國立清華大學 電機工程學系

實作專題研究成果摘要

Application of YOLO in Traffic Signs Detection

YOLO 架構之神經網路應用於交通號誌辨識

專題領域： 系統領域

組　　別： B424

指導教授：鄭桂忠 教授

組員姓名：許祐銨 王柏竣 謝秉均

報告摘要

路牌與交通號誌辨識系統的研究在現代交通安全與自動駕駛技術中具有關鍵性的重要性。該技術需具備即時識別限速、禁止轉彎等關鍵道路標誌的能力，以協助駕駛者遵守交通規則，並有效的預防交通事故。尤其在疲勞駕駛或注意力分散的情況下，該系統能夠提供關鍵的輔助功能，提升駕駛安全性。此外，路牌辨識技術是自動駕駛車輛實現安全導航的核心基礎。透過準確辨識交通標誌，自動駕駛車輛能根據規定做出決策，確保在不同駕駛環境中做出正確反應。為了實現即時判斷，該系統必須在本地運行，因此需要在保證判斷準確的前提下考慮模型的大小。不同天氣、光線條件以及全球多樣化標準的挑戰，使得提升判斷精準度、縮短運算時間以及確保模型的彈性變得尤為重要。為此，本研究採用 YOLO v8n 神經網絡架構，並使用 Kaggle 上的交通標誌檢測數據集，訓練和優化路牌及號誌辨識模型。

1. Introduction

YOLO（You Only Look Once）是一種基於深度學習的即時物體檢測演算法，屬於單階段檢測器的典範。與傳統的兩階段檢測器（如 R-CNN 系列）相比，YOLO 將整個物體檢測任務轉化為單次前向傳播過程，在輸入圖像上同時進行位置定位與類別預測。其核心思想是將圖像分割為固定數量的網格，每個網格負責檢測該區域內的物體，並預測邊界框（Bounding Boxes）的座標和物體分類標籤。

YOLO 架構通過卷積神經網絡（CNN）進行特徵提取，並且在最後的輸出層同時產生多個預測框。每個預測框攜帶邊界框的位置信息（中心點座標、寬度、高度）、物體類別的機率分布以及物體存在的置信度。這種方法允許 YOLO 在一次前向運算中生成物體的檢測結果，因此具有極高的推理速度，特別適合於即時性要求較高的應用場景。

YOLO 的高效性源於其端到端的訓練方式，該方式結合了全局上下文信息，避免了像滑動窗口或區域提議（Region Proposals）等耗時步驟。自從 YOLOv1 發表以來，該架構經過多次改進，最新版本如 YOLOv8，進一步提升了檢測的精確度和效能。YOLO 的應用範圍廣泛，從自動駕駛中的目標檢測到工業中的品質檢測，均展現了其強大的即時物體識別能力。

在這個章節，我們將介紹一些 YOLOv8 使用的技術、模型架構、限制以及本專題會使用到的資料前處理方式。

1.1. IOU：

YOLO 與其他 RCNN 模型一樣，使用交集聯集比（Intersection over Union, IoU）來

判定偵測的正確性。IoU 是評估預測邊界框與真實邊界框重疊程度的指標，計算方式為預測框與真實框交集的面積除以兩者聯集的面積。當 IoU 超過某個閾值時，預測結果被視為正確偵測 (True Positive)。這種方法能夠有效量化物體偵測模型的準確度，不僅應用於 YOLO，也廣泛應用於其他物體偵測模型中。

1.2. NMS：

非極大值抑制（Non-Maximum Suppression, NMS）是一種用於物體檢測的後處理技術，旨在消除重複的偵測結果，確保每個物體僅有一個邊界框被保留。NMS 首先會將置信度最高的邊界框放入「確定物件集合」中。接著，NMS 會檢查所有剩餘的邊界框，並選擇與已選出的邊界框具有高 IoU 的邊界框，將這些邊界框進行抑制（即刪除）。通過這樣的處理，每個物體最終僅保留一個邊界框，從而提高了物體檢測的準確性和可靠性。

1.3. Anchor-free Detection Head：

在 YOLOv8 中，放棄了先前 YOLO 系列使用的 anchor boxes，改為基於物體中心點去預測 bounding box，其運作流程為:

