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

報告者:謝佳衛
指導教授: 張保榮 教授

OUTLINE

1. Introduction
2. Related Work
3. Research Method
4. Experimental Results and Discussion
5. Conclusion

1. Introduction

Introduction

- 隨著工業4.0的推動下，創新技術對工廠進行重大改革。將大數據分析、雲端計算、5G、人工智慧及自動化系統等引入工廠，可顯著提高工廠的生產能力，為公司帶來最具經濟價值的同業競爭力。

- 物件偵測技術分為兩種模式——單階段和兩階段。單階段結合了定位和分類，兩階段將定位和分類分開。典型的單階段物件偵測包含SSD和YOLO，典型的兩階段目標檢測模型包含 Fast R-CNN、Faster R-CNN 和 Masked R-CNN。

Introduction(cont.)

- 傳統的單階段物件偵測具有速度快與準確性低的特性，兩階段物件偵測則是具有速度慢與準確性高的特性。差別在於框選目標是否需要進行分類，單階段物件偵測會先判斷框選目標是否為一個物體再進行分類，兩階段物件偵測則是將框選目標全部進行分類所以較為費時。

- 產線在自動輸送晶片的過程中，如果生產機台意外壓壞目前在晶片槽內的晶片，下一輪將晶片放在同一個位置時，會造成新一輪的晶片損壞。若沒有即時警示作業員停止機台移除破損晶片，將會造成不間斷的生產損失，甚至可能導致機台吸嘴偏移與受損。因此首要目標即是利用速度快的即時物件偵測能夠在機台運行時即時反應晶片槽情形。

Objective

- 本研究利用YOLOv5物件偵測模型部署於NVIDIA Jetson Xavier NX 來進行本地端的晶片槽晶片狀態檢測。不過當我們使用YOLOv5模型進行物件偵測時，必須具備即時運行的能力。對於傳統YOLOv5模型的執行速度而言，具有優化空間。因此，本研究的目的就是在優化程式執行速度的同時要能夠維持晶片狀態檢測的準確性。

- 本研究改良傳統YOLOv5物件偵測演算法，稱為M3-YOLOv5。在晶片槽晶片狀態檢測的表現上將傳統YOLOv5與M3-YOLOv5進行效能評估的比較。除了提高晶片狀態檢測的準確性，同時也減少計算資源的消耗，帶來程式執行速度的提升。最後，加入GSEH-YOLOv5與兩者進行效能比較來分析三者之間的優劣。

2. Related Work

You Look Only Once v1 (YOLOv1)

- Redmon和Farhadi等人提出的YOLOv1核心概念是將輸入影像視為具有相同寬度和高度的網格，並預測每個網格中的物件。在每個網格中使用兩個邊界框預測物件，而且只能預測同一個網格內類別機率最高的一種物件。

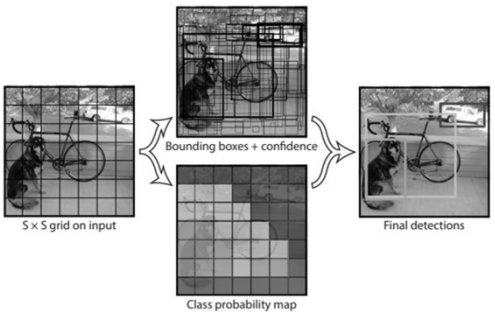


Figure 1. YOLOv1模型運作流程

You Look Only Once v1 (YOLOv1) (cont.)

- YOLOv1模型由24個卷積層與2個全連接層組成，其中1x1卷積用於減少特徵圖的通道數量來減少計算量。其中將所有邊界框對於同一類別的機率進行排列，透過後處理的方法非極大值抑制 (NMS)進行邊界框的篩選。

