國立臺灣海洋大學

通訊與導航工程學系

碩士學位論文

指導教授：張麗娜博士

YOLO 深度學習網路應用於漁船編號之辨識

YOLO_based Deep Learning in Fishing Vessel Number Recognition

研究生：林修賢

中華民國 110 年 7 月

YOLO 深度學習網路應用於漁船編號之辨識

YOLO_based Deep Learning in Fishing Vessel Number Recognition

研 究 生：林修賢　　Student : Siou-Hsien Lin

指導教授：張麗娜　　Advisor : Lena Chang

國 立 臺 灣 海 洋 大 學
通 訊 與 導 航 工 程 學 系
碩 士 論 文

A Thesis
Submitted to Department of Communications, Navigation and Control Engineering
College of Electrical Engineering and Computer Science
National Taiwan Ocean University
in partial fulfillment of the requirements
for the Degree of
Master of Science
in
Communications, Navigation and Control Engineering
July 2021
Keelung, Taiwan, Republic of China

中華民國 110 年 7 月

摘要

台灣四面環海，漁業更是台灣重要的經濟產業。近幾年來漁業資源永續發展的議題愈來愈受到重視，政府也藉由對漁船的進出港監控，來取得漁業保育與經濟發展的平衡。但以人力的方式進行船舶的監測，成本高且效率不佳。近年來，深度學習的技術蓬勃發展，結合深度學習與影像辨識的技術更是廣泛應用於很多領域。本研究將應用此技術於漁船編號自動辨識，以達智慧化港口管理之目的。因 YOLO 深度學習網路具快速偵測且準確度高的特性，本研究將應用 YOLO 深度學習網路進行漁船編號辨識，以提升船舶辨識的準確率，不僅可以節省人力，還可達漁船進出港口的自動化管理。

本研究重點有三個部分，一是建立台灣漁船影像數據集，二是找出最佳漁船船號辨識系統的架構，三是比較幾種深度學習網路在船號辨識上的優劣。本研究提出三種船號辨識的系統，分別為直接船號辨識系統、Region_based 船號辨識系統以及改良式 Region_based 船號辨識系統。直接船號辨識系統僅以一個網路直接辨識船號。Region_based 船號辨識系統是以二階段網路來達到船號辨識，第一階段網路先偵測船號區域，第二階段網路再針對此區域進行字元辨識。改良式 Region_based 船號辨識則是改良 Region_based 的系統，針對第一階段找出歪斜船號區域進行角度校正後，再進到第二階段船號辨識。實驗中也針對四種 YOLO 深度學習網路進行比較，包含 YOLOv3, YOLOv3-tiny, YOLOv4, YOLOv4-tiny。實驗結果顯示使用 YOLOv4 網路和改良式 Region_based 船號辨識的系統會得到最好的辨識效果。船號個別字元的辨識正確率有 95.7%，船號完整辨識的正確率也達 88.3%。若要降低硬體的成本或設置在運算能力較低的裝置上，YOLOv4-tiny 對於硬體的要求較低且也有不錯的正確率。

關鍵詞：漁船船號辨識、深度學習、YOLO 網路、YOLOv3、YOLOv3-tiny 、YOLOv4、YOLOv4-tiny

Abstract

Taiwan is surrounded by the sea, and fishing is an important economic industry in Taiwan. In recent years, the issue of sustainable development of fishery resources has received more and more attention, and the government has also used the monitoring of the entry and exit of fishing boats to achieve a balance between fishery conservation and economic development. However, it is costly and inefficient to monitor ships by means of manpower. In recent years, the technology of deep learning has developed vigorously, and the technology of combining deep learning and image recognition has been widely used in many fields. This research will apply this technology to the automatic identification of fishing vessel numbers for the purpose of intelligent port management. Because the YOLO deep learning network has the characteristics of rapid detection and high accuracy, this research will apply the YOLO deep learning network to identify the number of fishing boats to improve the accuracy of ship identification. It not only saves manpower, but also allows fishing boats to enter and exit the port. Automated management.

This research focuses on three parts. The first is to establish a Taiwanese fishing boat image data set, the second is to find the best fishing vessel number identification system architecture, and the third is to compare the advantages and disadvantages of several deep learning networks in ship number identification. This research proposes three ship number identification systems, namely the direct ship number identification system, the Region_based ship number identification system and the improved Region_based ship number identification system. The direct ship number identification system only uses a network to directly identify ship numbers. The Region_based ship number identification system uses a two-stage network to achieve ship number identification. The first-stage network first detects the ship number area, and the second-stage network performs character recognition for this area. Improved Region_based ship number identification is an improved Region_based system. After finding the skewed ship number area in the first stage and correcting the angle, it then proceeds to the second stage of ship number identification. The experiment also compares four YOLO deep learning networks, including YOLOv3, YOLOv3-tiny, YOLOv4, YOLOv4-tiny. The experimental results show that the system using the YOLOv4 network and the improved Region_based ship number identification system will get the best identification effect. The recognition rate of individual characters of ship number is 95.7%, and the correct rate of complete recognition of ship number is 88.3%. If you want to reduce the cost of hardware or install it on a device with lower

computing power, YOLOv4-tiny has lower requirements for hardware and has a good accuracy rate.
Keywords: Fishing Vessel Number Recognition, Deep Learning, YOLO, YOLOv3, YOLOv3-tiny, YOLOv4, YOLOv4-tiny

目錄

圖目錄

表目錄

第一章　研究背景

1.1　研究動機

台灣附近有許許多多的洋流交匯，因此漁業非常發達。近年來因為環保意識逐漸抬頭，環保團體對於漁業資源是否捕撈過度愈來愈重視。為了減少漁業濫捕的行為，在港邊對漁船的監控就顯得相當重要。但若以人力的方式，在漁船進出港時對漁船統一編號進行辨識，太過於浪費人力與時間。且台灣的漁船統一編號並沒有統一使用同一種字體，有些漁船編號甚至是漁民自己寫在船上，在光線不佳與距離超過肉眼可辨識距離時，以人力辨識容易造成誤判。因此需要一個智慧化的系統來精簡人力，並對進出港的漁船進行監測。

傳統往以影像處理與電腦視覺的方法建立漁船船號辯識的系統，必須先透過人工經驗法則進行目標物相關參數的設定，再進行特徵萃取。但是漁船船號在漁船影像中佔的比例太小，不容易辨識，漁船編號也有許多不同的大小、顏色和字體，很難找到涵蓋所有漁船編號的特徵進行萃取。

近年來由於圖形處理器(Graphics Processing Unit, GPU)的效能大幅提升，動態隨機存取記憶體(Dynamic Random Access Memory, DRAM)的單位容量成本大幅下降，需要大量記憶體空間及強大的平行運算效能的深度學習網路開始被重視。目前深度學習網路被應用在許多地方，例如人臉辨識、社交距離的偵測、檢查工廠產品的瑕疵偵測.....等，可見深度學習網路的應用範圍非常廣泛。利用深度學習網路進行物件偵測有很高的準確度而且運算速度相當快，因此本論文會研究運用這種方法，提出深度學習應用於漁船船號辨識的研究。

1.2　研究目的

本研究主要為應用現有基於深度學習概念的物件偵測網路架構對漁船影像中的船號進行分類與辨識，其重點有三個部分，一是建立台灣漁船影像數據集，二是找出最佳漁船船號辨識方法的架構，三是比較幾種深度學習網路在船號辨識上的優劣。

對深度學習網路來說，有數量多且豐富的訓練資料是很重要的。然而在網路上的漁船訓練集大部分都是外國漁船，外型與台灣漁船有所差別。且大部分漁船照片不是為了拍到漁船編號而拍攝，幾乎都沒拍到漁船統一編號，或者漁船編號太過模糊導致無法辨識。因此本論文使用港邊所拍攝漁船影像，以確保漁船編號清楚且可以辨識，並用來建立本論文所需之漁船訓練集。

在漁船影像中，漁船編號所占得比例太小，以深度學習網路進行物件偵測，直接辨識出漁船編號的成效不佳。因此本研究將原本一步到位直接辨識船號的方法取代為二階段，先找出漁船編號所在區域，再從漁船編號區域中進行辨識，解

決偵測物件過小的問題。在本研究中也嘗試旋轉歪斜船號圖片，減少因為船號號歪斜，導致無法辨識的情形，以提升準確率。

本研究在實驗中使用了幾種較為常見的YOLO深度學習網路模型進行訓練，並使用本論文提到辨識方法的架構來比較，找出船號辨識最合適的YOLO深度學習網路模型與辨識方法的架構。

1.3 論文架構

本論文架構分為五個章節，其內容分述如下:

第一章　研究背景:

說明研究動機、研究目的及論文架構

第二章　相關文獻回顧:

簡單介紹台灣漁船編號，回顧相關船號辨識方法與介紹論文中使用之深度學習網路

第三章　研究方法:

說明如何建置及處理資料集，運用在深度學習網路訓練，並說明三種辨識系統的架構及評估效能的方式。

第四章　實驗結果:

比較四種深度學習網路在使用三種辨識方法的準確度、運算速度與運算量，以選擇最佳配置，並對實驗進行討論。

第五章　結論與建議:

統整實驗結果與後續可改善方向。

第二章　相關文獻回顧

2.1　台灣漁船統一編號

台灣漁船會在建造時依照漁船噸級給予台灣漁船統一編號，頭二字會以 CT 開頭。第三個字是數字時所代表漁船的是噸位，分別是 CT0 (5 噸以下)、 CT1(5 到 10 噸)、 CT2(10 到 20 噸)、 CT3(20 到 50 噸)、 CT4(50 到 100 噸)、CT5(100 到 200 噸)、 CT6(200 到 500 噸)、 CT7(500 到 1000 噸)、 CT8(1000 噸以上)。此外還有不具備船外型的漁筏(以 CTR 開頭)和沒有封閉式甲板的舢舨(無動力以 CTX 開頭，有動力以 CTS 開頭)也會在船身上標記漁船統一編號。台灣漁船的漁船統一編號依照規定是標示在船頭兩側或船尾，中文船名和英文船名的下方，編號的大小因漁船噸位而定，不過有些漁船也會在船身標示漁船統一編號，顏色要與漁船背景顏色有所差異，淺色底深色字，深色的底則用淺色字。在字體方面則沒有規定，許多漁船噸位小的漁民在漁船編號掉漆後會自己手寫，再加上船頭兩側是曲面的，文字會因此而變形，因此在船號辨識上會有難度。

本研究因為漁筏和舢舨可在網路上找到的照片太少，在基隆附近漁港也找不到，但漁筏和舢舨的外型和漁船統一編號與一般漁船有所差異，若要以深度學習網路進行訓練，太少訓練資料，因此本研究的漁船資料不包含漁筏和舢舨。