(1) **中心點預測：** Anchor-free 設計基於物體的中心點進行偵測，模型直接學習物體的中心點位置 (center point)，不再依賴事先設置好的 anchor boxes。

(2) **邊界框回歸(Bounding Box Regression)：** 在偵測到物體中心後，模型根據該中心點預測邊界框的四個偏移量，分別為上、下、左、右的距離。這四個值共同描述了該框的邊界位置，而不依賴預設的框大小和比例。

(3) **類別預測(Classification)：** 在每個預測位置，模型會產生多個類別分數(Score)，表明該位置可能屬於各個類別的信心度(Confidence)。這樣的設計便於快速確定物體類型。

這樣的設計讓 YOLOv8 可以更靈活地適應不同大小和形狀的物體，因為它就不再受限於預定義的 anchor boxes 框架，並且有以下三點優勢：

- **簡化超參數調整：** 使用 anchor boxes 的模型通常需要針對不同數據集設置不同的 anchor 大小和比例，Anchor-free 模型免除了這一調整過程，讓訓練更便捷，並且對不同類型的數據集有更好的適應性。
- **減少計算量：** 由於不再需要計算 anchor box 與預測框的交集（Intersection over Union, IoU），Anchor-free 的設計能夠減少計算資源，從而提升運算速度。
- **適應性更強：** 由於 Anchor-free 設計直接基於圖像中物體的實際位置進行偵測，無需依賴預設的框架，因此對多尺度物體的偵測效果更好，尤其在小物體偵測上效果顯著提升。

1.4. YOLO 的限制：

儘管 YOLOv8 已改進前面幾代存在的問題，但也因此衍伸出一些其他的問題。

(1) 小物體偵測困難：

雖然已在小物體偵測上有所改進，但在處理小物體或低分辨率影像時，模型的準確度仍然有限，這是因為 YOLOv8 的 Backbone 結構會進行多層下採樣，可能導致小物體特徵在特徵圖中被過度縮小或模糊，無法有效地進行識別。

(2) 對遮擋物體的偵測效果有限：

在應對物體重疊或部分遮擋的情況時，偵測效果可能會顯著下降。因為 YOLOv8 主要依賴於物體中心點的 Anchor-free 檢測方式，當多個物體遮擋或重疊時，無法清晰分辨物體邊界，使得偵測效果下降。

(3) 對異常物體比例的適應性不佳：

YOLOv8 雖然是多尺度的，但對於極端大小的物體（例如過大或過小）仍然表現有限，是因為在預設的網路架構中，特徵提取層對多尺度物體的適應性有其設計範圍，對於超出該範圍的物體（例如遠距離拍攝的微小物體或超近距離拍攝的巨大物體），偵測效果會下降。

(4) 資源需求與硬體依賴：

需要較高性能的硬體支持（如 GPU）以達成實時偵測效果，且對於大批量的高分辨率影像處理時資源需求較高。

(5) 對特定場景適應性有限：

對於場景專用的資料集，或物體具有強背景依賴性時（例如醫學影像或工業偵測），往往需要進行大量的微調或訓練以達到理想效果。

1.5. 模型架構介紹：

YOLOv8 (You Only Look Once version 8)是 YOLO 系列最新的物體偵測模型，改進了先前版本的架構與演算法，使其在精度和速度上有更好的表現。YOLOv8 繼承了 YOLO 系列的核心思想，也就是利用單階段的卷積神經網路 (Convolutional Neural Network, CNN) 結構，在單張圖片的特徵圖上同時進行分類與定位，而這個結構可以分為四個主要的模組：

(1) Backbone (CSPDarknet53)：

YOLOv8 採用了 **CSPDarknet53** 作為 Backbone 模組，其目的在於高效地提取圖像的特徵。CSPDarknet53 的主要特點有：

- **Cross Stage Partial Network (CSPNet)：** 這種設計通過將特徵圖分為兩個分支，一個繼續進行卷積計算，另一個則跳過部分卷積層進行融合，減少了冗餘的計算量。同時，這種分支結構能夠在提取特徵的過程中保留更多細節，提升了模型的特徵表達能力。

