國立高雄大學資訊工程學系

碩士論文

應用 M3-YOLOv5 物件偵測演算法於晶片輪廓檢測

Applying M3-YOLOv5 Object Detection Algorithm to Chip Contour Inspection

研 究 生： 謝佳衛

指導教授： 張保榮 教授

中華民國一百一十二年一月

應用 M3-YOLOv5 物件偵測演算法於晶片輪廓檢測

指導教授：張保榮 教授
國立高雄大學資訊工程學系

研究生：謝佳衛
國立高雄大學資訊工程學系碩士班

摘要

本研究將 M3-YOLOv5 物件偵測演算法部署在嵌入式平台 Jetson Xavier NX 上，實現晶片槽內晶片輪廓的即時物件偵測和影像辨識的優異性能。只要檢測到有缺陷的晶片，它就會立即發出警告，並記錄晶片插槽的位置和發生的時間戳記。當警報響起時，操作將暫停一段時間，操作人員可以立即取出有缺陷的晶片，以避免擠入該晶片槽中的下一個晶片。本研究的目的是分析和比較 M3-YOLOv5 物件偵測演算法、GSEH-YOLOv5 物件偵測演算法和傳統 YOLOv5 物件偵測演算法在物件偵測和影像辨識方面的性能。結果，實驗顯示我們提出的使用嵌入式平台 Jetson Xavier NX 的方法的準確度為 98.9%，速度為 25 fps。換句話說，與傳統的 YOLOv5 相比，所提出的方法可以提高 0.4% 的精度和 12.6% 的速度，而與 GSEH-YOLOv5 相比速度略差，因為 GSEH-YOLOv5 旨在提高速度以犧牲更多的精度。最後，所提出的方法顯著優於其他方法。

關鍵字：Jetson Xavier NX、即時物件偵測、即時影像感知與辨識、M3-YOLOv5、注意力機制。

Applying M3-YOLOv5 Object Detection Algorithm to Chip Contour Inspection

Advisor: Dr. Bao-Rong Chang
Department of Computer Science and Information Engineering
National University of Kaohsiung

Student: Chia-Wei Hsieh
Department of Computer Science and Information Engineering
National University of Kaohsiung

ABSTRACT

In this paper, M3-YOLOv5 algorithm is deployed on the embedded platform, Jetson Xavier NX, to implement the excellent performance for real-time object detection and image recognition of the chip contour in the chip slot. As long as a defective chip is detected, it will immediately issue a warning and record the position in which chip slots and the timestamp of occurrence. When the alarm goes off, the operation will be paused for a while and the operators can immediately remove the defective chip to avoid squeezing the next chip in that chip slot. The objective of this paper is to analyze and compare the performance of object detection and image recognition among the M3-YOLOv5 algorithm, the GSEH-YOLOv5 algorithm, and the traditional YOLOv5 algorithm. As a result, the experiments show the accuracy of 98.9% and the speed of 25 fps with our proposed approach using the embedded platform Jetson Xavier NX. In other words, compared with the traditional YOLOv5, the proposed approach can increase the accuracy by 0.4% and the speed by 12.6% while compared with GSEH-YOLOv5 slightly worse in speed because GSEH-YOLOv5 aims to increase the speed to sacrifice more accuracy. Finally, the proposed approach outperforms the other methods significantly.

Keywords: Jetson Xavier NX, Real-time Object detection, Real-time Image Sensing and Recognition, M3-YOLOv5, and Attention Mechanism.

誌謝

本論文能夠順利完成，首先要感謝指導教授張保榮老師，張老師有耐心地指導研究方法以及實驗設計的部分，且對於論文的內文缺失部分也提出許多相當具有建設性的建議。

在研究所求學生涯中，在張老師的指導下，不僅學到了做研究應有的態度，也學到了做人處事的方法及許多研究外的種種事物，更讓我利用實驗室豐富的資源得到很多實作上的進步。這些研究建議，都讓困難都能迎刃而解。此外，張老師的研究課題都是現今學術與企業潮流的最新技術，可以讓我對於未來在職場上游刃有餘，這一切將歸功於老師的遠見。如今我即將畢業了，心中的感激之情難以言表。接下來要感謝的是在我研究所的學長姊們郁傑、炯霖、函霖，我們一起分享不同領域的知識、也會一起討論課業，在研究中你們一直不厭煩的給我建議，讓我的研究得以順利的進行，也感謝學弟翔宇，給予我許多研究之外的幫助，研究所期間非常感謝能遇見你們。

最後，感謝我最摯愛的家人。在求學生涯中，都給予我最大的支持與安慰，使得我能夠無後顧之憂而認真鑽研於研究之中，謝謝您們一直以來栽培，希望我能不辜負您們對我的期待。

Directory

List of Tables

List of Figures

Chapter 1. Introduction

With the promotion of Industry 4.0, innovative technology has carried out significant reforms in factories, introducing big data analytics, cloud computing, and autonomous or automatic sytem into the factories and significantly improving the production capacity of the factories. When the production line is automatically transporting chips, if the production machine accidentally crushes and leave the current chip sticking in the slot, the next round of placing a chip onto the same specitial location will cause the damage for both of chips, that is, making a production loss. Therefore, how to minimize the loss is the primary goal of this study. This paper intoduces innovative deep learning model to implement the real-time image detection for solving the problem. Generally speaking, the technique of object detection have two modes – single-stage and two-stage. Single-stage combines the localization and the classification. Two-stage separates the localization and the classification. According to the past object detection algorithms, the accuracy of two-stage object detection is higher than that of single-stage object detection. Typical two-stage object detection models are Fast R-CNN [1], Faster R-CNN [2], and Masked R-CNN [3]. However, the most significant disadvantage of two-stage object detection is the enormous amount of computation. As long as there are a large number of framed targets in the picture, the next step of object classification will require a large number of crafted targets to be classified, which is really time-consuming. Many applications in daily life require real-time object detection. Specific applications include vehicle tracking, street view analysis, mask-wearing detection, operator clothing detection, and product detection in factory production lines. Nevertheless, through continuous technical improvement of single-stage object detection, it can implement at a fast speed and maintain a high accuracy ratio. This study intends to use single-stage object detection for the application of image recognition, and chooses YOLOv5 model recently published in 2020 [4] [5] [6] [7] [8] [9] [10] [11] due to the high speed of object detection and the high accuracy of image recognition.

Chapter 2. Related Work

2.1 Literature review

The core concept of YOLOv1 model proposed by Redmon and Farhadi et al. is to treat the input image as a grid with the same width and height, and predict the objects in each grid. In the beginning, YOLOv1 uses two bounding boxes to predict objects in each grid, and only one object with the highest category probability could be expected in the same grid [7]. Redmon et al. proposed an improved method for YOLOv1, called YOLOv2. The primary purpose is to improve the speed and accuracy of the model, mainly adding batch normalization and anchor box mechanism [8].

Redmon et al. added a residual network [9] to YOLOv2 to improve the accuracy of YOLOv2 later. After deepening the network structure, people referred it to as YOLOv3. YOLOv3 dramatically improves its accuracy while maintaining the same speed as YOLOv2 [10]. Bochkovskiy and the others proposed the YOLOv4 architecture, with the primary goal of designing a fast operating system with new functions. The object detection applied in the mass production system can optimize parallel computation to improve the operational speed dramatically, not just reducing the theoretical computation indicator, billion float operations per second (BFLOPS). After that, Bochkovskiy and the others have further integrated the new methods to YOLOv4 to achieve the best performance of the object detection. This is called CSPNet that can significantly improves both the speed and the accuracy [11] concurrently.

Marco et al. found that adopting the model compression algorithms to shorten the model are most likely to reduce the inference time; however, it would pay the cost of lower accuracy. Alternatively, they gave another method, running the program in the embedded platform, which can dynamically determine which DNN model used to obtain the required the accuracy and inference time according to the input data [12]. Sun et al. proposed an object detection network built in an embedded platform. Based on that the Mobile-YOLO (M-YOLO) model [13] combines the residual module and the depthwise separable convolution [14] into the feature selection layer to reduce the computational complexity of the network.

Howard et al. intoduced MobileNet [15] that mainly uses depthwise separable convolution to build a lightweight [16] deep neural network. Depthwise separable convolution can obtain a feature map similar to the traditional convolution with less computation for object detection. This is because it implements the depthwise separable convolution operation in the feature selection layer.