2.2　相關船號辨識影像處理方法

以傳統影像處理方法辨識漁船船號會是兩階段的辨識，先找出船頭(漁船編號所在區域)，再將船頭進行字型辨識。Ferreira J.C.等學者在第八屆環境智能國際研討會提出的 Computer Vision Algorithms Fishing Vessel Monitoring — Identification of Vessel Plate Number [1] 文章中，分為四步驟，一開始將船頭朝左的照片做水平翻轉，將所有照片中的船頭都朝同一方向，再找出照片中船頭位置，之後對影像進行預處理，最後進行辨識。在實驗中先使用 Matlab 對船頭朝左的照片水平翻轉的。尋找船頭特徵的方法是採用 N. Dalal 等學者所提出的方向梯度直方圖 (Histogram of Oriented Gradients,HOG [2])所架構而成的級聯分類器 (cascade classifier)。級聯分類器是以大量的分類器組合而成，透過輸入正面影像(包含訓練目標)和負面影像(不含訓練目標)進行訓練，就可以得到所需的分類器。文章中提到以 200 張負面影像和 125 張正面影像進行訓練會有最好的結果，但精準率大約只有 50%。精準度較低主要的原因在於漁船距離，漁船距離太近或太遠都無法辨識。再來使用的是 Matlab 內建的光學字元辨識(Optical Character Recognition,OCR)找出辨識位置。最後使用最大穩定極值區域(Maximally Stable Extremal Region,MSER)的技巧將轉成紅色(尋找較暗顏色略好於黑白影像)後抓

出強度在 90(強度範圍 0 到 255)、畫素(pixels)區域大小在 30 以下的深色船號，而淺色船號可由相反步驟得到。最後將處理完的影像送到 Matlab 的光學字元辨識找出船號。

此影像處理方法為了找出船頭進行水平翻轉統一船頭方向，但對於被水平翻轉的船號會有辨識上的問題。以方向梯度直方圖級聯分類器在找出船頭區域的精準度也不佳。而且部份台灣漁船的漁船編號不只標示在船頭，有些漁船會在船身標示。如果漁船影像是從側面拍攝，船身的漁船編號會比船頭的編號來的清楚。在辨識船號方面，因為有船體的彎曲和顏色差，在辨識上也沒這麼容易。因此本論文對這兩部分使用 YOLO 深度學習網路進行替代。

2.3 YOLO

Jeseph Redmon 等學者提出的 YOLO [3,4,5] 的全名為 You Only Look Once，意思是說 YOLO 模型的特性只需要對圖片做一次 CNN 運算流程 Convolutional neural network(CNN),便能夠判斷裡面的物體類別及位置，大大提升辨識速度。YOLO 是單一網路設計，判斷結果會包含 Bounding Box(預測框)位置以及每個 Bounding Box 所屬類別和機率，整個網路是屬於 end-to-end(輸入原始資料，輸出結果)的，此種方法是屬於較易訓練且速度較快的。目前 YOLO 原作者 Jeseph Redmon 開發了三個版本，第一個版本 YOLOv1 在提升速度下犧牲了一些精度，因此第二個版本 YOLOv2 [6] 引入了其他深度學習方法進行改進，第三個版本 YOLOv3 [7] 則進一步將網路加深，提升準確度。在 2020 年 YOLO 技術的維修人員 Alexey Bochkovskiy 與中研院的廖弘源所長與王建堯博士共同研發了最新的 YOLO 第四版 YOLOv4 [8]，運算速度與 YOLOv3 差不多，精準度卻更高。YOLOv4 的論文與程式碼連結有被放入 YOLO 官方 github,可證明 YOLOv4 有被原作者所認可。在本論文中會簡單介紹 YOLOv3、YOLOv4 以及這兩種版本的簡化版，H. Gong 等學者提出的 YOLOv3-tiny [9] 與 Alexey Bochkovskiy 等學者所提出的 YOLOv4-tiny [10]。

2.3.1 YOLOv3

YOLOv3 主要分為三個部分，第一部分是採用一個新的特徵萃取網路進行特徵萃取，該網路是由 YOLOv2 的 Darknet-19 進行改良，並進一步加深網路，因為有 53 層卷積層，因此被命名為 Darknet-53 [7]，其架構如圖 1 所示。除了以 3×3 的卷積核進行特徵萃取及 1×1 的卷積核進行降維外，與 YOLOv2 相比雖然保留了 route 層的動作但刪去了拆分重組的動作，改以使用 shortcut 層。與 route 層不同的是，shortcut 層的功能為當前經卷積得到的特徵圖與指定較淺的特徵圖進行逐元素(element-wise)相加，route 層則是維度上的相加，並不會影響原來裡面的元素。shortcut 層的使用目的在於避免網路過深造成特徵消失而發生梯度消失的問題。

接下來為多尺度特徵圖的預測，YOLOv3 參考了 T.Y. Lin 等學者所提出的 Feature Pyramid Network(FPN [11])萃取大中小三個不同尺度的特徵，如圖 2 所示，不同於 YOLOv2 直接在每個網格的特徵圖上使用 5 種不同形狀、大小的 Anchor Box [5]，YOLOv3 在三個不同大小的特徵圖上分別使用 3 個不同形狀、大小的 Anchor Box，總共 9 個 Anchor Box，並再使用 K-means 聚類方法計算 Anchor Box 的尺寸。YOLOv3 的預設 Anchor Box 尺寸是利用 PASCAL VOC 數據集[12] 而來，其尺寸為(10×13), (16×30), (33×23), (30×61), (62×45), (59×119), (116×90), (156×98), (373×326)。數據集當中，若輸入影像尺寸為 480×480，則會產生先產生 15×15 的特徵圖，並在經由最鄰近內差法的上採樣得到 30×30 特徵圖，並在與網路前半部的特徵圖結合，完成後再繼續進行一樣的步驟得到 60×60 的特徵圖。在最小的特徵圖 15×15 上，其感受視野最大，因此適合檢測較大的目標因此用最大的 3 個 Anchor Box 尺寸(116×90),(156×98),(373×326)，中間大小的特徵圖 30×30 感受視野也是中等的，因此 Anchor Box 即選擇 3 個中等尺寸的大小(30×61), (62×45), (59×119)，最大的特徵圖，其能感知最多的細節，因此採用最小尺寸的 Anchor Box(10×13),(16×30),(33×23)。

在分類上，YOLOv3 取消採用歸一化指數函數(Softmax)分類器，原因在於類別標籤可能是重疊的，例如狗與動物，而 Softmax 分類器的使用前提為，每個 Box 只能對應到一個分類，這會造成在使用上的限制，並往往與事實不相符，因此 YOLOv3 改採用邏輯斯(Logistic)分類器，使一個 box 可以對應到多個分類，讓 YOLOv3 可以更好的應用在更複雜的應用場景。

YOLOv3 的損失函式大概可分為 Bounding Box 的偏移量預測和信心度預測，與 YOLOv2 不同的是，由於 YOLOv3 將類別分類器由 Softmax 改為 Logistic，因此在損失函式的類別預測也從平方誤差總和改為二值交叉熵(Binary Cross-Entropy,BCE)，做二元分類預測，二元的意思為有這個類別或沒有這個類別，而損失函式裡的總和即是代表針對所有可能出現的類別各做一次預測。

	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			

圖 1　Darknet-53 [7]

圖 2　多尺度偵測示意圖(小到大尺度) [21]

2.3.2 YOLOv3-tiny

YOLOv3 有很高的精準度，但對於部分情況來說，不需要這麼高的精準度，會比較著重在運算速度上，因此 YOLOv3-tiny 也因此而誕生了。YOLOv3-tiny 是由 YOLOv3 簡化而成，減少了大量的卷積層，尺度也從三個尺度減少成兩個尺度，只留下中間與最小尺度，YOLOv3 的架構如圖 3 所示，網路層數從 107 層減少為 24 層， Anchor Box 也從 9 個減少為 6 個。YOLOv3-tiny 犧牲了精準度，

但運算速度卻有很大的提升。也因為 YOLOv3-tiny 減少了大部分的網路層，所以 YOLOv3-tiny 對於硬體設備的要求沒有 YOLOv3 這麼高，可以在顯示卡較低階的環境進行訓練。

Layer	Type	Filters	Size/Stride	Input	Output
0	Convolutional	16	3 × 3/1	416 × 416 × 3	416 × 416 × 16
1	Maxpool		2 × 2/2	416 × 416 × 16	208 × 208 × 16
2	Convolutional	32	3 × 3/1	208 × 208 × 16	208 × 208 × 32
3	Maxpool		2 × 2/2	208 × 208 × 32	104 × 104 × 32
4	Convolutional	64	3 × 3/1	104 × 104 × 32	104 × 104 × 64
5	Maxpool		2 × 2/2	104 × 104 × 64	52 × 52 × 64
6	Convolutional	128	3 × 3/1	52 × 52 × 64	52 × 52 × 128
7	Maxpool		2 × 2/2	52 × 52 × 128	26 × 26 × 128
8	Convolutional	256	3 × 3/1	26 × 26 × 128	26 × 26 × 256
9	Maxpool		2 × 2/2	26 × 26 × 256	13 × 13 × 256
10	Convolutional	512	3 × 3/1	13 × 13 × 256	13 × 13 × 512
11	Maxpool		2 × 2/1	13 × 13 × 512	13 × 13 × 512
12	Convolutional	1024	3 × 3/1	13 × 13 × 512	13 × 13 × 1024
13	Convolutional	256	1 × 1/1	13 × 13 × 1024	13 × 13 × 256
14	Convolutional	512	3 × 3/1	13 × 13 × 256	13 × 13 × 512
15	Convolutional	255	1 × 1/1	13 × 13 × 512	13 × 13 × 255
16	YOLO				
17	**Route 13**				
18	Convolutional	128	1 × 1/1	13 × 13 × 256	13 × 13 × 128
19	Up-sampling		2 × 2/1	13 × 13 × 128	26 × 26 × 128
20	**Route 19 8**				
21	Convolutional	256	3 × 3/1	13 × 13 × 384	13 × 13 × 256
22	Convolutional	255	1 × 1/1	13 × 13 × 256	13 × 13 × 256
23	YOLO				

圖 3　YOLOv3-tiny 網路層[22]

2.3.3 YOLOv4

YOLOv4 與 YOLOv3 相比，大致可以把改進的部分分為五個部份。

第一部份是將 YOLOv3 的特徵萃取網路更進一步改良，把原本的 Darknet53 結合 Cross Stage Partial Network (CSPNet)的架構變成 CSPDarknet53，網路架構如圖 5 所示。在特徵萃取時，常常因為重複計算相同的特徵而造成計算量過高。CSPNet 將捲積層分成分成兩部分，第一部份不做任何運算，第二部分跟原本網路一樣進行特徵萃取，最後再把兩部分串聯在一起，示意圖如圖 4。透過 CSPNet 可以大幅減少運算所需的時間，準確度與原本相比可以持平或略高一些。CSPDarknet53 運用 CSPNet 的原理，在原本 Darknet53 中的卷積層中像 CSPNet 一樣，保留一部份不做運算，將特徵保留給下一層，在減少運算量的同時保持準確性。且經過實驗證明，CSPDarknet53 可以透過把 Andrew L Maas 等學者提出的 leaky_relu 激活函數 [13] 改成 Diganta Misra.學者提出的 Mish 激活函數 [14] 提高精準度，2 種激活函數差異如圖 6 所示。因此，YOLOv4 最後選擇 CSPDarknet53 作為特徵萃取網路。