- **深層卷積架構：** CSPDarknet53 包含多個 ResNet 塊，這些塊由殘差連接 (Residual Connections) 組成，使得模型在增加深度的同時仍然可以有效地訓練，減少梯度消失的問題。
- **特徵圖金字塔：** 通過不同層次的特徵圖來捕捉不同尺度的物體特徵，小物體通常在淺層特徵中，而大物體在深層特徵中。

(2) Neck (PANet)：

Path Aggregation Network (PANet) 是 YOLOv8 中的 Neck 模組，用於多尺度特徵融合。可以使高階語意特徵傳遞到較低層級，也可以將低階的細節資訊傳遞至高層。此外，也有利用自適應特徵池化(Adaptive Feature Pooling)將不同尺度的特徵進行融合，使得多尺度物體在特徵空間上可以有一致的表現，這對於多尺度物體偵測非常重要。

(3) Head (Anchor-free Detection Head)：

YOLOv8 的 Head 層引入了 **Anchor-free** 設計，大大簡化了物體偵測的流程，傳統的 YOLO 模型使用 anchor boxes 作為預設框架，但這些框需要手動設置，並且難以適應每個不同數據集。Anchor-free 設計改為直接預測框的中心點和寬高，而無需依賴先驗框，這不僅提高了模型效率，也減少了 anchor boxes 的調參工作。

(4) Detect Layer：

Detect Layer 是 YOLOv8 最終輸出結果的層，負責處理物體位置和類別分數，並利用 NMS 完成後處理過程。

1.6. HSV Pre-process：

我們會需要用到 HSV 對 dataset 進行前處理是因為想要精進模型在特定類別的判斷準確度，並且避免類似「將綠色草叢判斷為綠燈」的錯誤現象。HSV 是一種顏色模型，它的全名是「色相 (Hue)、飽和度 (Saturation) 和明度 (Value)」，經常用於數位影像處理、計算機視覺和圖形設計中。這個模型比 RGB (紅綠藍) 更接近人類的視覺感受，因此在一些應用中比較直觀。

- 色相(Hue)： 表示顏色的種類，例如紅色、綠色、藍色等，用角度來表示，範圍為 0°到 360°。例如，0°表示紅色，120°表示綠色，240°表示藍色。
- 飽和度(Saturation)：表示顏色的純度或鮮豔程度，範圍為 0 到 1(或 0%到 100%)。飽和度越高，顏色越鮮豔，飽和度為 0 時，顏色呈現灰階。
- 明度(Value)： 表示顏色的亮度，範圍為 0 到 1(或 0%到 100%)。明度越高，顏色越亮，為 1 時最亮，為 0 時顏色完全黑暗。

HSV 模型通常用於選擇或調整顏色，特別是在需要調整亮度和飽和度的情況下，例如影像處理中的物件偵測和分割。

2. 研究指標

在這次的研究中，如何判斷一個訓練好的模型是好是壞，量化之後的數據指標就扮演了非常重要的角色，在這個章節中，我們將介紹兩個主要的研究指標以及其所延伸的次要指標，並在第三個章節利用這些指標去判斷模型的優劣。