MobileNetV2 [17] is based on the inverted residual structure in which the shortcut connections are between the thin bottleneck layers. The intermediate expansion layer uses lightweight depthwise convolution to filter the features that are sources of non-linearity. In terms of the mobile phone CPU, MobileNetV3 [18] adjusts to it through a novel architecture of using hardware-aware network architecture search (NAS) and NetAdapt algorithm to improves its advancements. This study explores how automatic search algorithms and network design work together with complementary methods to come up with the contemporary technique level.

Vaswani et al. deviced the superior sequence conversion model based on a complex recursive or convolutional neural network, including an encoder and a decoder. The best-performing model connects the encoder and decoder through the attention mechanism [19]. The innovation of the SENet network proposed by Hu et al. is to pay attention to the relationship between feature vectors so that the model can actively understand the important features between different feature vectors [20].

2.2 Traditional YOLOv5 model

YOLOv5 uses some special techniques, such as mosaic data enhancement, adaptive image scaling, and the addition of a unique focus structure. Using these techniques can get an outstanding improvement in the field of object detection. Mosaic can significantly improve the detection ability of the YOLO series in small objects through the method of splicing and randomly zooming the image ratio. Object detection mainly used Image scaling to scale the length and width of the input image during detection, which can effectively reduce the short side length, reduce the amount of computation, and increase the speed of inference. The focus structure particularly increases the computation to splice and combine so that the feature map does not cause the loss of feature information due to compression. The architecture of the traditional YOLOv5 model is shown in Figure 1 to Figure 4.

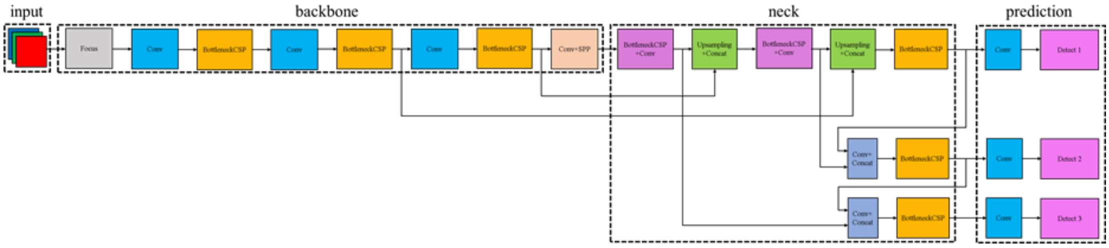


Figure 1. The architecture of the traditional YOLOv5 model

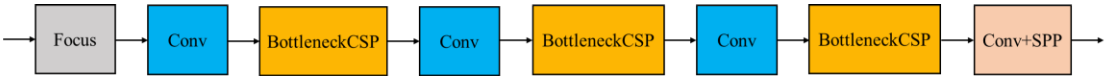


Figure 2. The backbone structure of the traditional YOLOv5 model

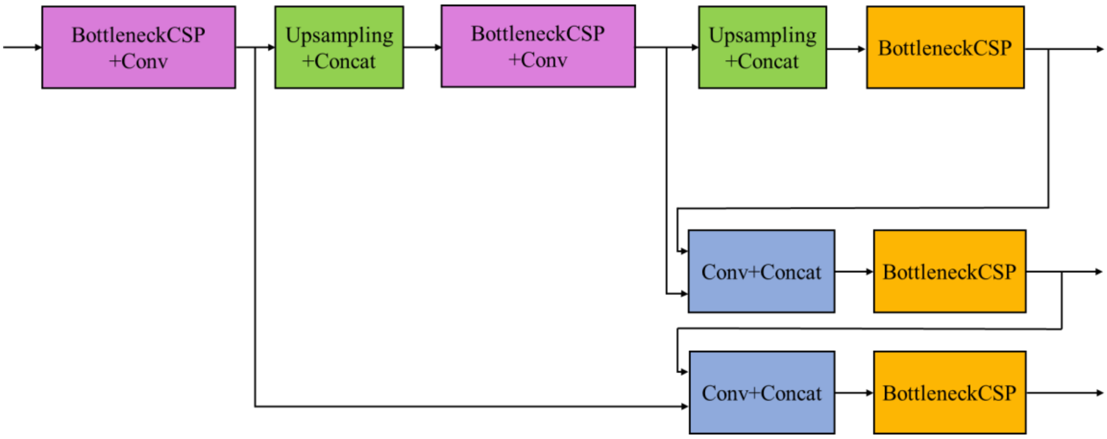


Figure 3. The neck structure of the traditional YOLOv5 model

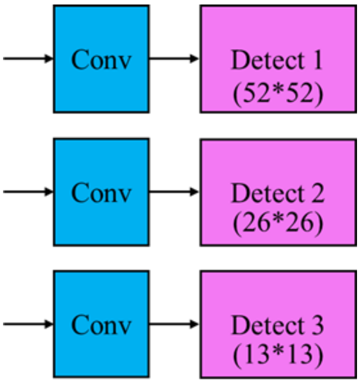


Figure 4. The prediction structure of the traditional YOLOv5 model

2.3 Alternative model GSEH-YOLOv5

Here introduced an alternative model, GSEH-YOLOv5, and its architecture is shown in Figure 5 to Figure 8. The difference between GSEH-YOLOv5 [21] and traditional YOLOv5 is that the GhostNet module with the attention mechanism replaces the CSP module. Moreover, the attention module replaces the remaining CSP module. The purpose of using the GhostNet module is to reduce unnecessary convolutional computations. The introduction of GhostNet allows the traditional YOLOv5 model to effectively reduce the number of computations and parameters during the inference. Since the method proposed in this study requires real-time inference in an embedded platform, its primary purpose is to maximize the speed of performance while maintaining the accuracy of the result as much as possible.