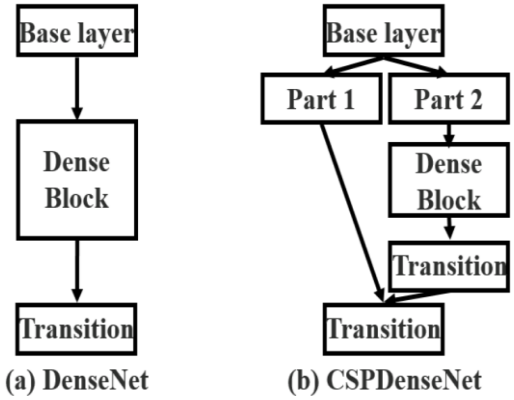


圖 4　無使用 CSPNet 與使用 CSPNet 的 DenseNet [23]

	Type	Filters	Size	Output
	Convolutional	32	3x3	256x256
	Convolutional	64	3x3/2	128x128
	CrossStagePartial			
1x	Convolutional	32	1x1	
	Convolutional	64	3x3	
	Residual			128x128
	Convolutional	128	3x3/2	64x64
	CrossStagePartial			
2x	Convolutional	64	1x1	
	Convolutional	64	3x3	
	Residual			64x64
	Convolutional	256	3x3/2	32x32
	CrossStagePartial			
8x	Convolutional	128	1x1	
	Convolutional	128	3x3	
	Residual			32x32
	Convolutional	512	3x3/2	16x16
	CrossStagePartial			
8x	Convolutional	256	1x1	
	Convolutional	256	3x3	
	Residual			16x16
	Convolutional	1024	3x3/2	8x8
	CrossStagePartial			
4x	Convolutional	512	1x1	
	Convolutional	512	3x3	
	Residual			8x8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Made By Jaredzzz (1143020206@qq.com)

圖 5　CSPDarknet53 [24]

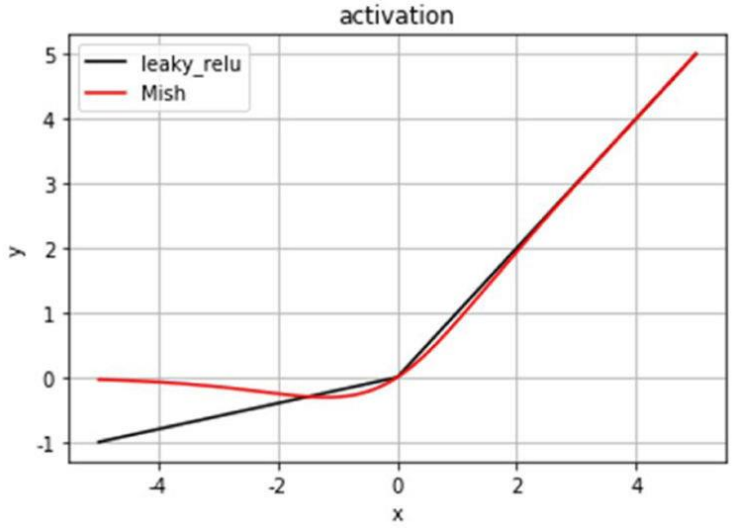


圖 6　Leaky_relu 與 mish 激活函數 [25]

第二部分是將 FPN 變成 K. He 等學者提出的 spatial pyramid pooling layer(SPP [15])和 Shu Liu 等學者提出的 Path Aggregation Network (PANet [16])。SPP 是將最後一層池化層用空間金字塔池化層取代，分別用大小 1×1，5×5，9×9，13×13 的核進行最大池化，再將不同大小的特徵圖串聯後輸出，SPP 示意圖如圖 7。利用 SPP 的方式，可以更有效的增加特徵的接收範圍。在 YOLOv3 裡，三個尺度特徵做法是類似 FPN 網路，下採樣 32 倍，再上採樣兩次得到大、中、小三個尺度。而 YOLOv4 使用的 PANet 是把下採樣與上採樣進行融合，從不同的檢測層進行參數整合，得到更多的特徵，PANet 示意圖如圖 8，FPN 與 PANet 比較如圖 9。

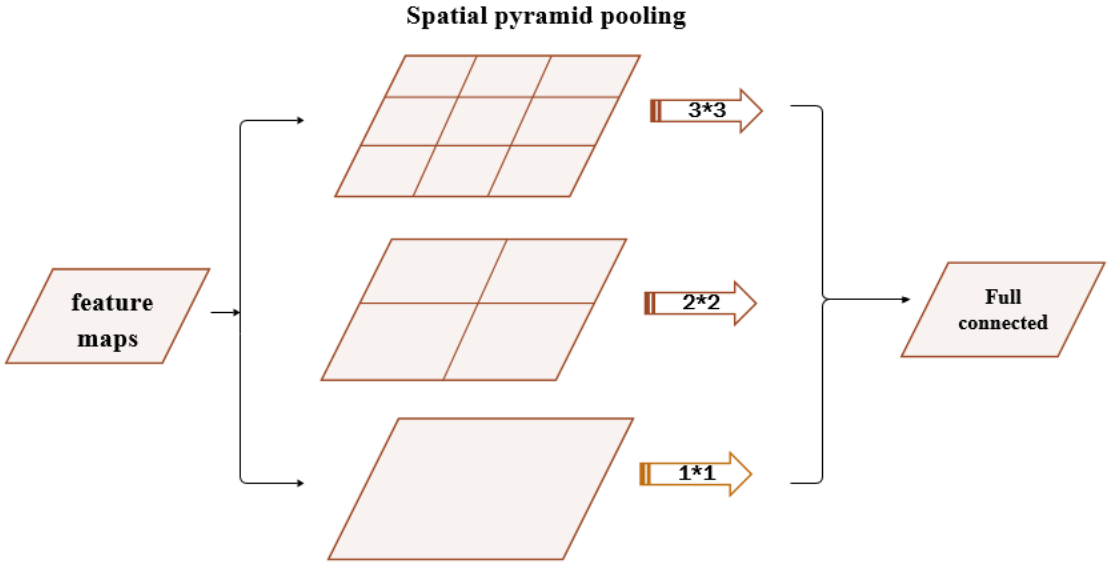


圖 7　SPP 示意圖 [26]

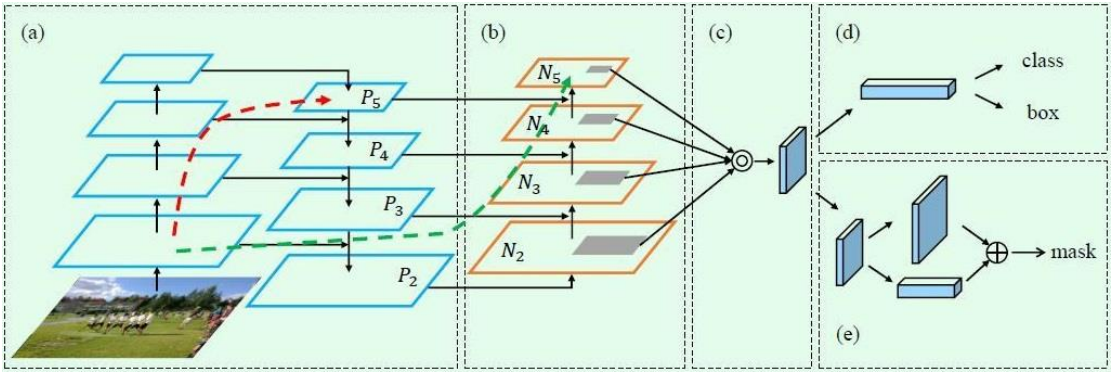


圖 8　PANet 示意圖 [27]

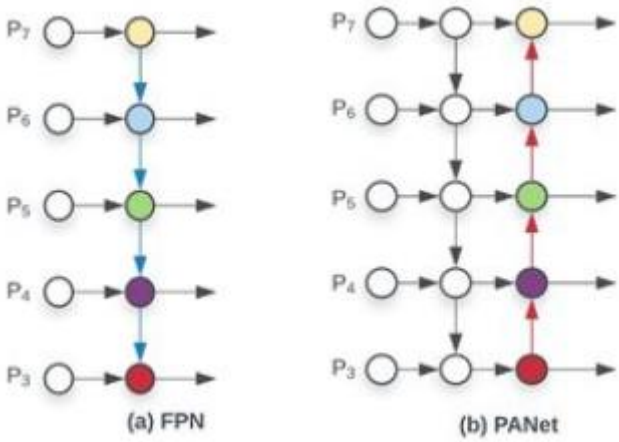


圖 9　FPN 與 PANet 比較圖 [28]

第三個是損失函式的公式。在做出錯誤預測時，損失函式的優劣會影響到訓練結果，能否將錯誤的訓練結果導回正確的方向是關鍵。YOLOv4 的損失函式使用的是 Zhaohui Zheng 等學者提出的 Complete IoU-loss(CIoU-loss [17])，在計算過程中會增加 ground truth(真實框)和預測框的重疊區域、最小化兩者之間的距離並保持框寬高比的一致性。CIoU 損失函式的公式如下:

$$L_{CIoU} = 1 - IoU + \rho^2(b, b^{gt})/c^2 + \alpha \cdot v \qquad (1)$$

IoU 代表預測框與真實框交集除以聯集

b, b^{gt}分別代表預測框與真實框的中心點

ρ代表計算兩個中心點的歐式距離(如圖 10 中的 d)

c 代表預測框與真實框最小聯集區域的對角線距離(如圖 10 中的 c)

α是權重函數，v是長寬比的懲罰項，公式如下

$$v = \frac{4}{\pi^2}\left(\arctan\frac{w^{gt}}{h^{gt}} - \arctan\frac{w}{h}\right)^2 \qquad (2)$$

$$\alpha = \frac{v}{(1 - \text{IoU}) + v'} \tag{3}$$

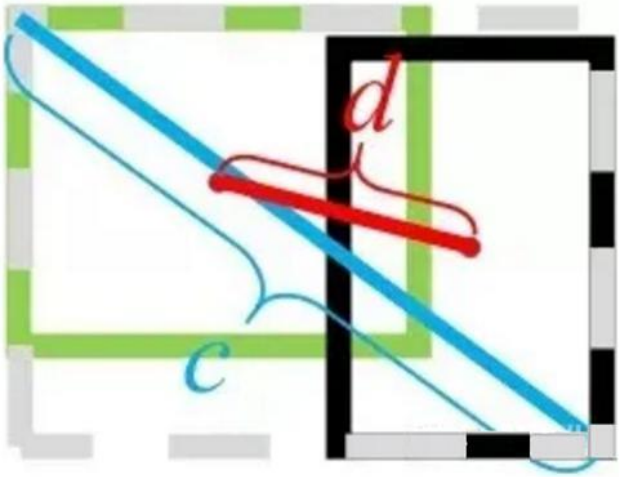


圖 10　兩種框的中心點距離(d)與聯集區域對角線距離(c) [29]

第四部分是擴增影像方面使用 Mosaic 數據增強。YOLOv4 中使用的 Mosaic 是參考 2019 提出的 CutMix 數據增強的方式，但 CutMix 只使用兩張圖片拼接再一起，而 Mosaic 數據增強則使用四張影像隨機的縮放、裁剪、色域等變化並拼接成一張影像來進行數據增強(如圖 11 所示)。使用 Mosaic 可以讓訓練數據集更豐富，也因為一次計算四張影像的數據，在訓練時可以減少單次訓練輸入的圖片量，一個 GPU 就可以得到比較好的效果。

