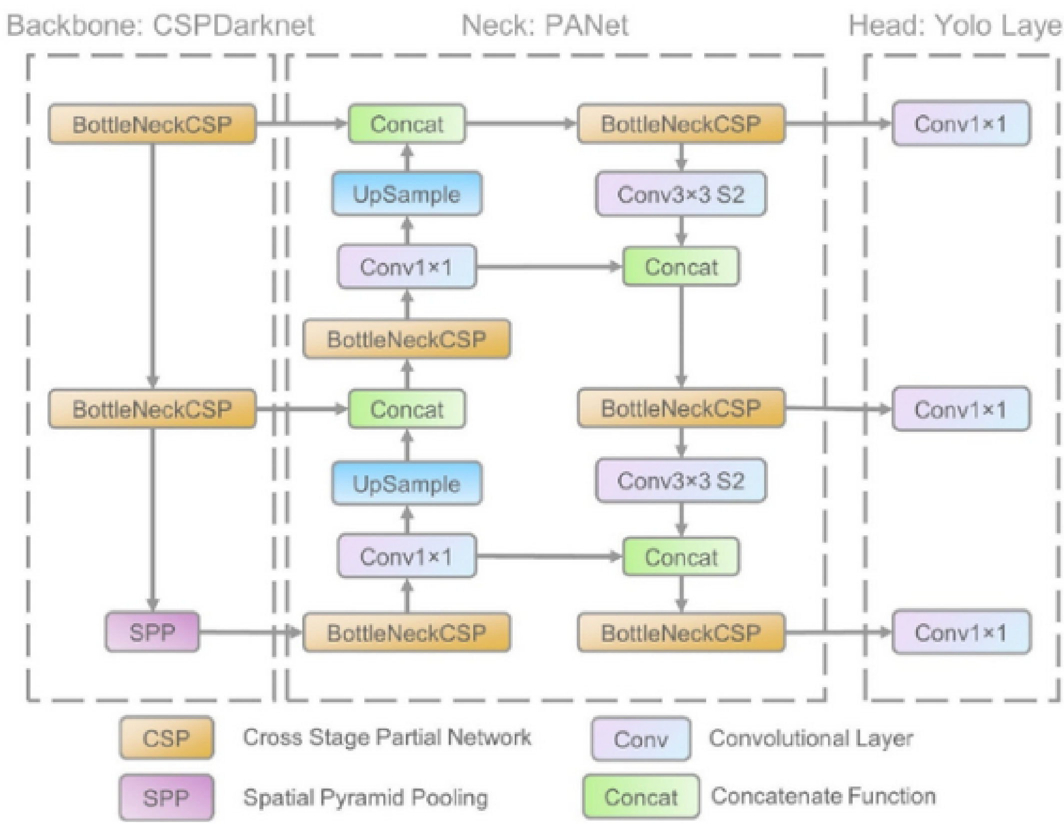



Figure 2.3: The architecture of YOLOv5[1]

Overall, YOLOv5 has three main part:

- Backbone.
- Neck.
- Head.

The basic purpose of Backbone is to extract significant features from an input image. To extract valuable, important features from an input image in YOLOv5, the CSP—Cross Stage Partial Networks are employed as the backbone. PANet is utilized in YOLOv5 as a neck to obtain feature pyramids. Pyramids of features are primarily created using the neck. Pyramids of features enable models to scale objects successfully in general. The ability to recognize the same thing in various sizes and scales is helpful. Primarily, the final detection step is carried out using the model Head. The final output vectors are generated with class probabilities, objectness scores, and bounding boxes after anchor boxes are applied to the features.

[1] https://iq.opengenus.org/yolov5/

### 2.2.2 Implementation of Deep Learning Framework

A GPU computer is typically required for deep learning to perform high-efficiency computation. In this study, an open-source Google Colab is used to create the environment for building and testing learning models. A dataset must be well-prepared and uploaded into the newly established YOLO folder in order to create a Google Colab project. Before installing YOLO algorithms, the user must verify all of the GPU settings. Finally, the model training and testing of the chosen YOLO algorithm can be carried out sequentially if the pre-defined weights are available. Following are steps for installing YOLOv5 and Google Colab, as well as for training and testing models.