2.1. Normalized Confusion Matrix (歸一化混淆矩陣)：

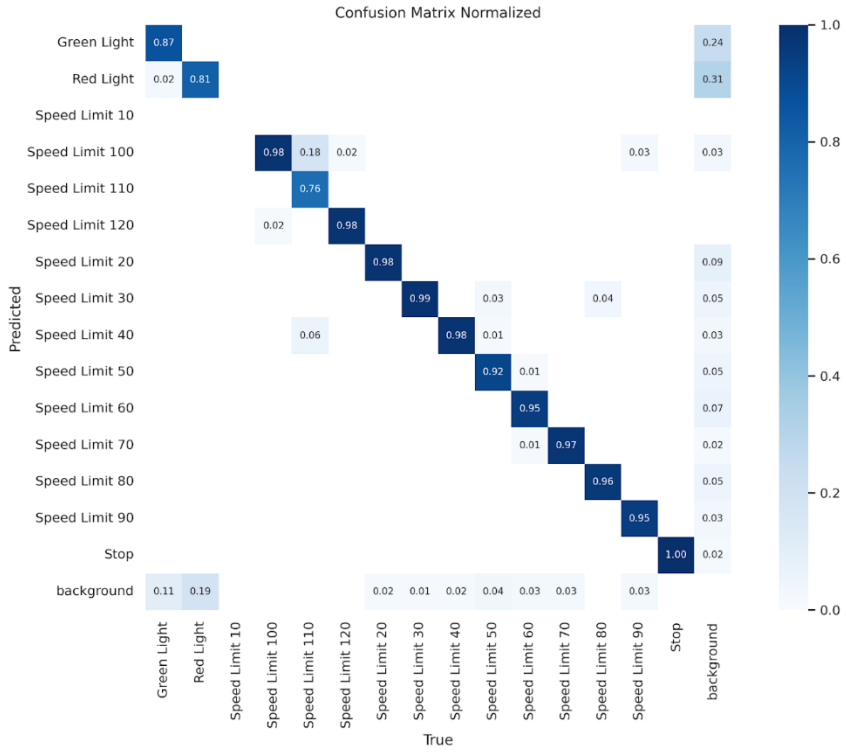

Fig.1: Normalized Confusion Matrix

當一個 batch 的資料經過 model 分類完後，可以依照資料真實的類別和模型預測的類別，將資料分成四類，True Negative、True Positive、False Negative、False Positive。

(1) True Negative (TN)： 在預測某一類別時，遇到不屬於該類別的資料，成功判斷資料不屬於此類別。以上圖為例的 green light 類別為例，不屬於 True green light 這行或 Predicted green light 這列的資料，都屬於 green light 的 True Negative，因為 model 成功預測這些資料真實類別不屬於 green light 的資料，不屬於 green light 這個類別。

(2) True Positive (TP)： 在預測某一類別時，遇到屬於該類別的資料，成功判斷資料屬於此類別。以上圖為例的 green light 類別為例，在 True green light 的這行，有 106 筆資料，被 model 預測是屬於 green light 類別，表示這些資料的真實類別和預測類別是一致的。

(3) False Negative (FN)： 在預測某一資料時，卻預測資料不屬於其真實類別。以上圖的 green light 類別為例，在 True green light 的這行，有 3 筆資料和 13 筆資料，被分別分類為 Red light 和 background，但這 16 筆資料的真實類別是屬於 green light。

(4) False Positive (FP)： 在預測某一資料時，卻預測資料屬於其不是真實類別的類別。以上圖為例，在 predicted green light 這列，有 14 筆的資料，被預測為 green light，但這些資料的真實類別是 background。

以下為運用上述四種參數計算所產生的指標值：

Precision (精確度)： 預測為正的樣本中，有多少是正確的

$$Precision = TP/(TP + NP)$$

Recall (召回率)： 真實類別為正的樣本中，有多少被預測為正

$$Recall\ =\ TP\ /\ (TP\ +\ FN)$$

F1-score： 綜合考慮精確度和召回率的指標

$$F1\ =\ 2\ *\ (Precision * Recall\ )\ /\ (Precision\ +\ Recall)$$

Accuracy (準確度)： 所有樣本中，模型正確預測的比例

$$Accuracy\ =\ (TP\ +\ TN)\ /\ (TP\ +\ TN\ +\ FP\ +\ FN)$$

因此，在 Fig.4 的 Normalized confusion matrix 中我們可以看出一個模型的 Precision，也就是精確度，且這項精確度已經被量化過，因此這在後面研究結果的章節扮演相當重要的角色。

2.2. mAP：

mAP（Mean Average Precision）是一種常用的評估指標，主要用於物體偵測任務中，以衡量模型在辨識和定位多個物體時的準確性，其綜合了**精確率（Precision）和召回率（Recall）**，可用於評估模型在各個分類上的整體表現。

而在下個章節我們會使用到"mAP50"和"mAP50-95"來做為評估模型好壞的指標，舉例來說，mAP50 中的 50 指的是 IOU 的 threshold，設定為 0.5，也就是 50%的重疊率。當 IoU >= 0.5 時，即認為預測框的定位是正確的。在 mAP50 中，模型需要在所有類別的 IOU threshold 為 0.5 時計算平均精確率（AP），並最終得到整體的平均值 mAP。