圖 11　Mosaic 數據增強 [30]

第五種是把 Nitish Srivastava 等學者所提出解決過擬合問題的 Dropout[18]換成 Golnaz Ghiasi 等學者所提出的 Dropblock[19]。原本 Dropout 是透過隨機刪除

減少神經元的數量，避免網路過分依賴某些權重。但卷積層就算隨機刪除神經元，還是可以從相鄰的單元學習到相關的信息。Dropblock 也是隨機減少神經元，但與 Dropout 不同的是，Dropblock 是將整個局部區域刪減，避免刪除的部分還是被學習到，Dropout 與 Dropblock 如圖 12 所示。

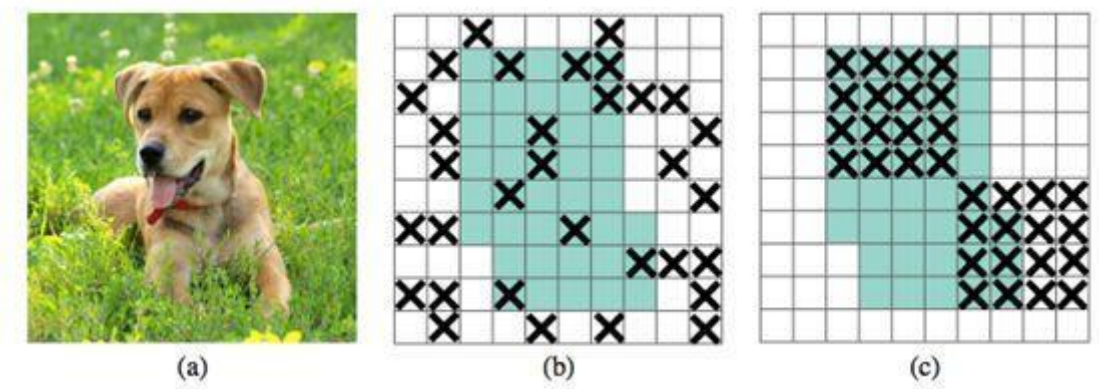


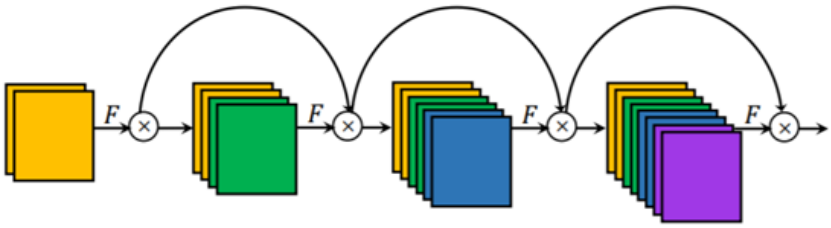
圖 12　(a)輸入影像 (b)Dropout (c)Dropblock [31]

2.3.4　YOLOv4-tiny

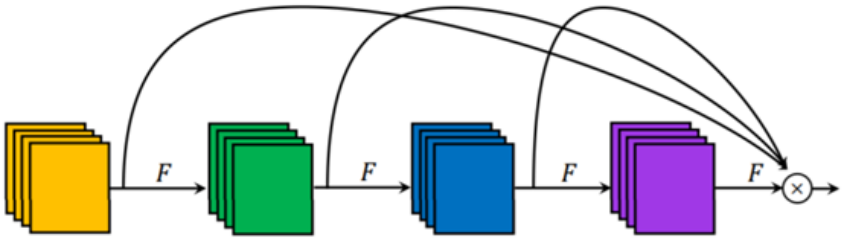
在 YOLOv4 發布不久，YOLOv4 的作者也接著發布 YOLOv4-tiny。網路層從 YOLv4 的 162 層縮減到 38 層(如圖 13 所示)，平均精度比 YOLOv4 低但跟 YOLOv3-tiny 相比卻有巨大的提升。YOLOv4-tiny 與 YOLOv3-tiny 的尺度一樣只有兩個，因此 Anchor Box 也只需要 6 個。但 YOLOv4-tiny 的骨架不是拿 CSPDarknet53 簡化，而是使用 CSPOSANet。CSPOSANet 是由 CSPNet 加上 Youngwan Lee 等學者所提出的 OSANet [20]組合而成，而 OSANet 是來自 VoVNet 的 OSA(One-shot Aggregation)模型，想法是來自 DenseNet 模型，到最後才把所有 layer 串聯在一起，架構如圖 14 所示。經實驗證明，CSPOSANet 比 CSPDarknet53 表現還好，因此使用 CSPOSANet 作為 YOLOv4 tiny 的骨架。

Layer	Type	Filters	Size/Stride	Input	Output
0	Convolutional	32	3 × 3/2	416 × 416 × 3	208 × 208 × 32
1	Convolutional	64	3 × 3/2	208 × 208 × 32	104 × 104 × 64
2	Convolutional	64	3 × 3/1	104 × 104 × 64	104 × 104 × 64
3	Route 2				
4	Convolutional	32	3 × 3/1	104 × 104 × 32	104 × 104 × 32
5	Convolutional	32	3 × 3/1	104 × 104 × 32	104 × 104 × 32
6	Route 5 4				
7	Convolutional	64	1 × 1/1	104 × 104 × 64	104 × 104 × 64
8	Route 2 7				
9	Maxpool		2 × 2/ 2	104 ×104 × 128	52 × 52 × 128
10	Convolutional	128	3 × 3/1	52 × 52 × 128	52 × 52 × 128
11	Route 10				
12	Convolutional	64	3 × 3/1	52 × 52 × 64	52 × 52 × 64
13	Convolutional	64	3 × 3/1	52 × 52 × 64	52 × 52 × 64
14	Route 13 12				
15	Convolutional	128	1 × 1/1	52 × 52 × 128	52 × 52 × 128
16	Route 10 15				
17	Maxpool		2 × 2/ 2	52 × 52×256	26 × 26 × 256
18	Convolutional	256	3 × 3/1	26 × 26 × 256	26 × 26 × 256
19	Route 18				
20	Convolutional	128	3 × 3/1	26 × 26 × 128	26 × 26 × 128
21	Convolutional	128	3 × 3/1	26 × 26 × 128	26 × 26 × 128
22	Route 21 20				
23	Convolutional	256	1 ×1/1	26 × 26 × 256	26 × 26 ×256
24	Route 18 23				
25	Maxpool		2 × 2/2	26 × 26 ×512	13 × 13 × 512
26	Convolutional	512	3 × 3/1	13 × 13 × 512	13 × 13 × 512
27	Convolutional	256	1 × 1/1	13 × 13 × 512	13 × 13 × 256
28	Convolutional	512	3 × 3/1	13 × 13 × 256	13 × 13 × 512
29	Convolutional	21	1 ×1/1	13 × 13 × 512	13 × 13 × 21
30	YOLO				
31	Route 27				
32	Convolutional	128	1 × 1/1	13 × 13 × 256	13 × 13 × 128
33	Upsample		2x	13 × 13 × 128	26 × 26 × 128
34	Route 33 23				
35	Convolutional	256	3 × 3/1	26 × 26 × 384	26 × 26 × 256
36	Convolutional	21	1 × 1/1	26 × 26 × 256	26 × 26 × 21
37	YOLO				

圖 13　YOLOv4-tiny 網路層 [32]



(a) Dense Aggregation (DenseNet)



(b) One-Shot Aggregation (VoVNet)

圖 14　DenseNet 與 OSA 模型 [33]

第三章　研究方法

本研究使用三種船號辨識系統進行比較，分別為直接船號辨識系統、Region_based 船號辨識系統以及改良式 Region_based 船號辨識系統。直接船號辨識系統僅以一個 YOLO 深度學習網路直接辨識船號。Region_based 船號辨識系統是以二階段網路來達到船號辨識，第一階段網路先偵測船號區域，第二階段網路再針對此區域進行船號的字元辨識。改良式 Region_based 船號辨識系統則是改良 Region_based 的系統，針對第一階段船號歪斜區域進行角度校正後，再進到第二階段船號辨識。實驗中每種方法都使用以下四種 YOLO 模型進行訓練: YOLOv3、YOLOv3-tiny、YOLOv4、YOLOv4-tiny，並比較所有情況的實驗結果。

3.1　資料來源

為了訓練本研究所需之深度學習網路模型，需要大量的漁船圖片作為訓練資料，為了辨識漁船編號，圖片裡漁船上的編號也必須清晰可辨識。但是網路上的漁船資料庫都是外國漁船，與台灣漁船相比外型相差太大，漁船編號的組成也不盡相同。因此本研究之漁船訓練集來源是在基隆正濱漁港和八斗子漁港拍攝，拍攝時間是在白天下午，從各種角度拍攝停靠在港邊或進出港的漁船影像，並加上網路上的台灣漁船照片進行補足，共計 705 張。

在訓練資料蒐集時，基隆附近的漁港沒有漁筏和舢舨這種漁船統一編號有特殊英文的漁船(例如 R,S.. . 等)，在網路上也找不到漁船統一編號足夠清晰的照片，對深度學習網路來說無法蒐集到足夠的照片訓練，故沒有加入訓練集中。

3.2　訓練集的建置

在訓練集的建置中，本研究使用 labelImg 對訓練資料進行標記(如圖 15)。labelImg 可以產生 xml 檔紀錄研究資料圖片的大小、標記的標籤(label)及邊界框(bounding box)在圖片中的絕對位置(如圖 16)。但 YOLO 深度學習網路所讀取的是 txt 檔且邊界框必須是相對位置，因此最後還必須用 Python 將 xml 檔轉成 YOLO 深度學習網路所能接受的 txt 檔(如圖 17)。