Creating Google Colab project:

1. Log into Google.
2. Log into Google Drive.
3. Create Google Colab.
4. Upload dataset in .zip form.

Installing YOLOv5:

1. Open Google Colab project.
2. Change the runtime to GPU.
3. Connect with Google Drive.
4. Clone YOLOv5[1].
5. Install requirements.
6. Get test pre-train models.
7. Install and test YOLOv5.

Training and testing model:

1. Get train pre-train models.
2. Create .yaml file to define elements within the dataset.
3. Train model.
4. Train and test the result.

[1] https://github.com/ultralytics/yolov5

## 2.3 Challenges

Object recognition requires on both sides that in order to achieve minimum error in object recognition, the model must be very robust and that the objects, when trained, must highlight their own characteristics to avoid fields. The combination of environments makes the objects look almost the same, which makes it difficult for the algorithm to determine the true value of the object. YOLOv5 is a modern and very powerful technology in object recognition. However, when performing tests on the notes, an error occurred. Among them are the details of the money that do not seem to make its difference except the price and the notes on it, for example, the 20,000 and 500,000 bills look completely different, but when they are placed in the same place that are not so dark but able to make the letters and features on the surface of money are not prominent, the two bills appear to be detected to be the same.

# Chapter 3

# Results of experiments

## 3.1 Evaluation Metric

**IoU** Intersection over Union is an evaluation metric used to measure the accuracy of an object detector on a particular dataset.

**mAP@0.5 and mAP@0.5:0.95** means that it is the mAP calculated at IOU threshold 0.5 and threshold (0.5, 0.55, 0.6, ..., 0.95). The general definition for the Average Precision(AP) is finding the area under the precision-recall curve. The process of plotting the model's precision and recall as a function of the model's confidence threshold is the precision recall curve. there is not any best metrix but mAP@0.5:0.95 is the most widely recognized object detection metric. So we use mAP@0.5:0.95 to judge the model result.

Model summary: 157 layers, 7034398 parameters, 0 gradients, 15.8 GFLOPs

| Class | Images | Instances | P | R | mAP50 | mAP50-95: 100% 2/2 [00:01<00:00, 1.02it/s] |
|---|---|---|---|---|---|---|
| all | 138 | 158 | 0.943 | 0.959 | 0.975 | 0.871 |
| 1000VND | 138 | 10 | 0.942 | 1 | 0.995 | 0.87 |
| 2000VND | 138 | 11 | 0.807 | 0.909 | 0.908 | 0.847 |
| 5000VND | 138 | 9 | 0.958 | 1 | 0.995 | 0.914 |
| 10000VND | 138 | 17 | 0.94 | 0.824 | 0.941 | 0.81 |
| 20000VND | 138 | 30 | 1 | 0.897 | 0.988 | 0.889 |
| 50000VND | 138 | 21 | 0.98 | 1 | 0.995 | 0.825 |
| 100000VND | 138 | 7 | 0.961 | 1 | 0.995 | 0.917 |
| 200000VND | 138 | 26 | 0.963 | 0.999 | 0.976 | 0.858 |
| 500000VND | 138 | 27 | 0.936 | 1 | 0.985 | 0.909 |

## 3.2 Benchmark data

Before getting to the benchmark data, we would like to make a small explanation about the terminology.