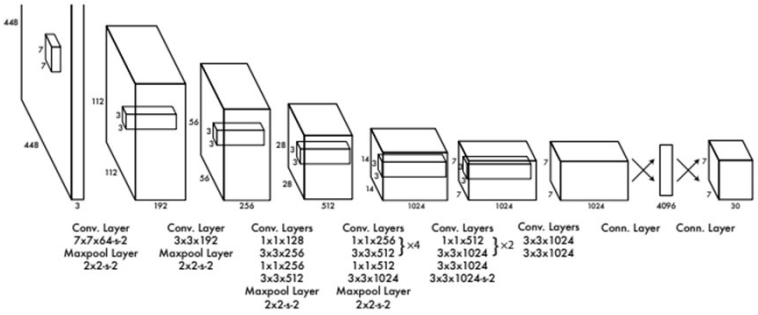


Figure 2. YOLOv1模型網路架構圖

You Look Only Once v2 (YOLOv2)

- Redmon等人為YOLOv1提出改進的方法，稱為YOLOv2。
- 改進目的是提高模型的速度和準確性，主要參考VGG16的網路架構進行優化將YOLOv2使用的網路架構稱作Darknet-19。
- 加入批次標準化和錨框機制不但大幅提升模型的執行速度外，也能夠顯著提升準確性。

Type	Filters	Size/Stride	Output
Convolutional	32	3×3	224×224
Maxpool		$2 \times 2/2$	112×112
Convolutional	64	3×3	112×112
Maxpool		$2 \times 2/2$	56×56
Convolutional	128	3×3	56×56
Convolutional	64	1×1	56×56
Convolutional	128	3×3	56×56
Maxpool		$2 \times 2/2$	28×28
Convolutional	256	3×3	28×28
Convolutional	128	1×1	28×28
Convolutional	256	3×3	28×28
Maxpool		$2 \times 2/2$	14×14
Convolutional	512	3×3	14×14
Convolutional	256	1×1	14×14
Convolutional	512	3×3	14×14
Convolutional	256	1×1	14×14
Convolutional	512	3×3	14×14
Maxpool		$2 \times 2/2$	7×7
Convolutional	1024	3×3	7×7
Convolutional	512	1×1	7×7
Convolutional	1024	3×3	7×7
Convolutional	512	1×1	7×7
Convolutional	1024	3×3	7×7
Convolutional	1000	1×1	7×7
Avgpool		Global	1000
Softmax			

Figure 3. YOLOv2網路架構Darknet-19

You Look Only Once v3 (YOLOv3)

- Redmon等人為了優化YOLOv2的準確性，在YOLOv2的基礎上進行改良，稱為YOLOv3。

- 加入殘差網路與多尺度檢測的概念，YOLOv3雖然加深網路架構造成執行速度的降低，不過能夠大幅優化YOLOv2準確性的不足。

	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Figure 4. YOLOv3網路架構Darknet-53

You Look Only Once v4 (YOLOv4)

- Bochkovskiy等人在YOLOv3的基礎上進行各部分的改進，稱為YOLOv4。其中提升效果最為明顯的改進則是加入CSPNet的架構，藉由CSP架構能夠在原先的Darknet-53中擷取更為豐富的特徵訊息之外，同時能夠顯著降低計算量。

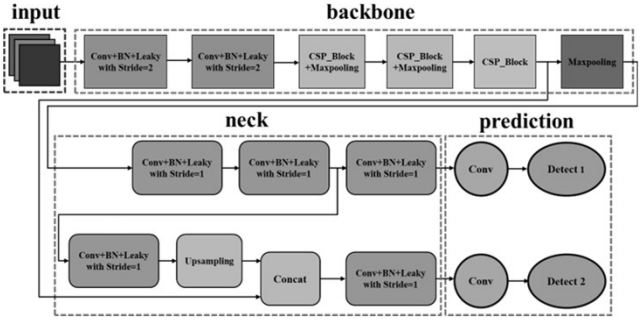


Figure 5. YOLOv4模型架構圖

You Look Only Once v5 (YOLOv5)

- Jocher等人提出的YOLOv5是在PyTorch上進行開發的，而不像前四版是在Darknet上進行開發。主要特色為加入Focus結構在架構前防止初期特徵訊息的流失，以及自適應圖片縮放讓YOLOv5進行推論時可以減少像素填充的數量而加速物件偵測的執行速度。