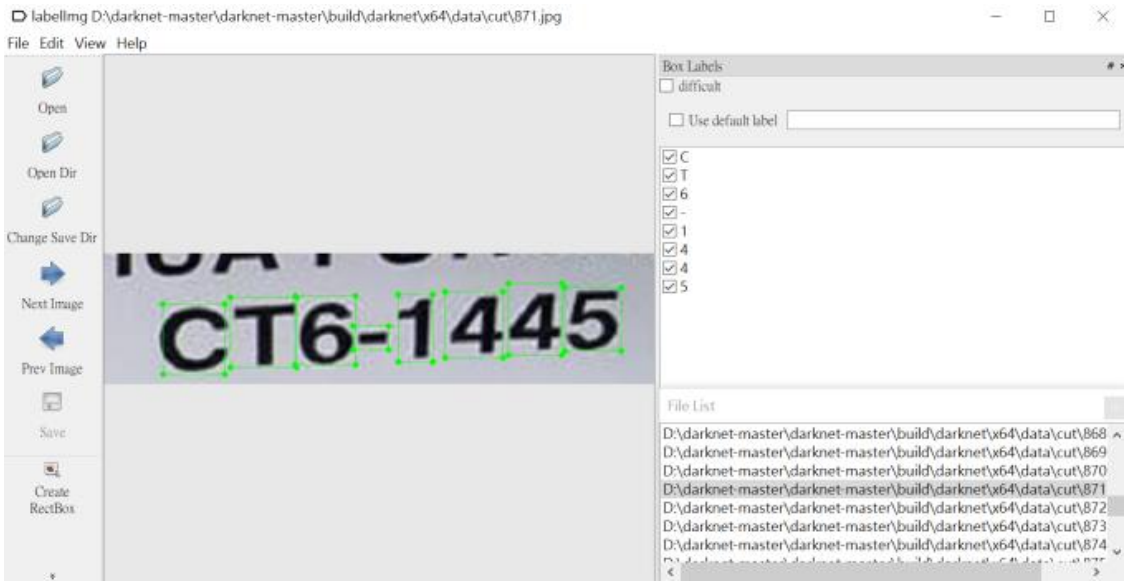


圖 15　Lablimg 標記標籤及邊界框

871.xml - 記事本

檔案(F)　編輯(E)　格式(O)　檢視(V)　說明

```
<annotation>
        <folder>cut</folder>
        <filename>871.jpg</filename>
        <path>D:\darknet-master\darknet-master\build\darknet\x64\data\cut\871.jpg</path>
        <source>
                <database>Unknown</database>
        </source>
        <size>
                <width>400</width>
                <height>95</height>
                <depth>3</depth>
        </size>
        <segmented>0</segmented>
        <object>
                <name>C</name>
                <pose>Unspecified</pose>
                <truncated>0</truncated>
                <difficult>0</difficult>
                <bndbox>
                        <xmin>43</xmin>
                        <ymin>37</ymin>
                        <xmax>88</xmax>
                        <ymax>88</ymax>
                </bndbox>
                <extra/>
        </object>
        <object>
                <name>T</name>
                <pose>Unspecified</pose>
                <truncated>0</truncated>
```

第 1 列，第 1 行　80%　Unix (LF)　UTF-8

圖 16　Lablimg 產生的 xml 檔 (記載圖片大小，標記類別，邊界框四點座標)

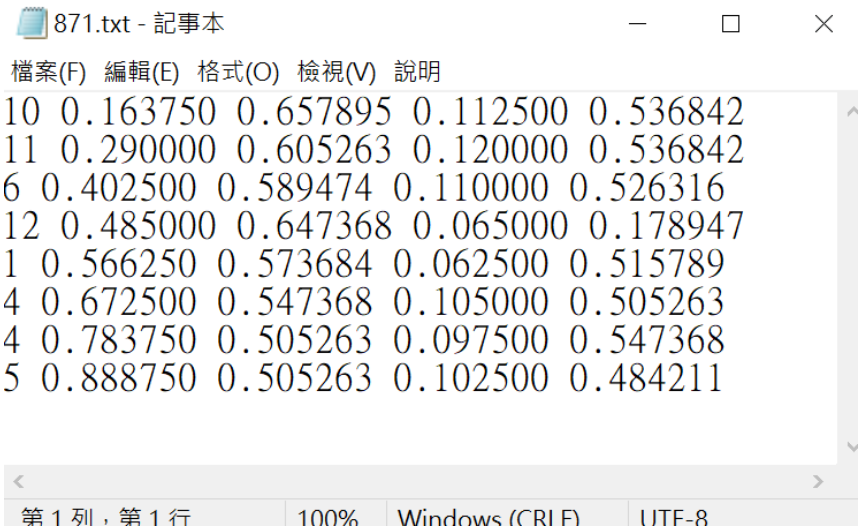


圖 17　YOLO 訓練所需 txt 檔(第一項為第幾類別，第二和第三項為邊界框中心點 x 和 y 座標除以圖片寬與高，第四和第五項為邊界框寬與高在圖片的寬與高)

3.3　直接船號辨識

在影像辨識上，一開始會希望可以建立一個步驟就同時完成偵測與辨識的模型，而 YOLO 深度學習網路本身就屬於一階段辨識。YOLO 深度學習網路在經過訓練後，輸入影像就可以同時得到辨識位置與結果，因此在本研究使用 YOLO 網路進行漁船船號的偵測與辨識。

使用一個 YOLO 深度學習網路進行訓練，輸入端為漁船影像(如圖 19)，輸出端為辨識結果的漁船船號(如圖 20)。漁船訓練集總計 705 張，以 7:2:1 分配訓練、驗證、測試集。將漁船訓練集標記 13 種類別(0~9, C, T, -)讓 YOLO 模型進行訓練，以直接辨識出圖片中的漁船編號。

圖 18　直接船號辨識流程圖

圖 19　直接船號辨識前

圖 20　直接船號辨識後

3.4 Region_based 船號辨識

由於漁船編號在大部分漁船影像中所佔的比例太小，直接辨識效果不佳，因此使用 Region_based 船號辨識把辨識分為兩階段:先從漁船影像中找出船號區域，再從船號區域中辨識漁船編號。兩階段都是使用 YOLO 網路進行訓練，第一階段輸入漁船影像(如圖 22)，輸出漁船船號區域(如圖 23)，第二階段輸入漁船船號區域(如圖 24)，輸出船號辨識結果(如圖 25)。由於第二階段輸入影像不一樣，因此需要再從漁船資料集建立一個船號區域資料集。

一開始先將漁船訓練集標記二種類別(中文一類, 英文及數字一類)讓 YOLO 網路進行訓練，找出漁船編號區域(英文與數字)。再將訓練集與驗證集的漁船影像輸入,得到第一階段辨識結果的位置並存取下來(存取下來的文字檔如圖 26)，再使用 Python 讀取並將漁船照片中漁船編號區域切割下來後存取成圖片，形成漁船編號區域訓練集。最後將漁船編號區域訓練集標記十三種類別(0~9, C,T, -)讓 YOLO 網路進行訓練，辨識船號區域圖片中的漁船編號。

在標記船號區域時，由於是分成中文、英文數字兩個類別，會連英文船名和國際識別編號一起標記。國際識別編號為漁船的電台呼號或國家辨識碼（B）和連字號（-）連結漁船統一編號(例:B-CT99999)，只有國外作業漁船或二十噸以上延繩釣漁船才會有，位置會標示在船身。也因為英文船名和國際識別編號標示的位置漁船統一編號也有可能出現，因此一起標記成一個類別，但因為英文船名和國際識別編號不是辨識目標而且訊練資料太少，所以不在船號區域訓練集中。

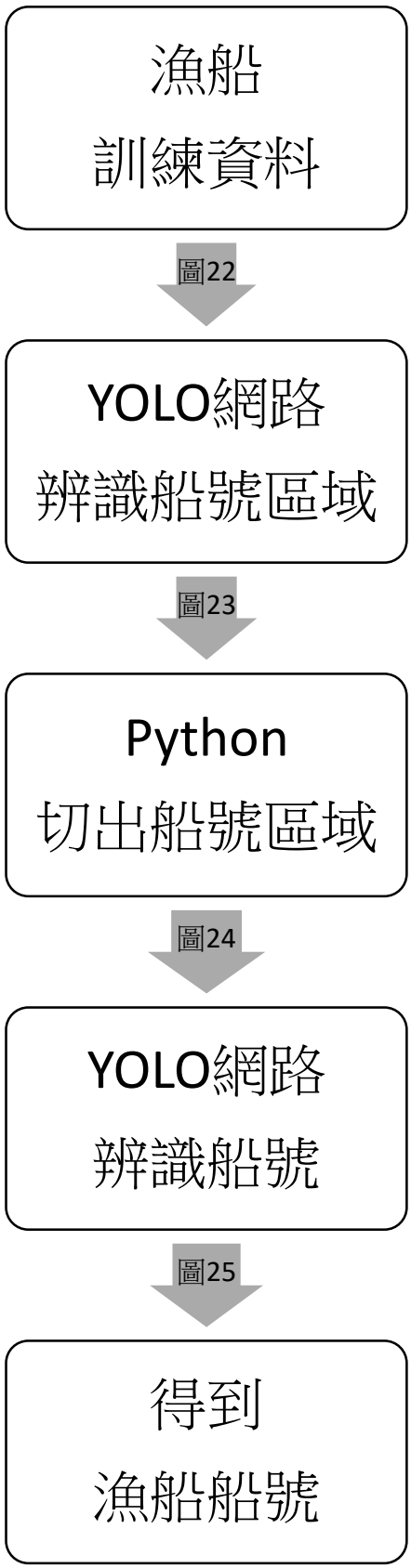


圖 21　Region_based 船號辨識流程圖

圖 22　船號區域辨識前

圖 23　船號區域辨識後

圖 24　船號辨識前

圖 25　船號辨識後

```
test_pics/0005.j
txt2: 97%       (left_x:  167   top_y:   81   width:   68   height:   24)
txt: 36%        (left_x:  201   top_y:  105   width:   67   height:   32)
test_pics/0008.j
txt: 98%        (left_x:  712   top_y:  612   width:  510   height:  162)
txt2: 99%       (left_x:  895   top_y:  723   width:  220   height:   76)
test_pics/0011.j
txt2: 98%       (left_x:  298   top_y:  211   width:  165   height:   50)
txt: 76%        (left_x:  468   top_y:  191   width:  166   height:   62)
txt: 47%        (left_x:  586   top_y:  191   width:   61   height:   48)
test_pics/0013.j
txt2: 94%       (left_x:  916   top_y:  696   width:  214   height:   80)
txt: 96%        (left_x:  933   top_y:  625   width:  219   height:   91)
test_pics/0016.j
txt2: 96%       (left_x:  252   top_y:  229   width:   56   height:   53)
txt: 94%        (left_x:  255   top_y:  196   width:   56   height:   59)
test_pics/0017.j
txt2: 42%       (left_x:   23   top_y:  124   width:   79   height:   47)
txt2: 99%       (left_x:  243   top_y:  107   width:  122   height:   31)
test_pics/0025.j
txt: 89%        (left_x: 1627   top_y:  832   width:  409   height:  136)
txt2: 84%       (left_x: 1978   top_y:  893   width:  458   height:   86)
test_pics/0034.j
txt: 98%        (left_x:  315   top_y:  634   width:  405   height:  140)
txt2: 79%       (left_x:  725   top_y:  659   width:  411   height:  101)
```

圖 26　二階段第一步辨識結果文字檔　(左至右為類別,置信度,預測框的最左方座標,最上方座標,寬度,高度)

3.5　改良式 Region_based 船號辨識

由於漁船編號的字體種類並未統一規定，因此有些小型漁船的漁民在漁船編號風吹日曬而剝落後會自行補上。但因為書寫者寫字習慣漁船編號會帶有角度的偏移。再加上漁船船頭是曲面的，拍攝漁船照片時漁船編號會跟著漁船帶有弧度而變形，只使用 Region_based 船號辨識無法有好的辨識率，因此需要更進一步進行改良。

沿用 Region_based 船號辨識的架構，不同的地方在於辨識船號後，再使用 Python 將未成功辨識船號區域的圖片進行旋轉，旋轉範圍在正負 30 度以內(若船號歪斜角度大於 30 度，則繼續增加旋轉角度)每旋轉 1 度存取成一張圖片，再將旋轉後船號區域圖片每一張都進行船號辨識，再找出最佳辨識結果。