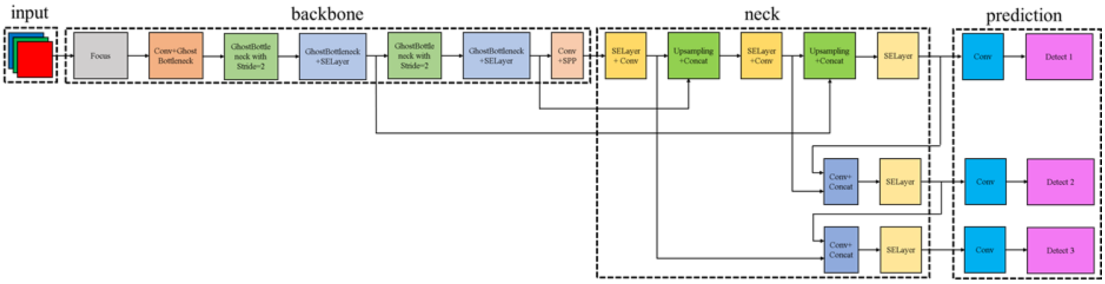


Figure 5. The architecture of the GSEH-YOLOv5 model

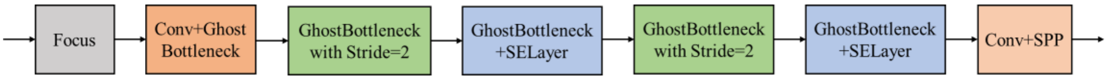


Figure 6. The backbone structure of the GSEH-YOLOv5 model

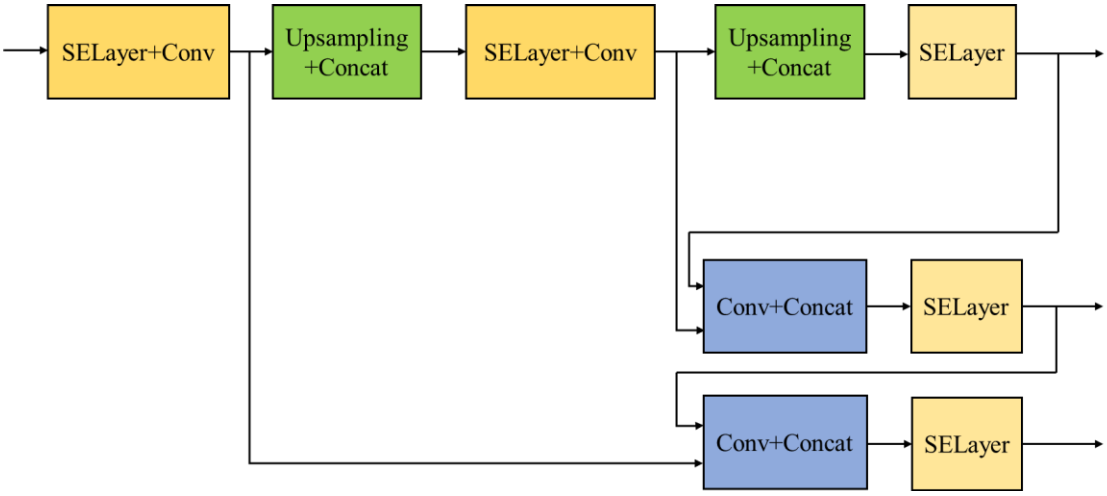


Figure 7. The neck structure of the GSEH-YOLOv5 model

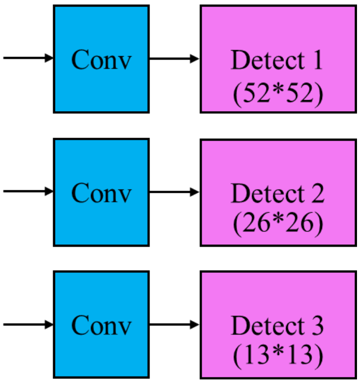


Figure 8. The prediction structure of the GSEH-YOLOv5 model

Chapter 3. Research Method

3.1 Data preparation and labeling

It is beginning to do manually labeling with data set provided by a semiconductor company in sourthen Taiwan. The Image labeling with LabelImg goes through a total of 553 training data and 89 validation data [22] where a single datum represents a frame. Next, we manually label the position with the target in each frame. After position labeling, there are three conditions — empty slot, slot occupied with an intact chip or a defective chip, as shown in Figure 9. In other words, we refer these to as empty condition, occupied condition, or defective condition in the following sections.

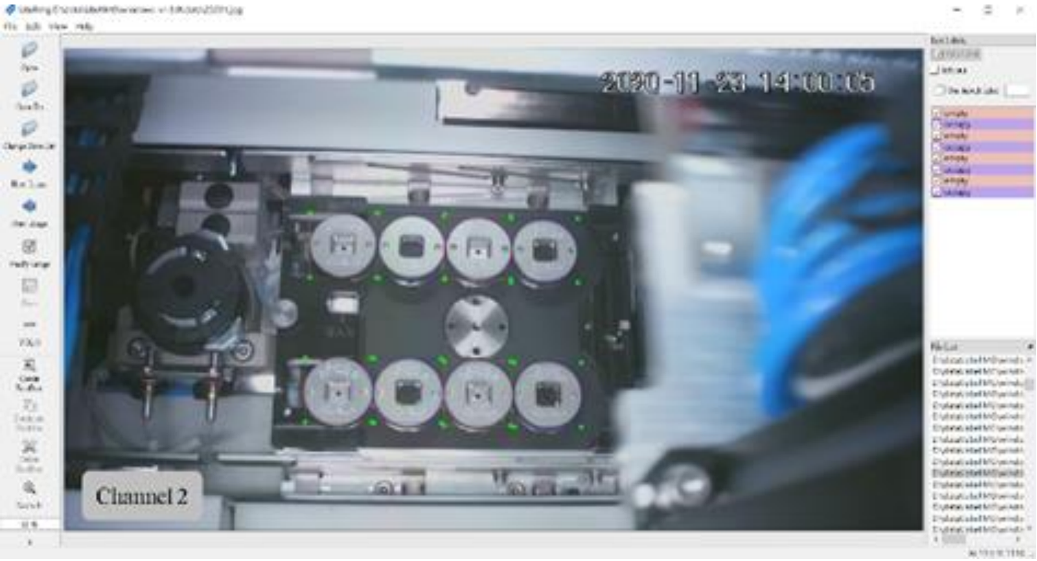
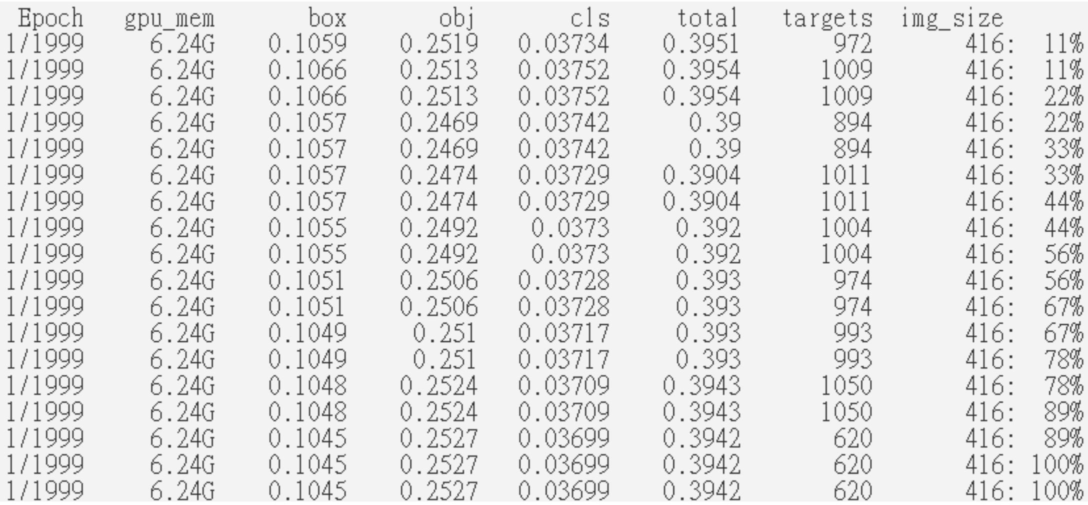


Figure 9. Image labeling with LabelImg

3.2 Model training and performance evaluation

This study uses PyTorch [23] as the training framework and uses the collected and labeled training dataset to train a model suitable for the situation. A screenshot of the actual training process is shown in Figure 10. If the accuracy rate does not reach the expected level, we need to adjust the parameters and check whether the mark quality of the dataset is good. The mAP uses to evaluate the quality of the model. The closer the value of mAP is to one, the better the model's performance, as shown in Figure 11.



```
 Epoch    gpu_mem       box       obj       cls     total   targets  img_size
1/1999      6.24G    0.1059    0.2519   0.03734    0.3951       972       416:  11%
1/1999      6.24G    0.1066    0.2513   0.03752    0.3954      1009       416:  11%
1/1999      6.24G    0.1066    0.2513   0.03752    0.3954      1009       416:  22%
1/1999      6.24G    0.1057    0.2469   0.03742      0.39       894       416:  22%
1/1999      6.24G    0.1057    0.2469   0.03742      0.39       894       416:  33%
1/1999      6.24G    0.1057    0.2474   0.03729    0.3904      1011       416:  33%
1/1999      6.24G    0.1057    0.2474   0.03729    0.3904      1011       416:  44%
1/1999      6.24G    0.1055    0.2492    0.0373     0.392      1004       416:  44%
1/1999      6.24G    0.1055    0.2492    0.0373     0.392      1004       416:  56%
1/1999      6.24G    0.1051    0.2506   0.03728     0.393       974       416:  56%
1/1999      6.24G    0.1051    0.2506   0.03728     0.393       974       416:  67%
1/1999      6.24G    0.1049     0.251   0.03717     0.393       993       416:  67%
1/1999      6.24G    0.1049     0.251   0.03717     0.393       993       416:  78%
1/1999      6.24G    0.1048    0.2524   0.03709    0.3943      1050       416:  78%
1/1999      6.24G    0.1048    0.2524   0.03709    0.3943      1050       416:  89%
1/1999      6.24G    0.1045    0.2527   0.03699    0.3942       620       416:  89%
1/1999      6.24G    0.1045    0.2527   0.03699    0.3942       620       416: 100%
1/1999      6.24G    0.1045    0.2527   0.03699    0.3942       620       416: 100%
```