**Noted** All samples are collected manually by our members, no image had been taken from the internet.

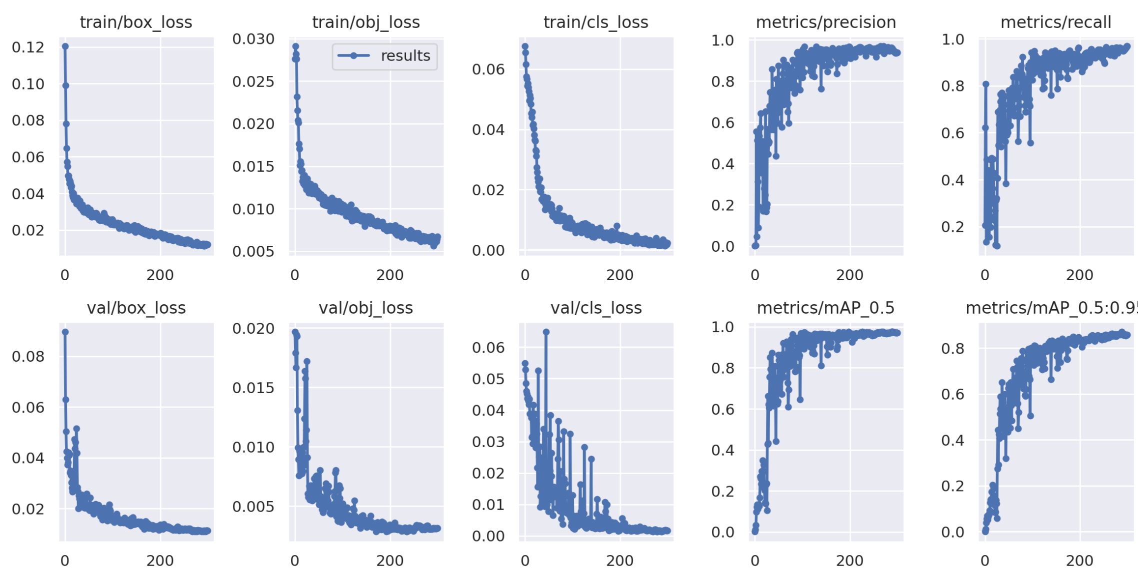
validation loss is a metric used to assess the performance of a deep learning model on the validation set. The validation set is a portion of the dataset set aside to validate the performance of the model. The validation loss is similar to the training loss and is calculated from a sum of the errors for each example in the validation set.