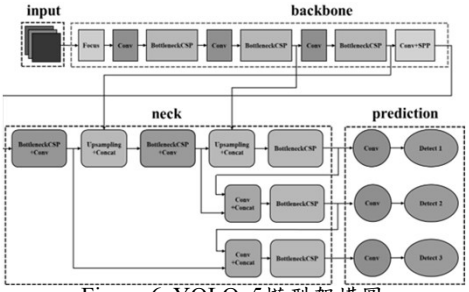


Figure 6. YOLOv5模型架構圖

MobileNet v1

- Howard等人提出的MobileNet v1中，使用新穎的深度可分離卷積技術可大幅減少傳統卷積的運算量。

Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
$5\times$ Conv dw / s1	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$
Conv dw / s2	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
Avg Pool / s1	Pool 7×7	$7 \times 7 \times 1024$
FC / s1	1024×1000	$1 \times 1 \times 1024$
Softmax / s1	Classifier	$1 \times 1 \times 1000$

Figure 7. MobileNet v1網路架構

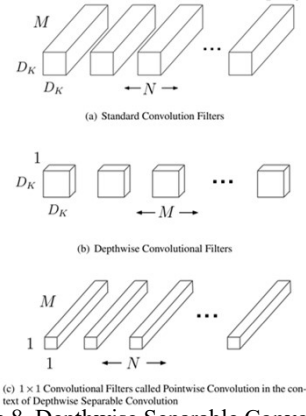


Figure 8. Depthwise Separable Convolution

MobileNet v2

- Sandler等人在MobileNet v1的基礎上優化，稱為MobileNet v2。其中加入殘差網路的概念並且加以優化而提出反向殘差，透過反向殘差可以達到在低維度傳遞資料，在高維度擷取特徵的目的。

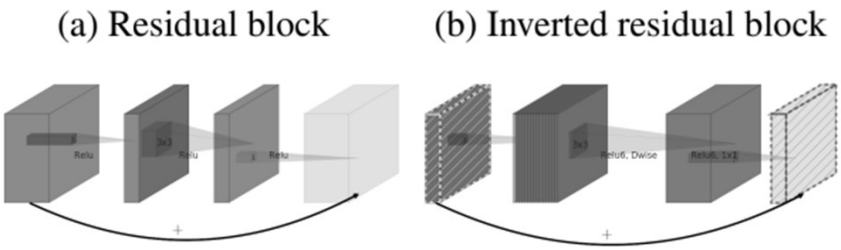


Figure 9. Inverted Residual Block

MobileNet v3

- Howard等人提出的MobileNet v3在架構上進行優化，加入網路架構搜尋以及注意力機制，分別進行整體網路架構的優化與強化重要特徵的影響力並減輕不重要特徵的影響力。

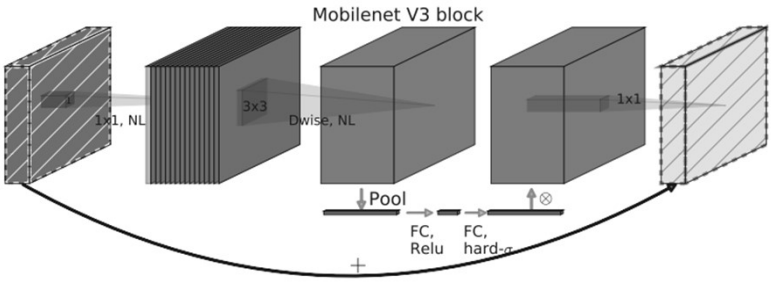


Figure 10. MobileNet v3 Block

3. Research Method

Data preparation and labeling

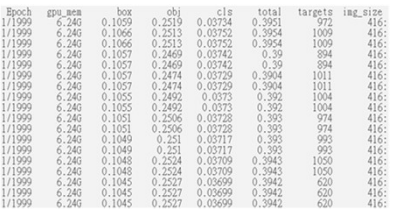
- YOLO物件偵測是屬於監督式學習，因此必須透過正確答案餵給模型進行學習，而正確答案則是藉由人工方式將目標物進行標註。
- LabelImg開源軟體可用於將目標物進行標註的工作，內容包含目標物的名稱及位於圖片中的位置。

Figure 11. LabelImg標註過程

Model training and performance evaluation

- 訓練與表現評估階段

1. 準備YOLO格式的資料集，將資料集分為訓練、驗證及測試集。
2. 開始模型訓練前設定超參數，經過反覆測試使模型具最佳表現。
3. 模型訓練完成後利用測試集驗證模型的表現。以物件偵測任務來說，需要考量兩項因素分別是執行速度FPS與準確度mAP。