Figure 10. Screenshot of the actual training process record

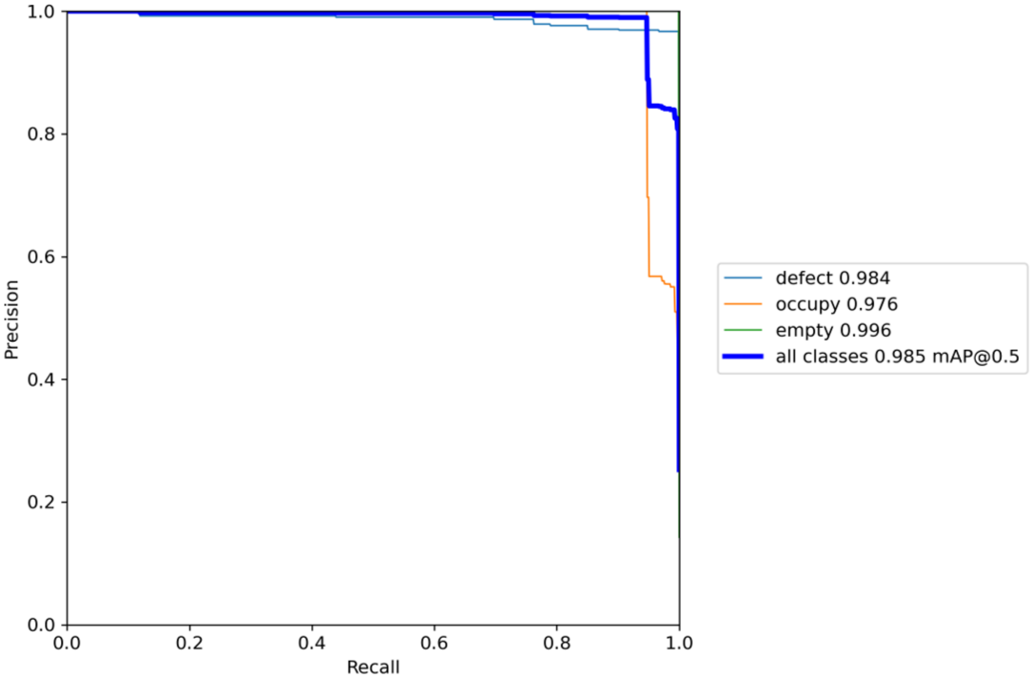
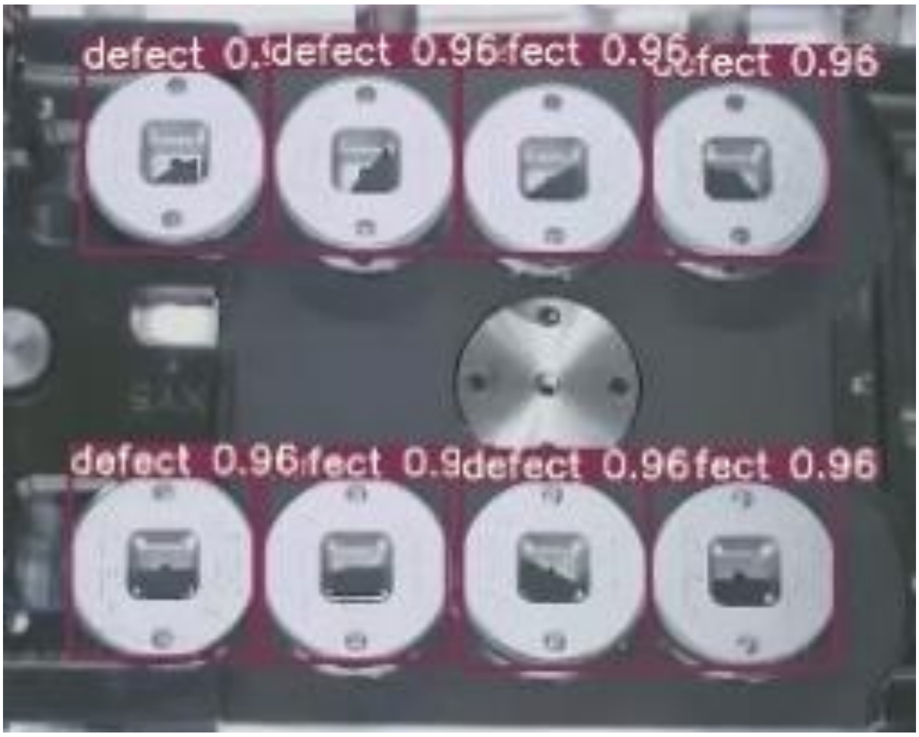


Figure 11. Line chart of mAP with traditional YOLOv5

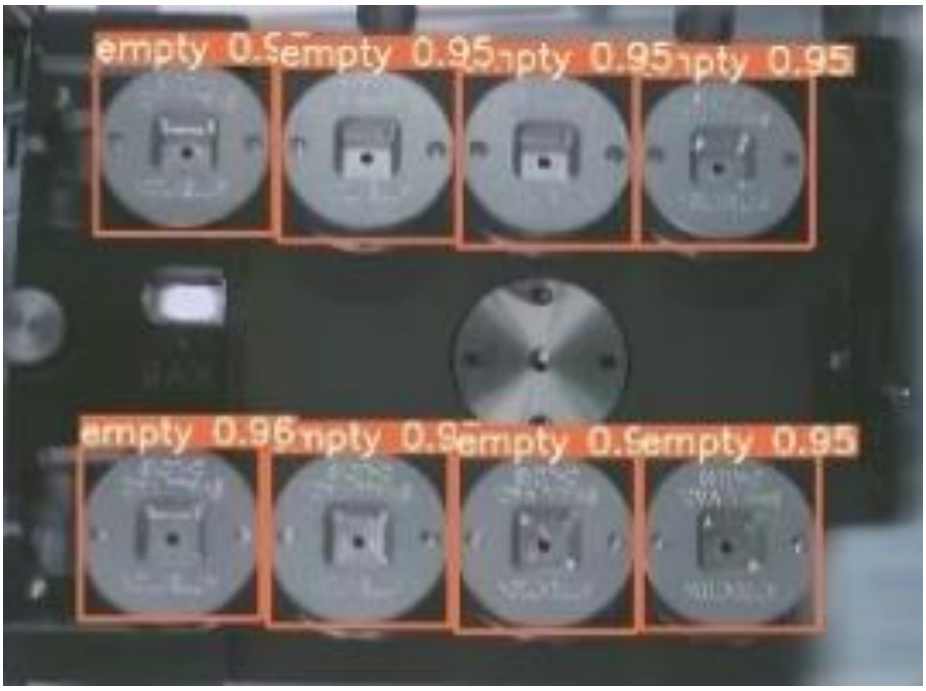
3.3 On-site image sensing and recognition

After completing the training model, a well-trained model can identify pictures, videos, and real-time streaming images. Therefore, we transplants the trained model to an embedded platform, Jetson Xavier NX, for object detection [24] applications. Jetson Xavier NX is a product of NVIDIA with artificial intelligence, and designed for high-speed image recognition, as shown in Figure 12. Once Jetson Xavier NX has completed the object detection, it will also output the image identification result in time, as shown in Figure 13. According to the image identification result, we will evaluate how good the embedded platform is in the execution performance.

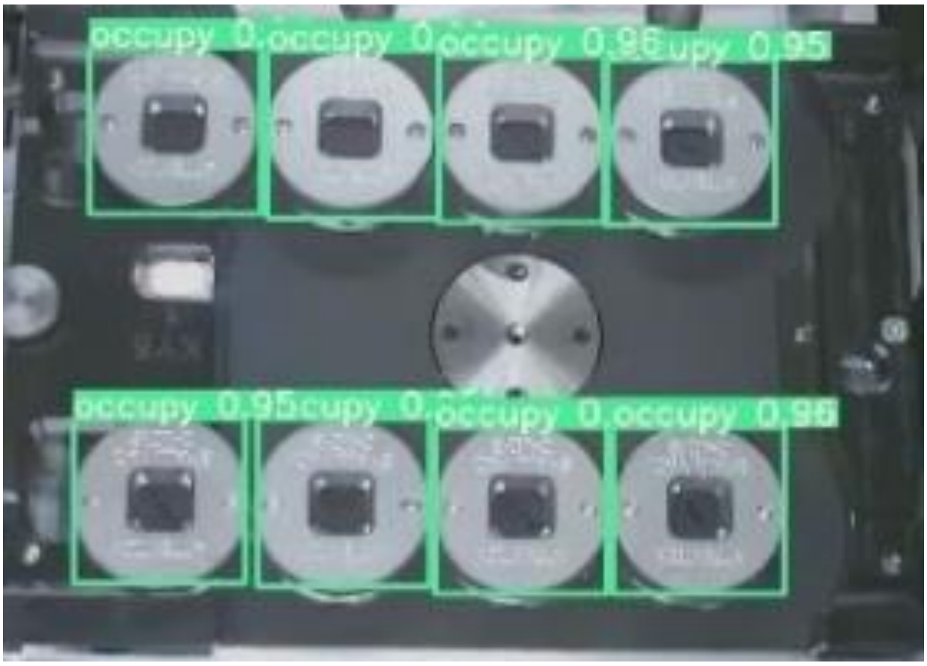
Figure 12. An embedded platform ― Jetson Xavier NX