Additionally, the validation loss is measured after each epoch. This informs us as to whether the model needs further tuning or adjustments or not. To do this, we usually plot a learning curve for the validation loss.

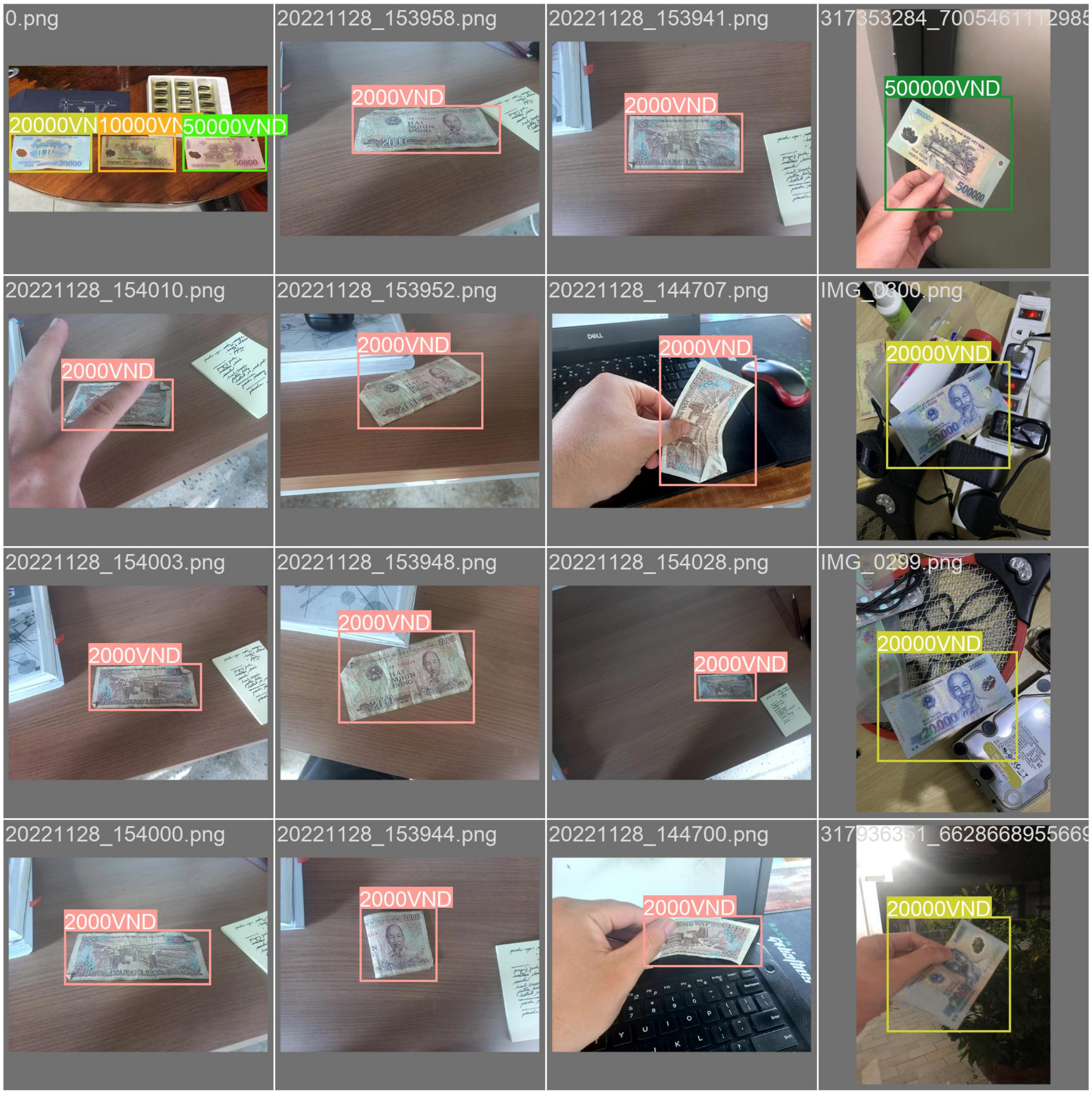
**box loss** The box loss represents how well the algorithm can locate the centre of an object and how well the predicted bounding box covers an object.