而 mAP50-95 的評估要求模型在多種不同的 IOU threshold 下都有良好的表現，因此相比 mAP50，它對模型的檢測精度要求更高，是更具挑戰性的評估標準。

3. 研究結果

在多次的訓練中，我們採用了許多不同的訓練方法以及參數，因此，這個章節我們會比較這些模型的指標數據，如 normalized confusion matrix 以及 mAP；此外，在某些 case 中，我們會拿一個以行車紀錄器錄製的影像，以一幀一幀輸入的方式，讓訓練好的模型來判別，並且比較不同訓練方式的模型，在同一影像中同一物體的判斷精確度(Precision)與召回率(Recall)。而我們比較的訓練有以下三種: (1)不同 epoch 數 (2)不同 input size (3)HSV 與 RGB 資料前處理方式。

3.1. 不同 epoch 的影響(皆為 RGB 顏色模型，input size=640*640)：

由 epoch=30 和 epoch=500 訓練出的模型，可以從他們的 Normalized confusion matrix 看出，在 epoch=500 中的斜對角線數值在每個類別都大過於 epoch=30 中的數值，此外，我們發現，當 epoch 大約訓練到 410 時，這時的 mAP 值以及 loss 就沒有再改變了，因此，我們可以得知愈大的 epoch 對準確度(Precision)來說是正相關的。

此外，我們也可以比對當 epoch 訓練到 30 以及 epoch 訓練到大約 410 時的 mAP 指標:

	mAP50	***mAP50-95***
Epoch=30	0.9598	0.83092
Epoch=410	0.9626	0.83918

Table.1: mAP 值的比較(不同 epoch)

由 Table.1 可以得知同樣的結果，也就是愈大的 epoch 會有更好的模型表現，因此，我們在接下來的模型改動中，皆使用約 300~410 的 epoch 來做訓練。

3.2. 不同 input size 的影響(皆為 RGB 顏色模型，epoch = 300)：

從訓練結果的 Normalized confusion matrix 可以發現，在將 input size 調整到 1024*1024 之後，整個模型在速限判別的精確度有了很大的進步，我們推測是因為速限路牌在影像的判定中需要提取到比較細節的像素，若像素太低的話，很容易將路牌上的數字判定為其他數字；因此，我們將一部以行車紀錄器錄製的影像分別輸入進這兩個訓練好的模型中，我們發現，在影片中的這個片段，即 Fig.2 與 Fig.3 之畫面，兩個模型在預測畫面中時速 50 的速限路牌時給出了不同的答案，而這也驗證了我們前面的推測，較大的 input size 會顯著提升速限路牌辨識的精確度。然而，愈大的 input size 隨之而來的代價就是超參數的暴增以及拉長

訓練時間，因此我們最後並未往更高的 input size 方向繼續前進，我們認為 1024*1024 對於交通號誌及路牌辨識已經相當足夠了。

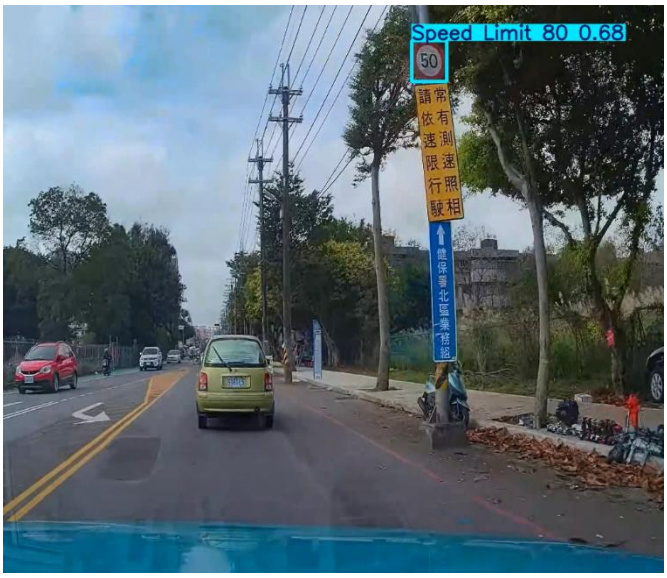

Fig.2: 行車紀錄器影像其中一畫面的辨識結果 (input size=640*640)