(a) defective condition



(b) empty condition

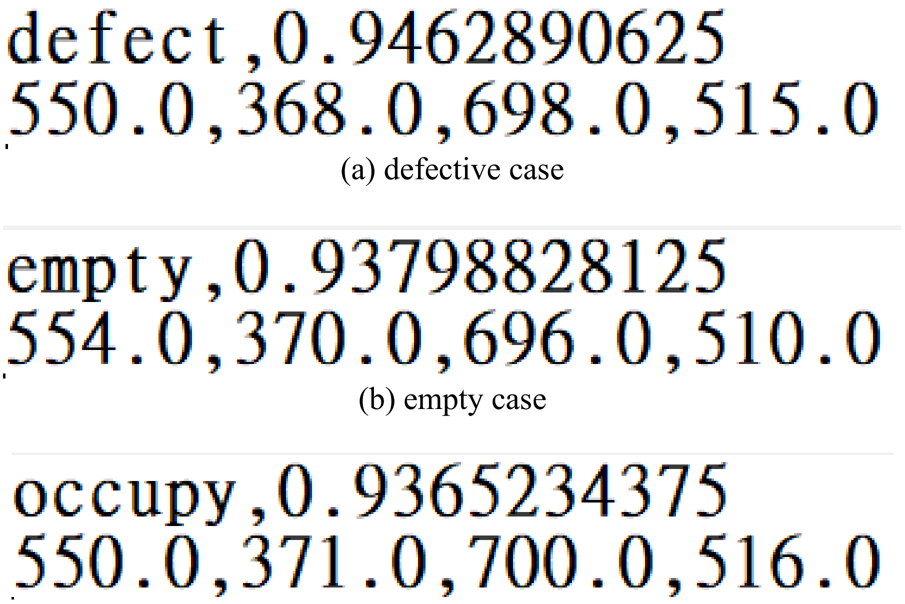


(c) occupied condition

Figure 13. Chip detection showing results

3.4 Spatial location and identification accuracy

This part describes the identification results of chip detection. These results include the type of object to be identified, the accuracy of identified object, and the spatial location of the object, as shown in Figure 14. It is noted that how to deal with the situation of detecting any defective chip at the specified spital location of a chip slot. Once any defective chip has been detected and given a warning message to monitors, people can shutdown or pause the production line operation immediately to avoid the damage of the followed chip put into the same spital location next time. Operators have to learn the guidline of this procedure so as to maintain the production lines secured and operative machines safely.



```
defect,0.9462890625
550.0,368.0,698.0,515.0
```

(a) defective case

```
empty,0.93798828125
554.0,370.0,696.0,510.0
```

(b) empty case

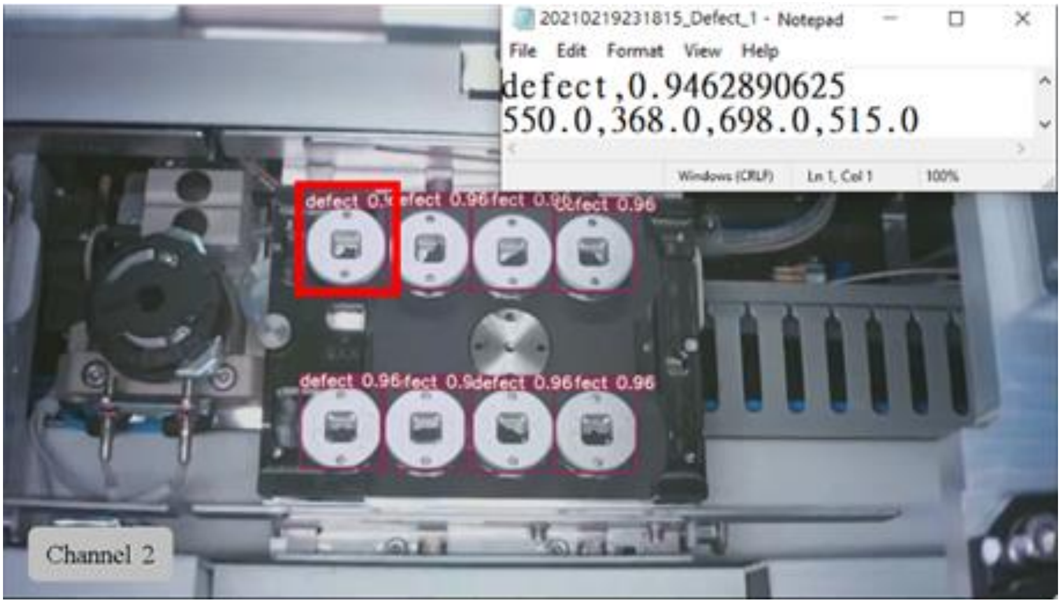
```
occupy,0.9365234375
550.0,371.0,700.0,516.0
```

(c) occupied case

Figure 14. Spatial location and the accuracy of identified objects

3.5 On-site detection of chip contour

This section utilizes the information about the chip detection in detail to establish a warning system for people. As the detection of any defective chip has occurred, people must stop the operative machine immediately to avoid the unexpected loss. With the assistance of chip inspection, the system can detects any defective chip at once and automatically send a warning message to monitor, as shown in Figure 15. The notebook records the information about the defective chip, and its corresponding actual spactial position as indicated in the box marked by color red. For the convenience, we number the chip slots initially. The way of numbering slots is from left to right and top to bottom, as shown in Figure 16. Technically speaking, system set the file name of notebook to the current timestamp to let people inuitively understand when the case happened and which chip slot has a defective one, as shown in Figure 17. This way is not only to show the timestamp directly for peolpe, but it also saves a lot of time for checking which chip slot.


20210219231815_Defect_1 - Notepad
File Edit Format View Help
Windows (CRLF) Ln 1, Col 1 100%
defect 0.9
defect 0.96
fect 0.96
Channel 2

```
defect,0.9462890625
550.0,368.0,698.0,515.0
```