**obj loss** Objectness is essentially a measure of the probability that an object exists in a proposed region of interest. If the objectivity is high, this means that the image window is likely to contain an object.

**cls loss** Classification loss gives an idea of how well the algorithm can predict the correct class of a given object.




The model improved swiftly in terms of precision, recall and mean average precision before plateauing after about 200 epochs. The box, objectness and classification losses of the validation data also showed a rapid decline until around epoch 200. We used early stopping to select the best weights.

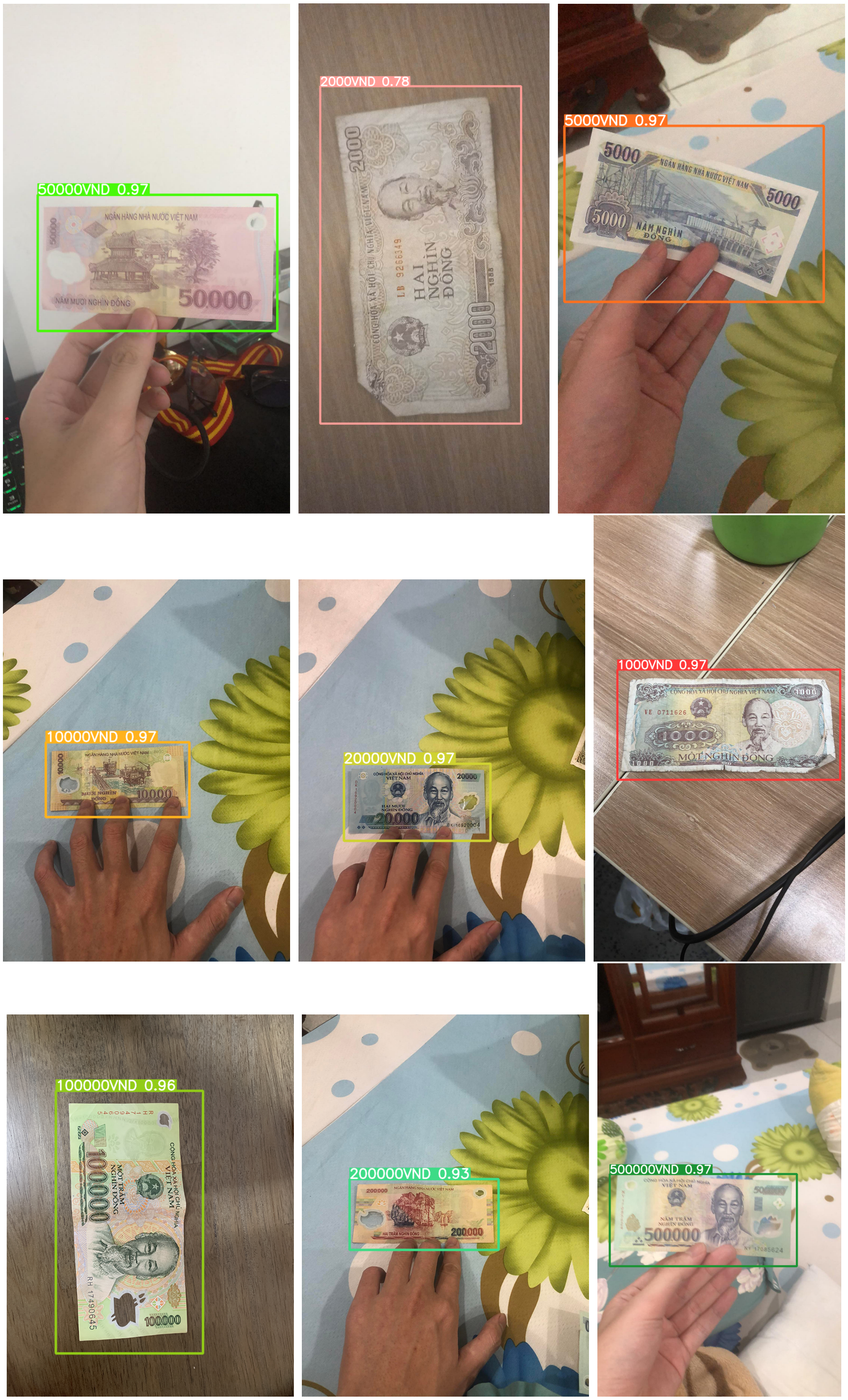

0.png
20000VN
10000VN
50000VND
20221128_153958.png
2000VND
20221128_153941.png
2000VND
500000VND
20221128_154010.png
2000VND
20221128_153952.png
2000VND
20221128_144707.png
2000VND
20000VND
20221128_154003.png
2000VND
20221128_153948.png
2000VND
20221128_154028.png
2000VND
IMG_0299.png
20000VND
20221128_154000.png
2000VND
20221128_153944.png
2000VND
20221128_144700.png
2000VND
20000VND

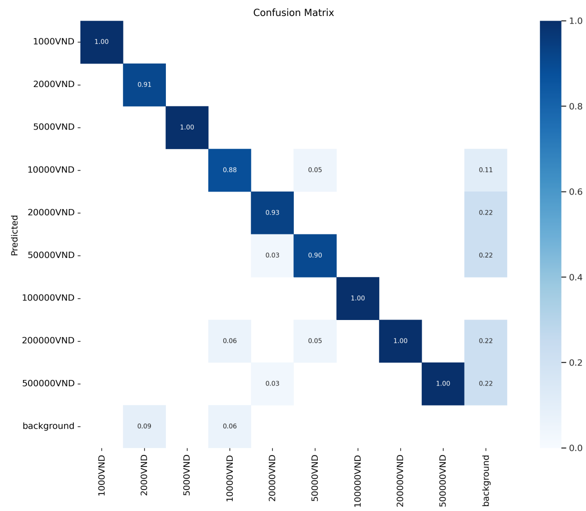

0.png
20000VND 0.9
50000VND 0.9
20221128_153958.png
2000VND 1.0
20221128_153941.png
2000VND 1.0
500000VND 1.0
20221128_154010.png
2000VND 1.0
20221128_153952.png
2000VND 1.0
20221128_144707.png
2000VND 1.0
20000VND 0.9
20221128_154003.png
2000VND 1.0
20221128_153948.png
2000VND 1.0
20221128_154028.png
2000VND 1.0
IMG_0299.png
20000VND 0.7
20221128_154000.png
2000VND 1.0
20221128_153944.png
2000VND 1.0
20221128_144700.png
2000VND 1.0
20000VND 1.0