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

Figure 12. 訓練階段過程

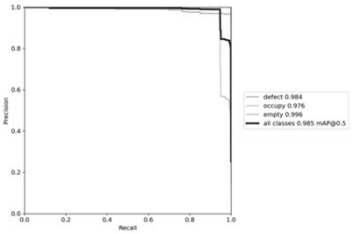
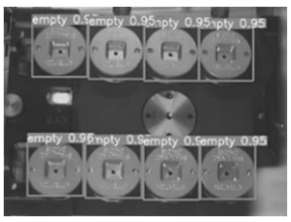


Figure 13. 訓練模型準確度

On-site image sensing and recognition

- 根據使用情境分為三種狀況，一種是晶片槽沒有晶片在內，另一種是晶片槽有正常完整的晶片在內，最後一種則是晶片槽內的晶片因為外力因素導致晶片破損而留下的晶片殘留物。
- 利用YOLOv5物件偵測進行的晶片槽晶片狀態檢測，右圖分別是defective、empty以及occupied的結果。



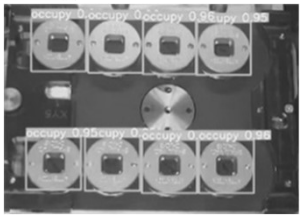


Figure 14. 晶片槽晶片狀態檢測

Spatial location and identification accuracy

- 接著將物件偵測後獲得的資訊包含辨識物件的類別、物件分類的準確率、物件的位置以及損壞晶片出現的時間點做為警示訊息的依據。
- 右圖分別代表單一晶片槽的三種情形呈現，內容包含物件種類、判斷為此種類的信心程度以及框選位置的座標。

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 15. 顯示框選物件的位置和偵測物件的信心程度

On-site detection of chip contour

- 掌握檢測晶片的詳細資訊可以建立一套晶片警示系統，用來即時提醒作業員機台上有狀況需要進行排除。因為當出現損壞的晶片時，必須能夠立刻停止機器運轉，以避免損失擴大。
- 因此，在晶片警示系統的輔助下，只要晶片狀態檢測系統偵測到晶片損壞時，晶片警示系統就會自動向使用者發送與損壞晶片相關的訊息。
- 記事本包含有關損壞晶片的詳細訊息，其對應的實際晶片位置是紅色框的位置。

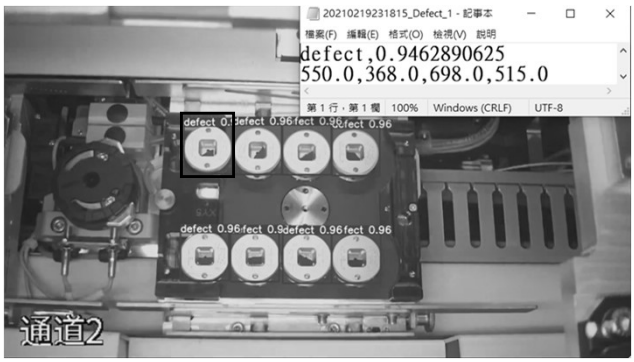


Figure 16. 顯示特定晶片槽位置的訊息

On-site detection of chip contour (cont.)

- 為了作業員檢視方便，我們首先將晶片槽進行編號。編號方式為由左而右，由上到下。
- 接著將記事本的檔案名稱設定為檢測出損壞晶片時當下的時間點，並且告訴作業員是位於第幾個晶片槽出現損壞晶片，讓作業員不需要每一個晶片槽皆進行檢視，節省時間。

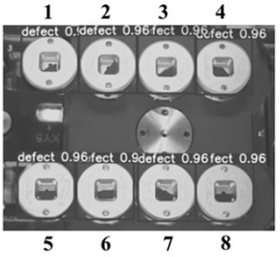


Figure 17. 為晶片槽進行編號

Figure 18. 提供時間戳記及晶片槽編號

Modification of the YOLOv5 network architecture

- 修改 YOLOv5 網路架構，以MobileNetV3架構取代傳統的backbone部分，稱為M3-YOLOv5。
- 目的在於減少特徵提取所需的計算量和用於保留較有價值的特徵。