Figure 15. Display information about the exact location

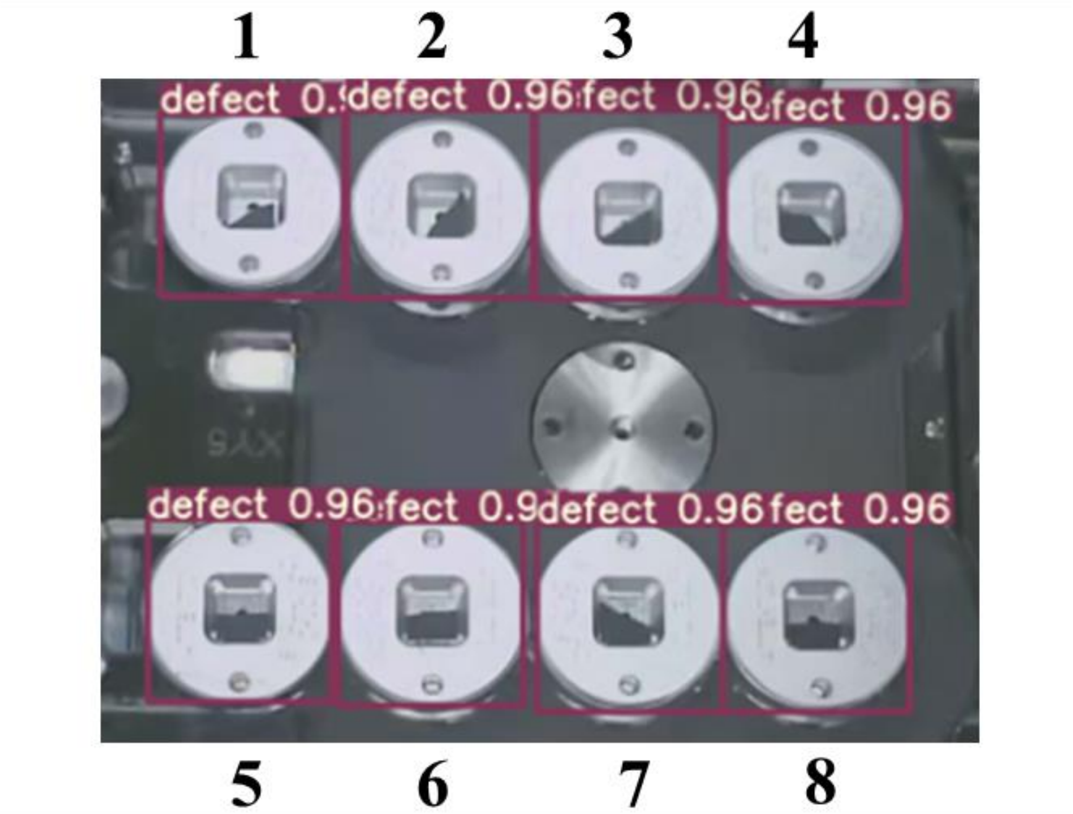
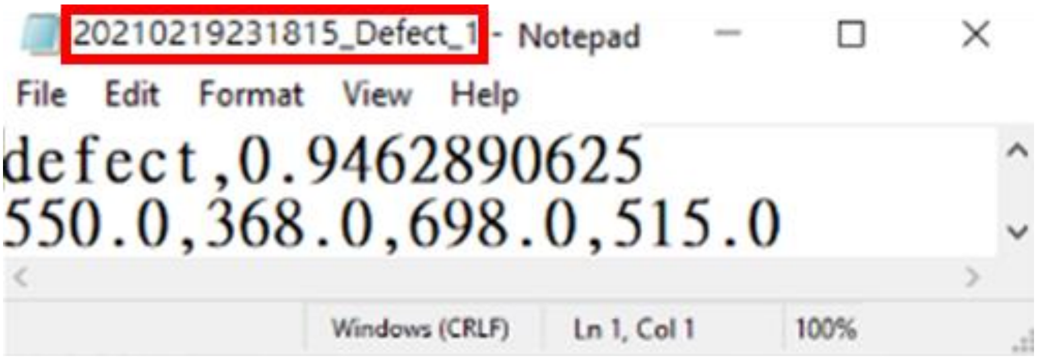


Figure 16. Number all the chip slots



```
20210219231815_Defect_1 - Notepad
File  Edit  Format  View  Help
defect,0.9462890625
550.0,368.0,698.0,515.0
Windows (CRLF)   Ln 1, Col 1   100%
```

Figure 17. Provide timestamps and slot numbers of all the chip slots

3.6 Modification of the YOLOv5 network architecture

The YOLOv5 network architecture is modified to reduce the number of computations required for feature extraction and the number of parameters used to generate valuable features. The main modules use the backbone of MobileNetV3 [18] and the SE module of SENet [20] to replace the whole traditional backbone and call it M3-YOLOv5 model, as shown in Figure 18 to Figure 21.

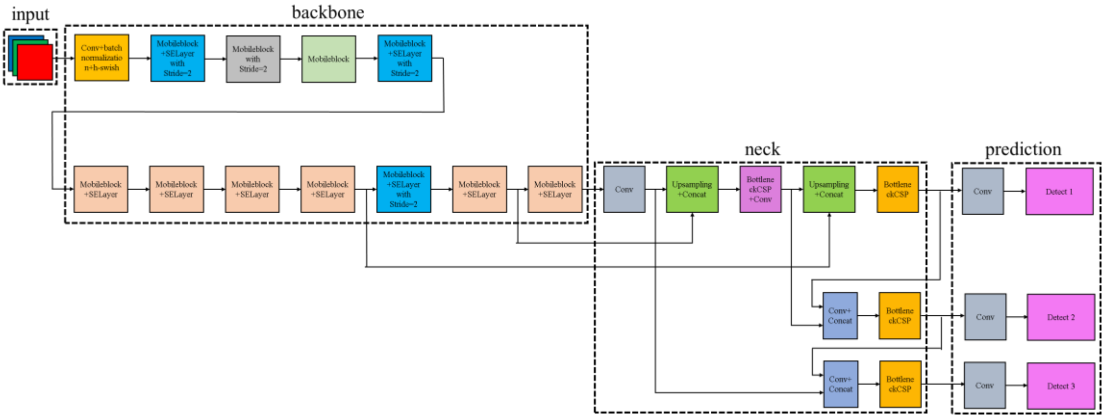
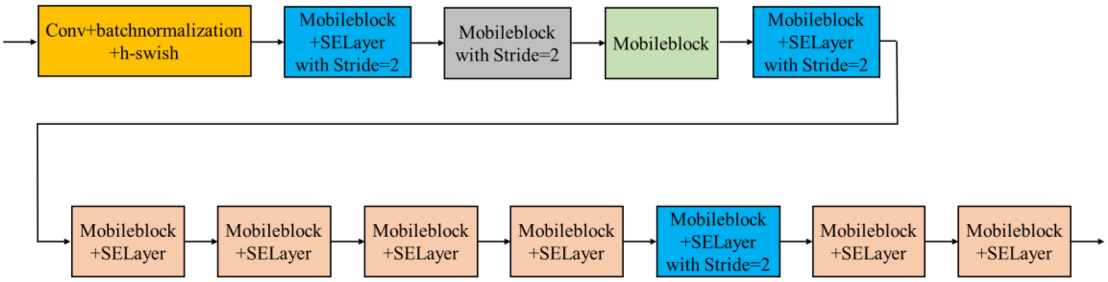


Figure 18. The architecture of the M3-YOLOv5 model



Conv+batchnormalization +h-swish
Mobileblock +SELayer with Stride=2
Mobileblock with Stride=2
Mobileblock
Mobileblock +SELayer with Stride=2
Mobileblock +SELayer
Mobileblock +SELayer
Mobileblock +SELayer
Mobileblock +SELayer
Mobileblock +SELayer with Stride=2
Mobileblock +SELayer
Mobileblock +SELayer

Figure 19. The backbone structure of the M3-YOLOv5 model

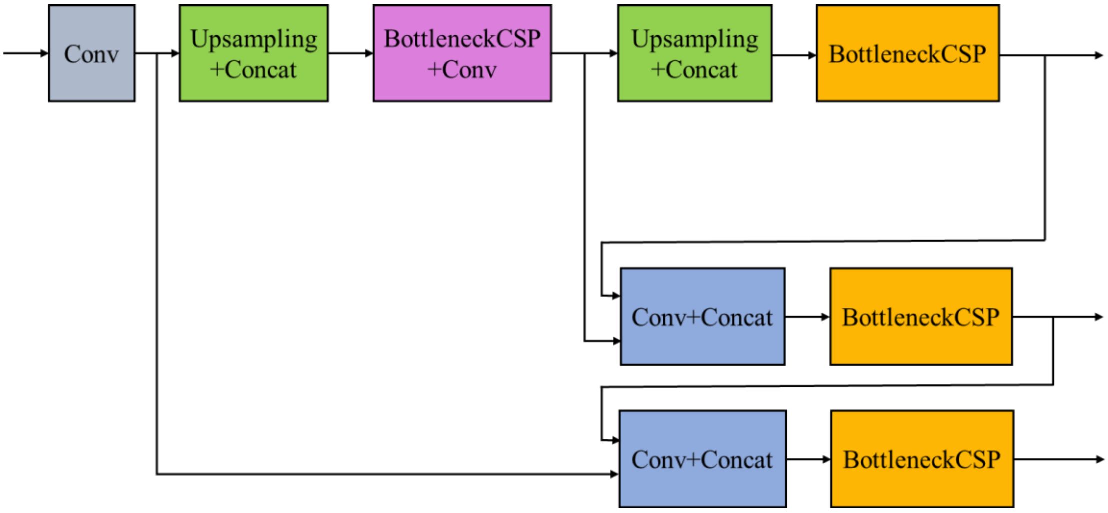


Figure 20. The neck structure of the M3-YOLOv5 model

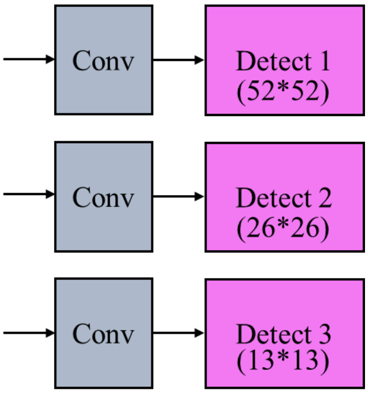


Figure 21. The prediction structure of the M3-YOLOv5 model

3.7 Modification of the YOLOv5 network architecture

The identification performance of the improved M3-YOLOv5 model is evaluated, and the overall average accuracy obtained by training with the improved model is shown in Figure 22.