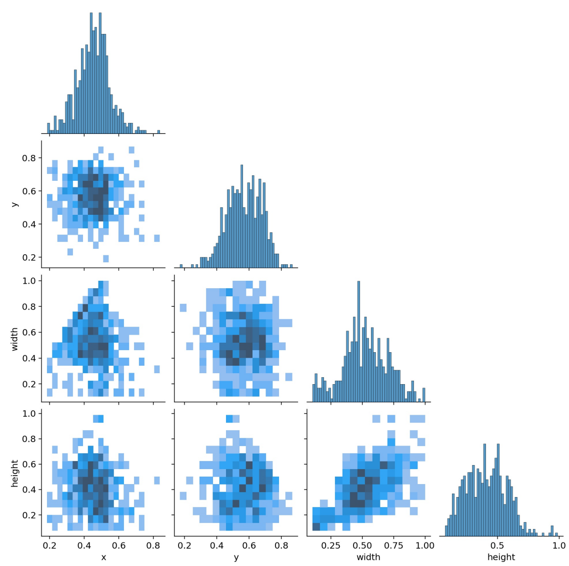

5000VND
500000VND
50000VND
500000VND
500000VND
10000VND
20000VND
50000VND
5000VND
500000VND
500000VND
500000VND

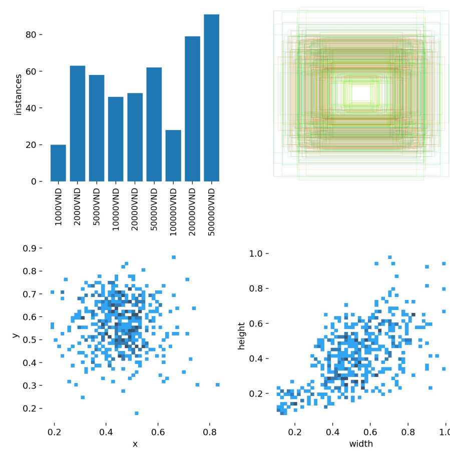

## 3.3 Results

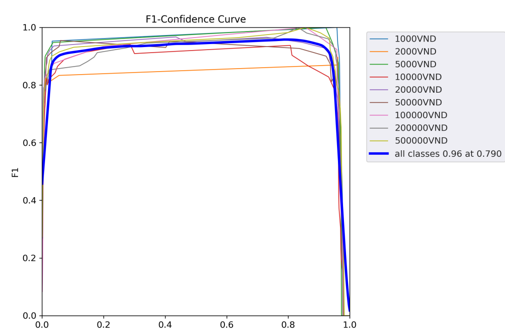
**Samples** Here are some photos have been detected by our model.


50000VND 0.97
2000VND 0.78
5000VND 0.97
10000VND 0.97
20000VND 0.97
1000VND 0.97
100000VND 0.96
200000VND 0.93
500000VND 0.97




```
Model summary: 157 layers, 7034398 parameters, 0 gradients, 15.8 GFLOPs
                 Class     Images  Instances          P          R      mAP50   mAP50-95: 100% 2/2 [00:01<00:00,  1.02it/s]
                   all        138        158      0.943      0.959      0.975      0.871
               1000VND        138         10      0.942          1      0.995       0.87
               2000VND        138         11      0.807      0.909      0.908      0.847
               5000VND        138          9      0.958          1      0.995      0.914
              10000VND        138         17       0.94      0.824      0.941       0.81
              20000VND        138         30          1      0.897      0.988      0.889
              50000VND        138         21       0.98          1      0.995      0.825
             100000VND        138          7      0.961          1      0.995      0.917
             200000VND        138         26      0.963      0.999      0.976      0.858
             500000VND        138         27      0.936          1      0.985      0.909
```

More will be sent to you in a zip file.

# Chapter 4

# Conclusion and Future Work

**To conclusion** , our training model is stable at detecting a single denomination in a picture and also many denominations in a picture with different color. For example: 10.000VND, 20.000VND, 50.000VND. It is a bit difficult for out model to detect the denominations with nearly same color like 5.000VND, 20.000VND, 500.000VND as our money do not have features so that the model can distinguish.

**Future work:** we aim for practical applications such as identifying counterfeit money, converting money to bills from Vietnam to other countries, furthermore, it could be proposed to be integrated into payment machines in retail stores.

# Bibliography

[1] F.Sultana, A.Sufian, and P.Dutta. A Review of Object Detection Models based on Convolutional Neural Network. *CoRR*, abs/1905.01614, 2016. https://arxiv.org/abs/1905.01614.

[2] J.Redmon, S.Divvala, R.Girshick, and A.Farhadi. You Only Look Once: Unified, Real-Time Object Detection. *CoRR*, abs/1506.02640, 2016. https://arxiv.org/abs/1506.02640.

[3] P.Jiang, D.Ergu, F.Liu, and B.Ma. A Review of Yolo Algorithm Developments. *Procedia Computer Science*, 199:1066–1073, 2016. https://www.sciencedirect.com/science/article/pii/S1877050922001363.