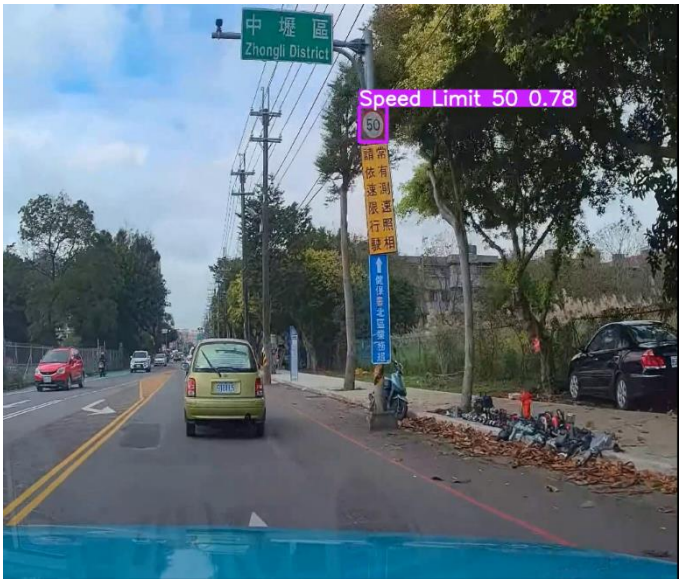

Fig.3: 行車紀錄器影像其中一畫面的辨識結果 (input size=1024*1024)

此外，當我們看到 mAP 指標時，由 Table.2 也可以看出，使用較高的 input size 可以得到整體表現較好的模型。

	mAP50	***mAP50-95***
640*640	0.96139	0.83265
1024*1024	0.97631	0.8565

Table.2: mAP 值的比較(不同 input size)

3.3. RGB 與 HSV 顏色模型的影響(其餘參數皆相同，epoch = 300，input size=1024*1024)：

在最一開始我們會有調整成 HSV 顏色模型的想法，是因為想要精進模型在綠燈類別的判斷準確度，以及避免模型將綠色草叢判斷為綠燈。然而，我們在這兩種訓練方式的 normalized confusion matrix 發現，在使用 HSV 顏色模型後，綠燈的精確度並沒有因此提高，反而有降低的現象。

此外，Fig.4 和 Fig.5 分別是 RGB 與 HSV 顏色模型對行車紀錄器影像的判斷結果，我們挑了一個可以同時錄到兩個綠燈的畫面來看，可以看出，使用 RGB 的 Fig.4 在這個畫面把所有的綠燈都正確地判斷出來，反之，使用 HSV 的 Fig.5 則只有判斷出最近的這個綠燈，且信心度也非常低，因此，以我們的訓練結果來說，HSV 沒辦法給綠燈這個類別提升判斷的精確率(Precision)以及召回率(Recall)。

Fig.4: 行車紀錄器影像其中一畫面的辨識結果 (use RGB)

Fig.5: 行車紀錄器影像其中一畫面的辨識結果 (use HSV)

而上述的現象也可以用 mAP 指標來驗證，使用 HSV 的模型確實在 mAP50 以及 mAP50-95 的表現上不如使用 RGB 的模型。

	mAP50	***mAP50-95***
RGB	0.97631	0.8565
HSV	0.96452	0.84111

Table.3: mAP 值的比較(不同顏色模型)

4. 結論

在第三部分研究結果的比較當中，我們發現了下列三種使模型表現較佳的調整：

- **調高模型的訓練 epoch：** 提高 epoch 是一種非常直接能提升模型表現的方法，但是當 epoch 到了一定的值之後，模型表現的提升便會趨緩，因此，也沒辦法無止盡地提高，且愈高的 epoch 就需要更長的訓練時間。

- **使用較大的 input size 訓練模型：** 這能夠讓模型在判斷一些精細度要求較高的物體時，有較好的表現，例如速限路牌上的數字。然而，愈大的 input size 所需要的超參數以及訓練時長都會暴增，因此我們最後認為 1024*1024 是一個符合我們需求，並且能達到好的辨識效果的 input size。