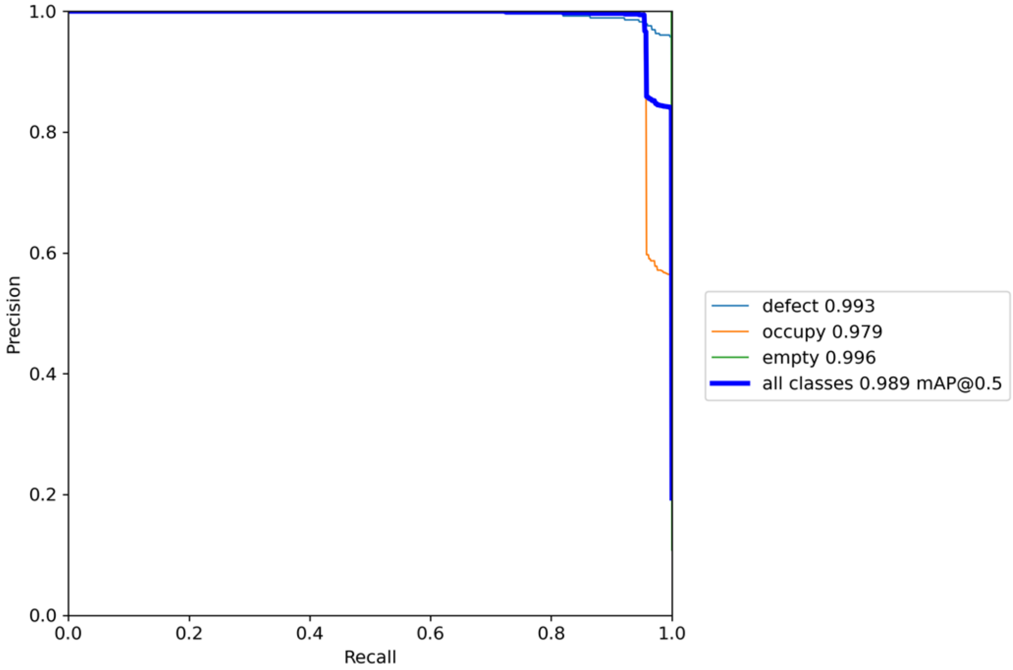


Figure 22. Line chart of mean average precision of M3-YOLOv5

Chapter 4. Experimental Results and Discussion

4.1 Experimental environment

The training environment is mainly implemented in Workstation, and the test environment is based on the Jetson Xavier NX embedded platform, as shown in Table 1. The software copyright is shown in Table 2.

Table 1. Hardware specifications

Resource	Workstation	Jetson Xavier NX
GPU	NVIDIA GeForce RTX 2080 Ti	384-core NVIDIA Volta™ GPU with 48 Tensor Cores
CPU	Intel (R) Core(TM) i9-7900X CPU @ 3.30GHz	6-core NVIDIA Carmel ARM®v8.2 64-bit CPU 6MB L2 + 4MB L3
Memory	96GB	8 GB 128-bit LPDDR4x @ 1600 MHz 51.2GB/s
Storage	256GB*1(SSD) 2TB*1(HDD)	16 GB eMMC 5.1

Table 2. Software copyright

Software	Copyright
labelImg	Free/License
Anaconda® Individual Edition	Free/License
Jupyter Notebook	Free/License
PyTorch	Free/License
JetPack	License

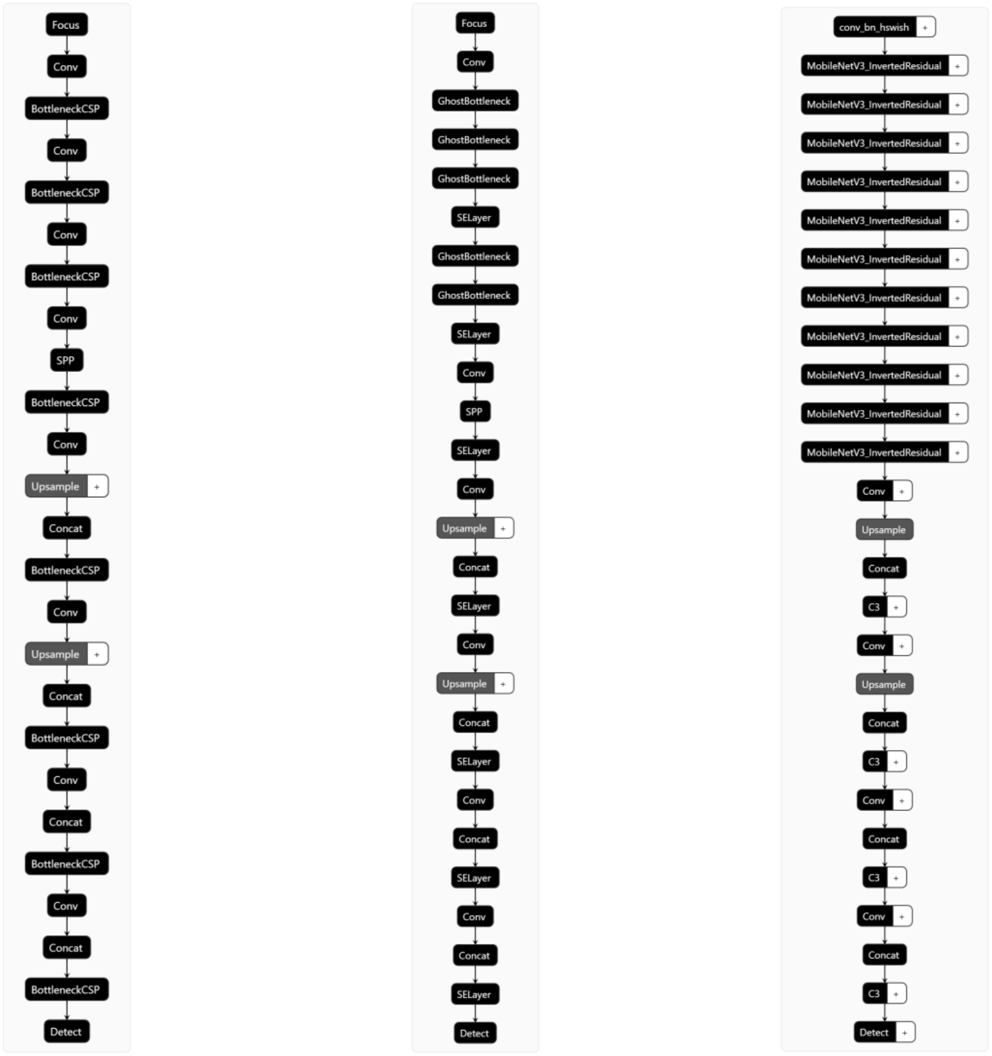
4.2 Experimental design

We performed two experiments of the object detection effect, and the operating speed of the trained YOLOv5 model on the embedded platform, Jetson Xavier NX in this section. Then, we compared the performance of the two improved lightweight YOLOv5 models and the traditional YOLOv5 model.

4.3 Experimental results

4.3.1 Demonstrating the three YOLOv5-related architectures

Here shown the different feature extraction function between the traditional YOLOv5 model and the backbone of MobileNetV3 published in 2019 proposed as a key feature extraction. We have demonstrated the differences among the three YOLOv5-related architectures, the traditional YOLOv5, the improved GSEH-YOLOv5, and the improved M3-YOLOv5 as shown in Figure 23.