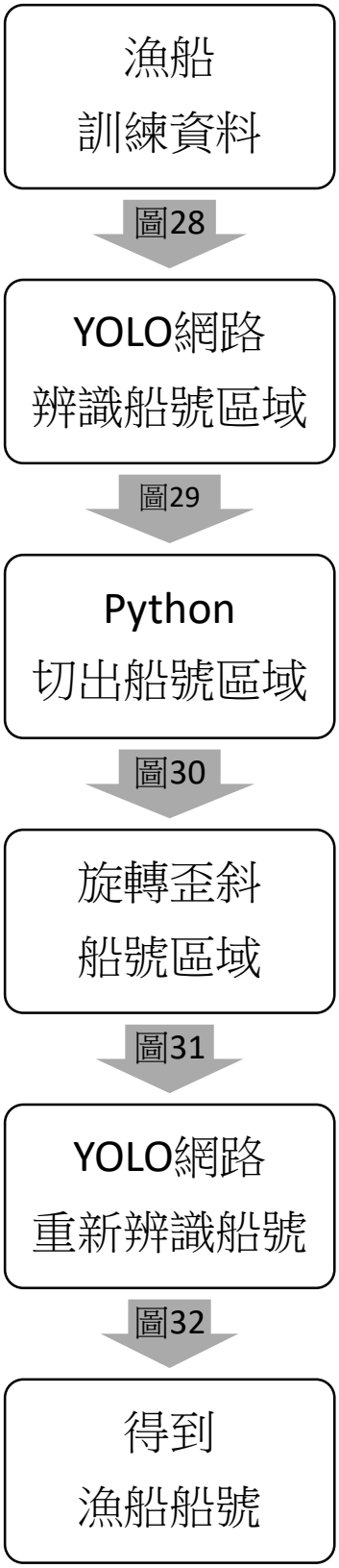


圖 27　改良式 Region_based 船號辨識流程圖

圖 28　船號區域辨識前

圖 29　船號區域辨識後

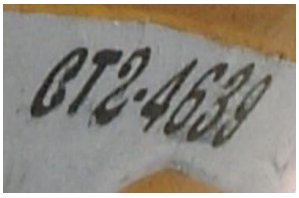


圖 30　因歪斜無法辨識之船號區域 (CT2-4639)

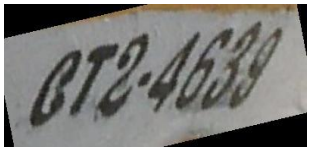


圖 31　旋轉歪斜船號區域

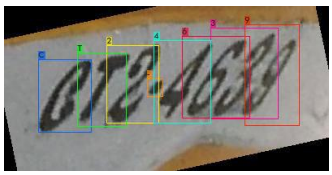


圖 32　辨識船號

3.6 輸入影像尺寸選擇

輸入影像尺寸的大小會影響深度學習網路的效能。影像尺寸愈大，會在訓練時保留較多的細節，但會增加深度學習網路的運算量。相反的，使用小尺寸影像進行訓練，運算速度很快，但對於小目標很多的影像準確度也會跟著下降。

本研究以 YOLOv3 深度學習網路和 Region_based 船號辨識系統進行測試，選擇 480×480、416×416、352×352 和 288×288 四種影像尺寸進行測試，在找出船號區域和辨識船號的 YOLO 網路都使用相同的影像尺寸，測試結果如表 1。

表 1　影像尺寸影響

YOLOv3	480×480	416×416	352×352	288×288
測試資料字元辨識率	88.34%	83.09%	82.10%	81.61%
整體船號正確率	75.32%	66.23%	64.94%	66.23%

測試結果輸入影像為 480×480 時不管是測試資料字元辨識率還是整體船號正確率都是最高，輸入尺寸愈小，測試資料字元辨識率愈低，且直接船號辨識中的船號大部分都是小目標，為了盡量保留船號特徵，輸入圖片尺寸要愈大愈好，因此選定輸入影像為 480×480。

3.7 Subdivisions 選擇

在 YOLO 網路中，Batch 代表每一次迭代送進網路訓練的圖片數量，當內存不夠大時，可以增加 Subdivisions，Subdivisions 代表將 Batch 分成幾份，每次送進 Batch/Subdivisions 張影像，送了 Subdivisions 次才完成一次迭代。當使用 YOLOv3 在 Region_based 二階段進行船號辨識的訓練時，在 Batch 為 16，Subdivisions 也選用 16 時會無法正常運算(如圖 33)。在經過實驗以後，發現不管 Batch 設定在 8、16、32，只要 Subdivisions 選用的數字大於 Batch 的二分之一都無法良好的進行訓練，map 在訓練 2000 至 3000 次以後就會開始往下掉，在 Subdivisions 為 Batch 一半時才不會發生異常(如圖 34)。因此最後的 YOLO 網路 Batch 選用 16，Subdivisions 使用 8 來進行訓練。

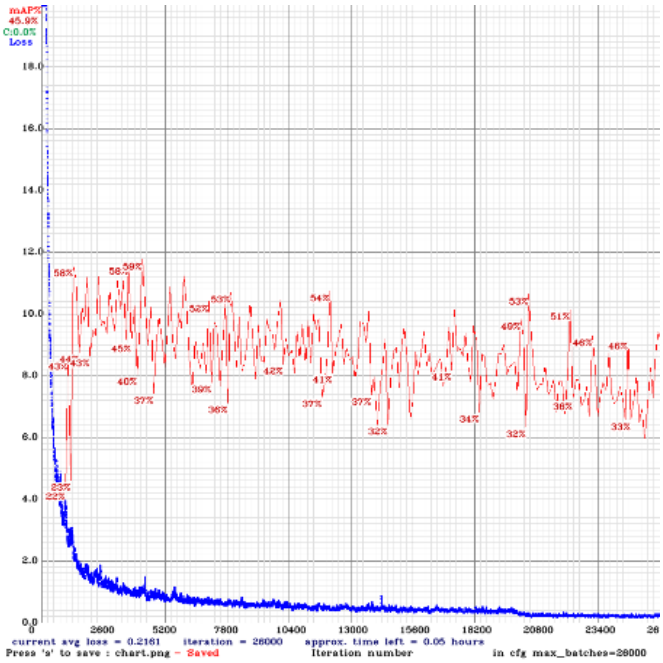


圖 33　YOLOv3 船號辨識 Batch:16, Subdivisions:16 訓練過程 map

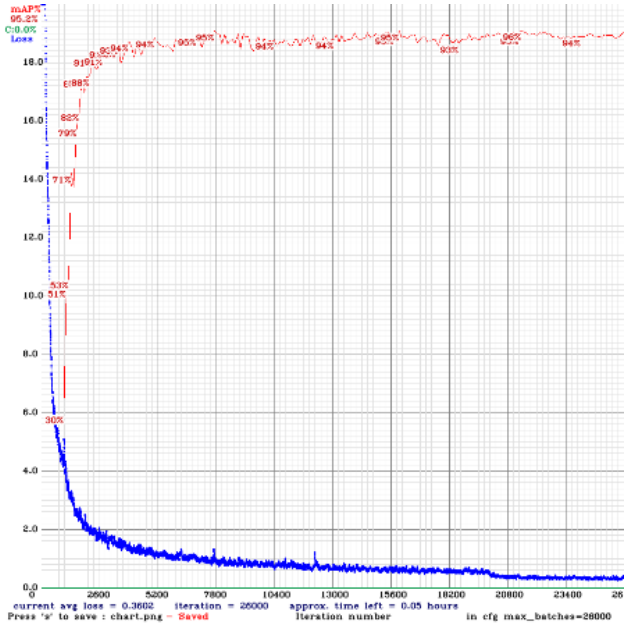


圖 34　YOLOv3 船號辨識 Batch:16, Subdivisions:8 訓練過程 map

3.8　深度學習網路效能評估

本研究將在具有 NVIDIA RTX3080 顯示卡的高效能電腦進行實驗。輸入具有漁船船號的漁船影像，透過 darknet 建置 YOLO 神經網路模型，訓練好後再將即時影像輸入訓練好的網路進行船號辨識。

對於深度學習網路訓練的模型，會使用精準度(Precision)、召回率（Recall）

和平均精度(Average precision,ap)三個指標進行評估。
精準度與召回率定義如下:

$$\text{Precision} = \frac{\text{TP(預測為目標又正確)}}{\text{TP} + \text{FP(預測為目標)}} \tag{4}$$

$$\text{Recall} \quad = \frac{\text{TP(預測為目標又正確)}}{\text{TP} + \text{FN(真實樣本是目標)}} \tag{5}$$

其中，TP(True Positive), FP(False Positive), FN(False Negative)及 TN(True Negative)的概念如表 2 的混淆矩陣(Confusion Matrix)所示，TP 代表真實樣本是目標且預測結果也是目標；FN 為真實樣本是目標，但預測結果不是目標；FP 為真實樣本不是目標，但預測結果是目標；TN 為真實樣本不是目標，且預測結果也不是目標。

表 2　混淆矩陣

	實際有	實際沒有
預測有	TP	FP
預測沒有	FN	TN

在一般情況下，準確率跟召回率會互相影響。精準度高召回率就會低，反之亦然，因此將精準度與召回率兩個數值分開來看並不適合判別整體效能。將精準度與召回率的關係畫成曲線圖(PR curve)，在曲線下方的曲面面積即為平均精度。利用平均精度可清楚比較個別框架的偵測效能。
另外於效能評估中，使用 IOU 作為在資料集中檢測相應物體準確度的一個標準。IOU 代表預測框與被標記的目標物交集與聯集的比值，也就是預測與實際的相關度。相關度愈高，IOU 愈高。通常以 0.5 作為一個閾值，大於此閾值為 TP，反之則為 FP。

第四章　實驗結果

實驗環境如表 3，操作系統使用 Linux Ubuntu 18.04，處理器使用 Intel i7-8700K，記憶體使用 DDR4 2666 16GB，顯示卡使用 NVIDIA RTX 3080，操作的軟體是 Darknet。

對於四種 YOLO 深度學習網路會採用平均精度(Mean Average Precision,map)、每秒幀數(Frames Per seconds,fps)十億浮點數運算量(Billion Floats Operations,BFLOPs)進行效能、速度、運算量的比較。並比較直接、Region_based、改良式 Region_based 三種方法在測試集的字元正確率與整體船號的正確率。測試集中有 70 張漁船影像，77 個船號區域。實驗中圖片尺寸皆設定為 480×480,Batch 皆設定為 16,subdivisions 皆設定為 8,Max Batches 辨識船號區域設定為 6000，辨識船號皆設定為 26000，learning rate 皆設定為 0.001。每類別在訓練資料與測試資料中的數目如表 4 所示。

表 3　實驗環境

OS	Linux Ubuntu 18.04
CPU	Intel i7-8700K
RAM	DDR4 2666 16GB
Graphics	NVIDIA RTX 3080
Software	Darknet

表 4　類別資料數

資料數 類別	訓練與驗證集 資料數	測試集 資料數
0	361	42
1	477	50
2	288	51
3	294	35
4	275	21
5	189	37
6	455	58
7	176	30
8	218	25
9	191	31
C	583	77
T	577	77
-	561	75
total	4644	609