- **使用 RGB 顏色模型而非 HSV：** 在最一開始我們是想透過 HSV 的顏色模型來提升舊有模型在綠燈類別上的判斷表現，然而，我們從最後的結果發現，其對綠燈的判斷表現並未提高，反而降低了；當然，這不排除我們可能是在 RGB 與 HSV 的色調數值轉換上出現了一些問題，但我們這次的實驗結果確實是得出了 HSV 不如 RGB 的結論，或許在未來我們可以更加深入地調整並且研究這個部分。

因此，我們最後選擇了使用 300 個 epoch、1024*1024 的 input size、以及 RGB 的顏色模型來訓練最終的 YOLOv8，並達到最高的預測表現。

總結來說，我們認為 YOLOv8 的模型架構非常適合運用在交通號誌的辨識上，因為他不僅能在低計算成本下實現快速的推理，且在不同的 IOU threshold 值都有良好的表現(如上述提到的 mAP50-95)，這非常適合嵌入在自動駕駛系統中，即時地辨識交通號誌，幫助駕駛者或系統迅速做出反應。此外，我們在最後訓練方式的比較中，也找到了使模型預測表現更好的趨勢，也期許在未來能夠以這樣訓練好的模型，連接其他周邊電路實際操作在 FPGA 上。

5. 心得：

這一年對深度神經網路（DNN）的學習，不僅讓我們理解了其多層次特徵學習的強大，也通過 YOLO 架構在即時物體辨識中的應用，深化了我們對 DNN 在具體任務中靈活性的認識。YOLO 作為單階段檢測模型，其特點在於快速、高效地處理影像辨識，這讓我們進一步理解了 DNN 如何在不同應用需求下進行結構上的優化，以實現平衡準確度與速度的目標。

在 YOLO 的實驗過程中，我們進一步了解了 DNN 模型中的參數調整對整體表現的影響。YOLO 不僅需要合理的 epoch 來避免過度或不足訓練，還依賴於 input size 的選擇。例如，我們發現增大影像尺寸可以顯著提升模型對小型或高細節物體（如速

限標誌）的辨識精度，但這也同時增加了計算負擔。在 YOLO 模型訓練中，如何適應不同場景需求並調整參數，成為一項重要的挑戰。這讓我們對 DNN 在實際應用中的平衡性有了更深的體會，理解了如何根據場景調整模型以適應運行環境的計算資源。

總結來說，這一年透過學習 DNN 和 YOLO 架構，我們不僅深化了對深層神經網路運作機制的理解，也在實際應用中掌握了模型訓練、參數優化的技巧，認識到模型在即時性需求下的優勢與挑戰，特別是如何在未來應用中進一步提升模型的靈活性和精確度，使其在更多的智能系統中實現真實價值。

6. 參考資料

[1] https://medium.com/@juanpedro.bc22/detailed-explanation-of-yolov8-architecture-part-1-6da9296b954e

[2] https://medium.com/lifes-a-struggle/mean-average-precision-map-%E8%A9%95%E4%BC%B0%E7%89%A9%E9%AB%94%E5%81%B5%E6%B8%AC%E6%A8%A1%E5%9E%8B%E5%A5%BD%E5%A3%9E%E7%9A%84%E6%8C%87%E6%A8%99-70a2d2872eb0

[3] https://www.kaggle.com/code/pkdarabi/traffic-signs-detection-using-yolov8

[4] https://rumn.medium.com/yolo-data-augmentation-explained-turbocharge-your-object-detection-model-94c33278303a

[5] https://medium.com/ai%E5%8F%8D%E6%96%97%E5%9F%8E/learning-model-%E4%BB%80%E9%BA%BC%E6%98%AFap-map-iou-%E8%BD%89%E9%8C%84-dd586fe93189

Dataset: https://www.kaggle.com/datasets/pkdarabi/cardetection

7. 團隊分工方式

	許祐銨	王柏竣	謝秉均
專題報告	50%	30%	20%
專題摘要	50%	30%	20%
專題海報	10%	30%	60%
訓練與研究模型	30%	30%	40%