(a) Traditional YOLOv5 (b) GSEH-YOLOv5 (c) M3-YOLOv5

Figure 23. Architectures of three YOLOv5-related models

4.3.2 Estimation of modele training time and video inference time

With the Jetson Xavier NX embedded platform, here mainly estimated the time-consuming of training the traditional YOLOv5 model and the modified M3-YOLOv5 model given the same training data set. The test of a video with 1805 frames proceeded to record the time taken for object detection and image recognition for each frame. Eq. (1) Calculate the Average Inference Time (AIT) of three different YOLOv5 models for each image under the same test video. In Eq. (1), VIT_{ijk} represents the time to infer the use of the test video, and FN represents the total number of frames of the test video.

$$AIT_{ijk} = \frac{VIT_{ijk}}{FN}, where\ i = 1,2,\ldots,l, j = 1,2,\ldots,m, k = 1,2,\ldots,n \qquad (1)$$

Technically speaking, we set the parameters of training model as follows: the input image size 416x416, the batch size 64, and the number of iterations 300. After the experiment, the estimated results are listed in Table 3. In Table 3, the first column represents the training time for three different YOLOv5 models based on the same parameter settings. The second column stands for the time consuming of object detection and image recognition according to the total of 1805 image frames. The third column is the average time of object detection and image recognition for each frame.

Table 3. Training and inference times (unit: second)

Method	YOLOv5	GSEH-YOLOv5	M3-YOLOv5
Training	4167.7	4136.4	2224.8
Inference	199.528	179.541	206.088
Average	0.0321	0.0256	0.0365

4.3.3 Real-time detection speed and recognition accuracy

The performance of real-time object detection depends on the number of captured and computable frames per second and the accuracy of image recognition. Equation (2) is used to calculate the number of frames per second with which three YOLOv5-related models can detect objects in real-time, where $RAIT_{ijk}$ is the time required for each image in the real-time image source from the camera with 480×640 resolution. Equation (3) is used to calculate the accuracy of the three YOLOv5-related models, where c_{ijk} represents the identified categories and APc_{ijk} represents the accuracy of each class.

$$FPS_{ijk} = \frac{1}{RAIT_{ijk}}, where\ i = 1,2,\ldots,l, j = 1,2,\ldots,m, k = 1,2,\ldots,n \qquad (2)$$

$$mAP_{ijk} = \frac{APc_{ijk}}{c_{ijk}}, where\ i = 1,2,\ldots,l, j = 1,2,\ldots,m, k = 1,2,\ldots,n \qquad (3)$$

Equation (2) is used to calculate the speed in the real-time object detection with the Jetson Xavier NX embedded platform. After that, Eq. (3) is used to calculate the average accuracy of the three YOLOv5-related models after training with the same parameters, as shown in Table 4.

Table 4. Speed and accuracy of models

Method	YOLOv5	GSEH-YOLOv5	M3-YOLOv5
Speed (fps)	22.222	27.397	25
Accuracy (%)	98.5	97.5	98.9

4.3.4 Operational cost

The number of parameters used and the number of computations considerably vary among the three YOLOv5-related models, as shown in Table 5.

Table 5. Numbers of parameters and FLOPs of models

Method	YOLOv5	GSEH-YOLOv5	M3-YOLOv5
Parameters (#)	7251912	4182136	3205296
FLOPs (GFLOPs)	16.8	6.9	6.0

4.4 Discussion

The model size of the traditional YOLOv5 is about 14.4 MB. The experimental results show that the traditional YOLOv5 model can perform real-time object detection in an embedded platform Jetson Xavier NX at the speed of 22.222 fps and obtain the image recognition accuracy of 98.5%. On the other hand, the model size of the GSEH-YOLOv5 is 8.3 MB that is 42.4% less than that of the traditional YOLOv5 model, and it can perform real-time object detection in an embedded platform Jetson Xavier NX at the speed of 27.397 fps that is 23.3% higher than that of the traditional YOLOv5 model did. However, its image recognition accuracy is 97.5% that is 1% lower than that of the traditional YOLOv5 model. Our proposed approach, the M3-YOLOv5 model, with the model size of 6.5 MB that is 54.9% less than that of the traditional YOLOv5 model, and it can carry out real-time object detection in an embedded platform Jetson Xavier NX at thespeed of 25 fps that is 12.6% higher than that of the traditional YOLOv5 model. It's image recognition accuracy is 98.9% that is 0.4% higher than that of the traditional YOLOv5 model. As a result, our propose approach outperforms the others.

Chapter 5. Conclusion

In this paper, the object-tracking algorithm with the improved M3-YOLOv5 is used to perform real-time recognition of a chip contour and detect whether it is defective. This study evaluated the performance about the recognition accuracy and detection speed between the proposed approach, GSEH-YOLOv5, and the traditional one. For real-time object detection and image recognition using Jetson Xavier NX embedded platform, the experiments show that the proposed approach achieves better recognition accuracy and detection speed than that of the traditional one. When compared with GSEH-YOLOv5, the proposed approach is little efficient in speed because it sacrifices the accuracy to increase its detection speed. Finally, the proposed approach outperforms the other methods significantly.

References

[1] R. Girshick, "Fast R-CNN," 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 2015, pp. 1440-1448.

[2] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," Proceedings of the 28th International Conference on Neural Information Processing Systems, pp. 91–99, 2015.

[3] K. He, G. Gkioxari, P. Dollár and R. Girshick, "Mask R-CNN," 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 2017, pp. 2980-2988.

[4] C. Wang, H. Mark Liao, Y. Wu, P. Chen, J. Hsieh and I. Yeh, "CSPNet: A New Backbone that can Enhance Learning Capability of CNN," 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Seattle, WA, USA, 2020, pp. 1571-1580.

[5] T. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan and S. Belongie, "Feature Pyramid Networks for Object Detection," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 2017, pp. 936-944.

[6] S. Liu, L. Qi, H. Qin, J. Shi and J. Jia, "Path Aggregation Network for Instance Segmentation," 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 2018, pp. 8759-8768.

[7] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," IEEE Conference on Computer Vision and Pattern Recognition, pp. 779–788, 2016.

[8] J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," IEEE Conference on Computer Vision and Pattern Recognition, pp. 6517-6525, 2017.

[9] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778, 2016.

[10] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement,". arXiv:1804.02767, 2018.

[11] A. Bochkovskiy, C.-Y. Wang and H.-Y. Mark Liao, "YOLOv4: Optimal speed and accuracy of object detection", arXiv:2004.10934, 2020.

[12] V. S. Marco, B. Taylor, Z. Wang, Y. Elkhatib, "Optimizing Deep Learning Inference on Embedded Systems Through Adaptive Model Selection," arXiv:1911.04946[cs], Nov. 2019.

[13] Y. Sun, C. Wang and L. Qu, "An Object Detection Network for Embedded System," 2019 IEEE International Conferences on Ubiquitous Computing &

Communications (IUCC) and Data Science and Computational Intelligence (DSCI) and Smart Computing, Networking and Services (SmartCNS), Shenyang, China, 2019, pp. 506-512.

[14] F. Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 2017, pp. 1800-1807.

[15] A.G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, et al., "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," arXiv:1704.04861 [cs], Apr. 2017.

[16] X. Chen et al., "A Light-Weighted CNN Model for Wafer Structural Defect Detection," IEEE Access, vol. 8, pp. 24006-24018, 2020.

[17] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov and L. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 4510-4520.

[18] A. Howard et al., "Searching for MobileNetV3," 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 1314-1324.

[19] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, "Attention Is All You Need," arXiv:1706.03762 [cs], Jun. 2017.

[20] J. Hu, L. Shen and G. Sun, "Squeeze-and-Excitation Networks," 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 2018, pp. 7132-7141.

[21] Bao-Rong Chang, Hsiu-Fen Tsai, Chia-Wei Hsieh, and Mo-Lan Chen, "Chip Contour Detection Based on Real-Time Image Sensing and Recognition, " Sensors and Materials, Vol. 34, No. 1, pp. 1-12, 2022. Published in advance: June 15, 2021.

[22] M. Saqlain, Q. Abbas and J. Y. Lee, "A Deep Convolutional Neural Network for Wafer Defect Identification on an Imbalanced Dataset in Semiconductor Manufacturing Processes," IEEE Transactions on Semiconductor Manufacturing, vol. 33, no. 3, pp. 436-444, 2020.

[23] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al., "Pytorch: An imperative style, high-performance deep learning library," In Advances in neural information processing systems, pp. 8026-8037, 2019.

[24] V. Mazzia, A. Khaliq, F. Salvetti and M. Chiaberge, "Real-Time Apple Detection System Using Embedded Systems With Hardware Accelerators: An Edge AI Application," in IEEE Access, vol. 8, pp. 9102-9114, 2020.