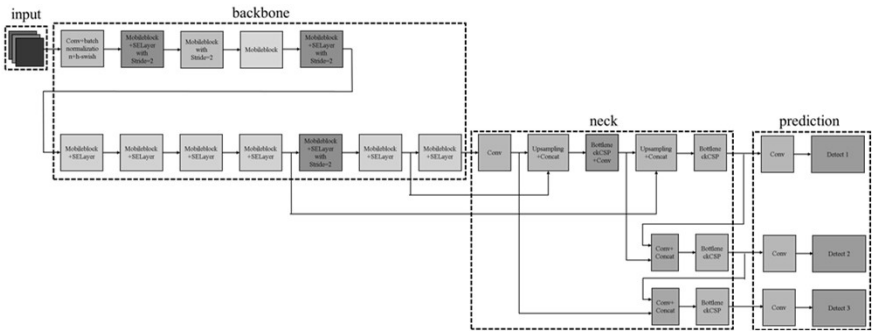


Figure 19. M3-YOLOv5 模型架構圖

Average accuracy of M3-YOLOv5 model

- 測試M3-YOLOv5模型的晶片槽狀態辨識結果，並評估該模型的性能。
- mAP折線圖由兩項指標組成，分別為Precision及Recall。Precision代表預測為目標物且實際上也確實為目標物的比例，Recall代表實際為目標物也確實被預測為目標物的比例。

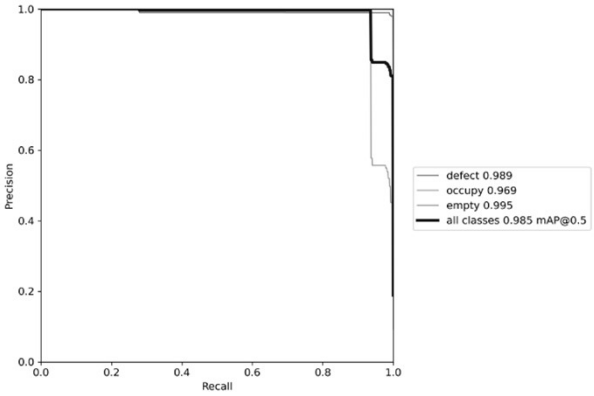


Figure 20. M3-YOLOv5 模型的mAP折線圖

4. Experimental Results and Discussion

Experimental environment

- 本研究使用GPU 工作站快速進行模型訓練，以減少傳統 CPU 訓練模型所花費的處理時間。並且將模型部署至Jetson Xavier NX中，達成本地端運算。

Figure 21. GPU 工作站

Figure 22. Jetson Xavier NX 嵌入式平台

Experimental environment(cont.)

- 本研究使用開源軟體來建立運行環境

Table 1. 開源軟體套件

Software	Version
labelImg	1.8
Anaconda® Individual Edition	4.9.2
Jupyter Notebook	6.1.4
TensorFlow	2.2
TensorRT	7.2.3
PyTorch	1.6
JetPack	4.5

Experimental design

- 本研究針對三種不同的YOLOv5模型在嵌入式平台Jetson Xavier NX的運行進行操作成本、執行速度和準確度上的比較與分析。
  1. Demonstrating the three YOLOv5-related architectures：呈現三種不同YOLOv5模型的架構。
  2. Estimation of model training time and video inference time：評估三種不同YOLOv5模型的訓練時間與推論時間，分析其優劣。
  3. Real-time detection speed and recognition accuracy：分析晶片檢測系統使用三種不同YOLOv5模型的執行速度與準確度。
  4. Operational cost： 三種不同YOLOv5模型之間的操作成本比較。