4.1　直接船號辨識實驗結果

本實驗 YOLO 網路直接船號辨識時的效能與正確率比較。表 4 為直接船號辨識網路效能，其中 FPS 的值是在輸入 1080P 影片得出，圖 35 為網路訓練過程 map 的變化，x 軸為訓練次數，y 軸為 map，圖 36 為網路的 PR curve，表 5 為測試資料字元正確率，表 6 為以 YOLOv4 進行測試的混淆矩陣，表 7 為測試資料整體船號正確率，即船號區域中的所有字元皆正確才會算作正確。

表 5　直接船號辨識網路效能

網路 / 效能值	v3	v3-tiny	v4	v4-tiny
map	40.88%	9.48%	54.01%	3.20%
BFLOPs	87.060	7.278	79.416	9.061
FPS	131.6	700.7	103.2	633.3

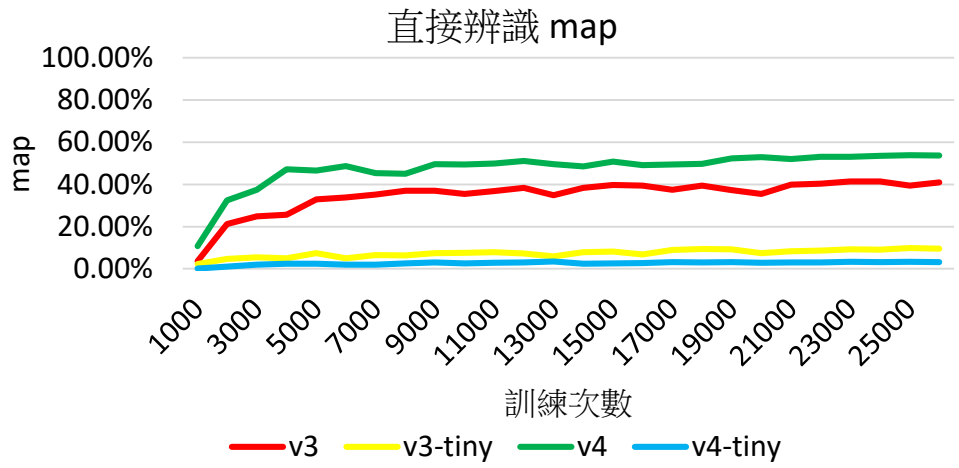


圖 35　直接船號辨識網路訓練過程 map

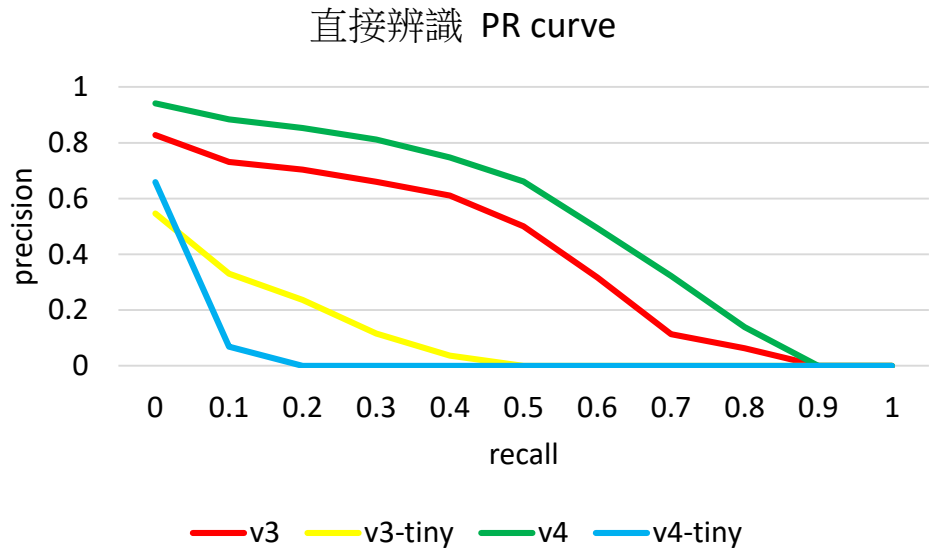


圖 36　直接船號辨識網路 PR curve

表 6　直接船號辨識測試資料字元正確率

網路 / 類別	v3	v3-tiny	v4	v4-tiny
0	47.62%	7.14%	52.38%	2.38%
1	36.00%	8.00%	38.00%	0.00%
2	35.29%	9.80%	33.33%	5.88%
3	22.86%	8.57%	22.86%	5.71%
4	42.86%	0.00%	42.86%	4.76%
5	35.14%	0.00%	40.54%	5.41%
6	41.38%	8.62%	44.83%	1.72%
7	43.33%	13.33%	43.33%	0.00%
8	36.00%	8.00%	24.00%	4.00%
9	32.26%	9.68%	41.94%	3.23%
C	71.43%	33.77%	71.43%	7.79%
T	46.75%	15.58%	45.45%	6.49%
"-"	53.33%	13.33%	46.67%	1.33%
total	44.83%	12.64%	44.83%	3.94%

表 7　YOLOv4 直接辨識測試結果混淆矩陣

真實 / 預測	0	1	2	3	4	5	6	7	8	9	C	T	-
0	23	0	1	0	0	0	0	0	0	0	0	0	
1	0	17	0	1	0	0	0	0	0	0	0	1	1
2	0	0	14	0	0	0	0	1	0	0	0	0	0
3	0	0	2	8	0	3	0	0	1	1	0	1	0
4	0	0	1	0	9	1	0	0	0	0	0	0	0
5	0	0	0	1	0	15	0	0	0	0	0	0	1
6	2	0	0	0	0	1	26	0	3	0	0	0	0
7	0	0	0	0	0	0	0	13	0	0	0	1	0
8	0	0	0	1	0	0	0	0	6	2	0	0	0
9	0	0	0	0	0	0	1	0	0	14	0	0	0
C	1	0	0	0	0	0	0	0	0	0	56	1	0
T	0	0	0	0	0	0	0	0	0	0	2	35	0
-	1	0	0	0	0	1	1	0	0	0	0	0	34
未辨識	15	33	33	24	12	16	30	16	15	14	19	38	39
總數	42	50	51	35	21	37	58	30	25	31	77	77	75
正確率(%)	54.8	34.0	27.5	22.9	42.9	40.5	44.8	43.3	24.0	45.2	72.7	45.5	45.3

表 8　直接船號辨識整體船號正確率

正確率 / 網路	整體船號正確率
v3	14.3%
v3-tiny	0%
v4	18.2%
v4-tiny	1.2%

由實驗結果得知 YOLOv3 和 v4 因為網路層較 v3-tiny 和 v4-tiny 多所以正確率較高，v3-tiny 和 v4-tiny 對於小目標幾乎沒有辨識能力。在訓練大約 5000 次以後，map 便不會明顯增加，其中又以 YOLOv4 的 map 比較高，運算量最少和運算速度最快皆為 YOLOv3-tiny，測試資料字元辨識率以類別 C 的辨識率最高，因為類別 C 訓練資料較多，訓練數量與 C 接近的 T 和 - 辨識結果容易包含彼此因而辨識錯誤。從混淆矩陣可以得知直接辨識結果大部分都無法辨識，但不管是何種 YOLO 網路，在輸入圖片是完整漁船影像的情況下，直接辨識漁船編號訓練的效果都不佳，測試的字元辨識率連 50%都不到，整體船號的正確率也相當低，因此需要使用 Region_based 船號辨識將過程分成兩階段進行漁船船號辨識。

4.2 Region_based 船號辨識實驗結果

本實驗為 YOLO 網路在 Region_based 船號辨識時的效能與正確率比較。表 9 為 Region_based 辨識船號區域的網路效能值，由於辨識船號區域的兩個類別中，英文及數字才是我們想要的，所以也要將英文及數字的 ap 加入比較，圖 37 為辨識船號區域的網路訓練過程 map 的變化，圖 38 為辨識船號區域網路的 PR curve，表 10 為辨識船號的網路效能值，圖 39 為辨識船號的網路訓練過程 map 的變化，圖 40 為辨識船號網路的 PR curve，表 11 為測試資料字元正確率，表 12 為以 YOLOv4 進行測試的混淆矩陣，表 13 為測試資料整體船號正確率。

表 9　Region_based 船號區域辨識網路效能值

網路 / 效能值	v3	v3-tiny	v4	v4-tiny
map	91.55%	84.52%	92.82%	91.81%
英文及數字 ap	89.63%	83.88%	91.94%	89.69%
BFLOPs	86.953	7.255	79.310	9.038
FPS	131.6	705.8	103.0	651.3