Demonstrating the three YOLOv5-related architectures

- 利用模型視覺化工具Netron將傳統YOLOv5 、 GSEH-YOLOv5 及 M3-YOLOv5整體架構進行繪製。

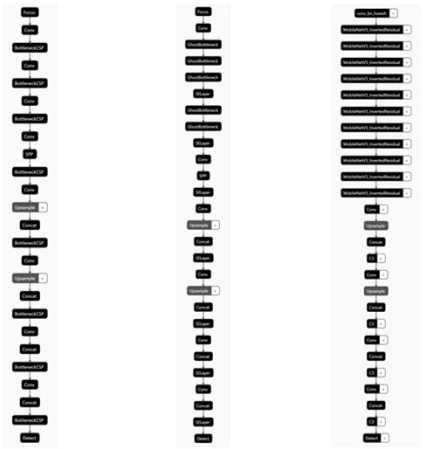


Figure 23. 三種不同YOLOv5模型架構

Estimation of model training time and video inference time

- 主要在嵌入式平台Jetson Xavier NX上測試三種不同的YOLOv5模型對相同訓練資料集訓練所需花費的時間。
- 使用的輸入影像來源為1805幀的測試影片，計算將整部影片推論完畢所需花費的時間。

Table 2. 訓練及推論時間(unit: second)

Method	YOLOv5	GSEH-YOLOv5	M3-YOLOv5
Training	4167.7	4136.4	2224.8
Inference	199.528	179.541	196.088

Real-time detection speed and recognition accuracy

- 即時物件偵測的效能主要會以物件偵測每秒可處理的幀數及準確度兩項數值做為模型的評估指標。其中FPS數值越高反映物件偵測的執行速度越快，準確度百分比越高表示模型越能精準辨識晶片槽的晶片狀態。從實驗數據顯示GSEH-YOLOv5具有較快速的執行速度，不過準確度與傳統YOLOv5相比較低。改良的M3-YOLOv5在執行速度與準確度方面皆能優於傳統YOLOv5。

Table 3.三種不同YOLOv5模型的執行速度與準確度

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

Operational cost

- 使用 thop工具來計算三種不同YOLOv5模型的參數量與計算量。從計算結果可得知，傳統YOLOv5模型的參數量與計算量相對較大。GSEH-YOLOv5與M3-YOLOv5皆可大幅降低傳統YOLOv5模型的參數量與計算量，由此也可反映透過大量降低參數量與計算量確實可以提升物件偵測的執行速度。

Table 4. 三種不同YOLOv5模型的參數量與計算量

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

Discussion

- 降低參數量與計算量將會影響模型的大小。GSEH-YOLOv5模型與傳統YOLOv5模型相比，模型大小較傳統 YOLOv5 模型小42.4%。
- 本研究提出的改良式M3-YOLOv5模型與傳統YOLOv5模型相比，模型大小較傳統 YOLOv5 模型小54.9%。
- 執行速度與準確度方面，GSEH-YOLOv5模型與傳統YOLOv5模型相比，執行速度提升23.3%，準確度減少1%。
- 本研究提出的改良式M3-YOLOv5模型與傳統YOLOv5模型相比，執行速度提升12.5%，準確度提升0.4%。

5. Conclusion

Conclusion

- 本研究應用改良型M3-YOLOv5模型對晶片槽晶片狀態進行即時檢測辨認目前機台上是否有破損晶片的出現。
- 本研究提出的M3-YOLOv5模型在執行速度與準確度部分皆能優於傳統YOLOv5模型。
- 本研究提出的M3-YOLOv5模型與GSEH-YOLOv5 模型相比，雖然GSEH-YOLOv5模型的執行速度快9.6%，不過是犧牲準確度1.4%做為代價。
- 本研究建立晶片警示系統，可在機台出現破損晶片時立即告知作業員發生時間點以及位於八個晶片槽中的第幾個晶片槽有破損晶片的出現，加速排除異常狀況的工作。

Conclusion (cont.)

- 在本研究中，主要使用python程式語言進行撰寫。不過目前是以終端機的方式進行程式執行，所以未來會使用視覺化界面讓使用者能夠更為直覺且友善地使用物件偵測之程式。介面上會包含選擇訓練完成的模型以及即時物件偵測的影像畫面呈現於介面，一旦發現破損晶片就會通知使用者。
- 另外，本研究也將會與時俱進地研究最新版本的YOLOv7物件偵測演算法。透過優化其架構，使執行速度與準確度部分得到顯著提升。讓晶片檢測系統能夠進一步得到效能提升，並且在計算量負擔減少的條件下，甚至可選用成本較低的嵌入式平台進行本地端部署。

簡報結束
感謝聆聽