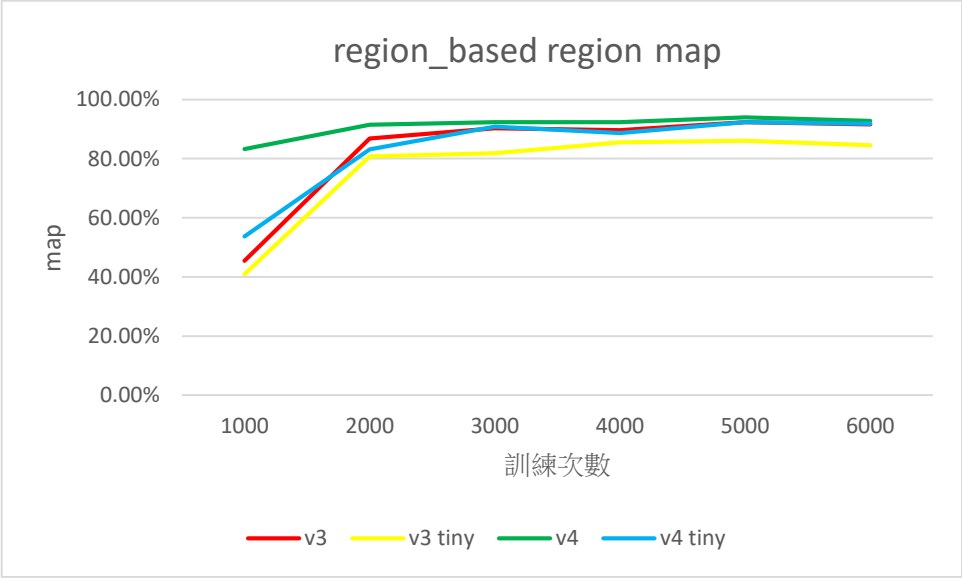


圖 37　Region_based 船號辨識第一階段網路訓練過程 map

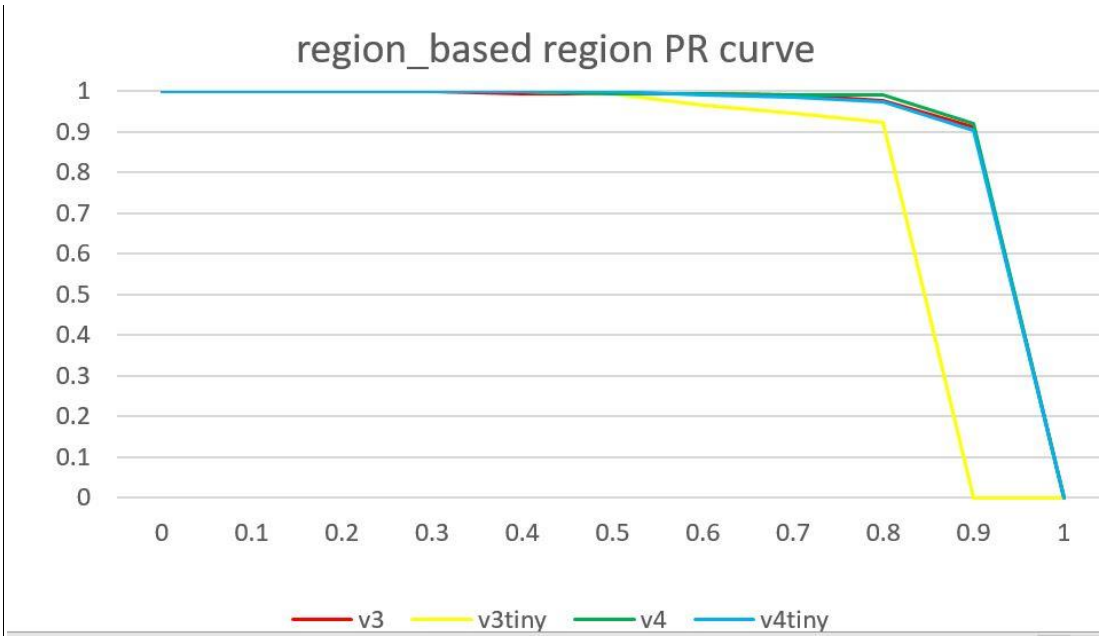


圖 38　Region_based 船號辨識第一階段網路 PR curve

表 10　Region_based 船號辨識第二階段網路效能

網路 / 效能值	v3	v3-tiny	v4	v4-tiny
map	94.74%	92.62%	96.36%	93.84%
BFLOPs	87.060	7.278	79.416	9.061
fps	131.5	705.7	102.9	642.8

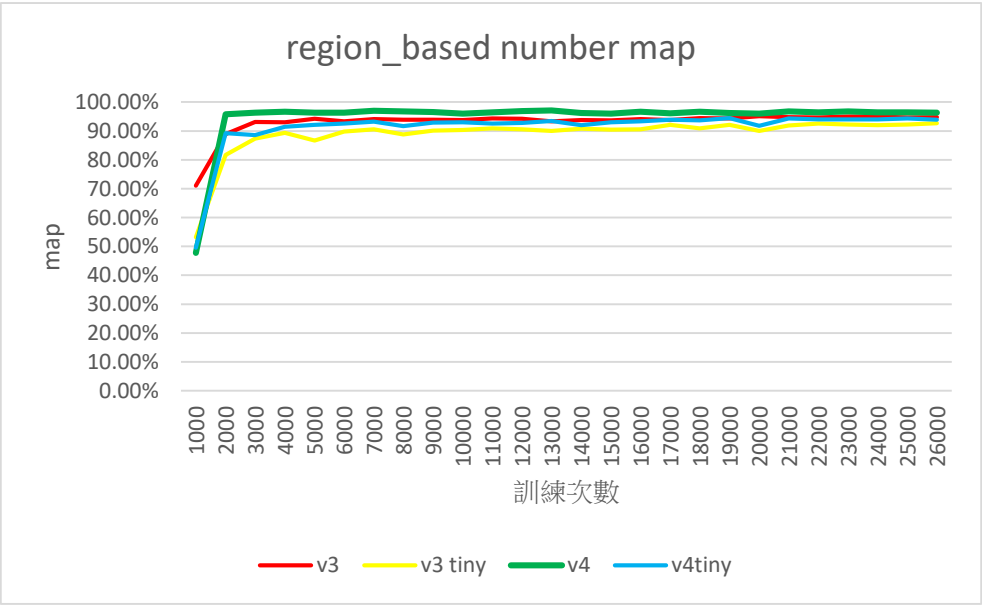


圖 39　Region_based 船號辨識第二階段網路 map

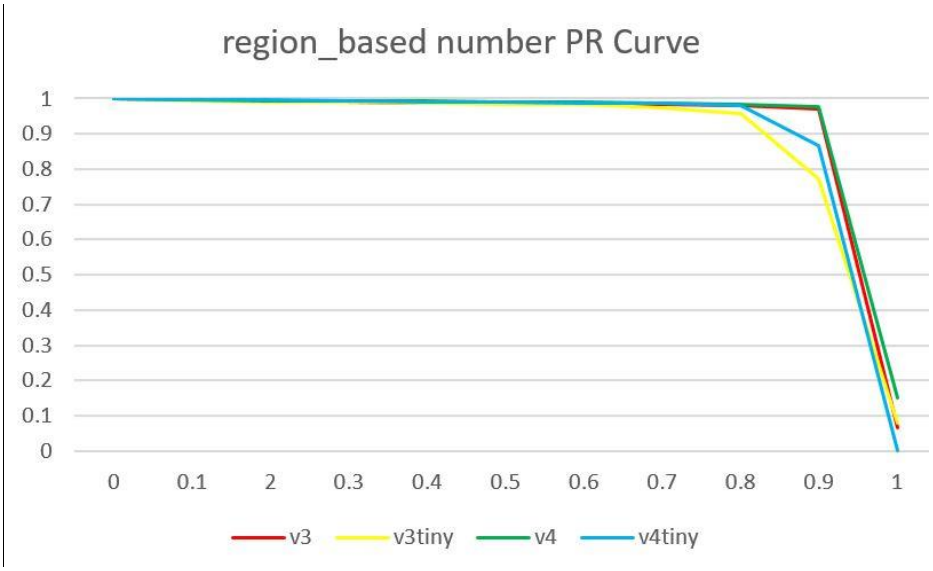


圖 40　Region_based 船號辨識第二階段網路 PR curve

表 11　Region_based 船號辨識測試資料字元正確率

網路 / 類別	v3	v3-tiny	v4	v4-tiny
0	88.10%	76.19%	83.33%	90.48%
1	98.00%	80.00%	96.00%	96.00%
2	82.35%	64.71%	86.27%	84.31%
3	80.00%	57.14%	82.86%	65.71%
4	80.95%	71.43%	95.24%	80.95%
5	78.38%	59.46%	91.89%	81.08%
6	93.10%	81.03%	96.55%	87.93%
7	86.67%	70.00%	90.00%	86.67%
8	76.00%	64.00%	96.00%	80.00%
9	90.32%	83.87%	96.77%	93.55%
C	90.91%	77.92%	94.86%	90.91%
T	90.91%	79.22%	96.10%	92.21%
"-"	96.00%	81.33%	96.00%	88.00%
total	88.83%	74.55%	92.94%	87.36%

表 12　YOLOv4 Region_based 辨識測試結果混淆矩陣

真實 / 預測	0	1	2	3	4	5	6	7	8	9	C	T	-
0	35	0	0	0	0	1	0	0	0	0	0	0	0
1	0	48	0	0	0	0	0	1	0	0	0	0	0
2	0	0	44	1	0	0	0	0	0	0	0	0	0
3	0	0	0	29	0	1	0	0	0	0	0	0	0
4	0	0	0	0	20	0	0	0	0	0	0	0	0
5	0	0	0	1	0	34	0	0	0	0	0	0	0
6	3	0	0	0	0	0	56	0	0	0	0	0	0
7	0	1	1	0	0	0	0	27	0	0	0	0	0
8	0	0	0	0	0	0	0	0	24	1	0	0	0
9	2	0	1	0	0	1	0	0	0	30	0	0	0
C	0	0	0	0	0	0	0	0	0	0	73	0	0
T	0	0	0	0	0	0	0	10	0	0	0	74	0
-	0	0	0	0	1	0	0	0	0	0	0	0	72
未辨識	2	1	5	4	0	0	2	1	1	0	4	3	3
總數	42	50	51	35	21	37	58	30	25	31	77	77	75
正確率(%)	83.3	96.0	86.3	82.9	95.2	91.9	96.6	90.0	96.0	96.8	94.8	96.1	96.0

表 13　Region_based 船號辨識船號正確率

正確率 / 網路	整體船號正確率
v3	72.7%
v3-tiny	46.8%
v4	77.9%
v4-tiny	64.9%

從混淆矩陣可以得知,同樣使用 Region_based 的系統,類別 5 和 8 在 YOLOv3 的辨識率是較低的，因為訓練資料較少且與其他類別較為相像，但在 YOLOv4 類別 5 和 8 的辨識率沒有明顯低於平均，這代表 YOLOv4 可以用較少訓練資料達到比 YOLOv3 更好的辨識率。不管是辨識船號區域還是辨識船號的 YOLO 深度學習網路,在訓練 3000 次以後,map 就不會明顯的增加,其中又以 YOLOv4 有最高的 map，運算量最少和運算速度最快皆為 YOLOv3-tiny，但 map 在四種 YOLO 網路中是最低的。由於 YOLOv3-tiny 尋找船號區域的 YOLO 網路較多船號區域未辨識出來，因此最終字元辨識率較低。使用 Region_based 船號辨識對於 YOLO 網路來說雖然

會提升運算量和增加運算時間，但能有效提升字元正確率和整體船號正確率，未辨識的數量大幅減少，辨識率最好的 YOLOv4 可以有 92.9%的正確率。但以完整船號辨識正確率來看，YOLOv4 只有 77.9%。因此需要再將 Region_based 船號辨識進行改良。

4.3 改良式 Region_based 船號辨識實驗結果

本實驗為檢驗 YOLO 網路在改良式 Region_based 船號辨識系統的效能比較。YOLO 網路架構與 Region_based 船號辨識相同，差別在於船號區域校正後的辨識的結果。表 14 為改良式 Region_based 船號辨識測試資料字元正確率，表 15 為以 YOLOv4 進行測試的混淆矩陣，表 16 為改良式 Region_based 船號辨識測試資料整體船號正確率。

表 14 改良式 Region_based 船號辨識測試字元正確率

網路 / 類別	v3	v3-tiny	v4	v4-tiny
0	90.48%	80.95%	90.48%	92.86%
1	100.00%	88.00%	100.00%	100.00%
2	84.31%	74.51%	90.20%	90.20%
3	82.86%	60.00%	88.57%	82.86%
4	90.48%	76.19%	95.24%	85.71%
5	94.59%	64.86%	97.30%	91.89%
6	96.55%	86.21%	98.28%	93.10%
7	96.67%	80.00%	96.67%	90.00%
8	80.00%	76.00%	96.00%	88.00%
9	96.77%	87.10%	100.00%	96.77%
C	92.21%	83.12%	96.10%	93.51%
T	90.91%	85.71%	96.10%	94.81%
"-"	96.00%	84.00%	97.33%	92.00%
total	92.28%	80.46%	95.73%	92.45%

表 15　YOLOv4 改良式 Region_based 辨識測試結果混淆矩陣

真實 / 預測	0	1	2	3	4	5	6	7	8	9	C	T	-
0	38	0	0	0	0	0	0	0	0	0	0	0	0
1	0	50	0	0	0	0	0	0	0	0	0	0	0
2	0	0	46	0	0	0	0	0	0	0	0	0	0
3	0	0	0	31	0	0	0	0	0	0	0	0	0
4	0	0	0	0	20	0	0	0	0	0	0	0	0
5	0	0	0	1	0	36	0	0	0	0	0	0	0
6	1	0	0	0	0	1	57	0	0	0	0	0	0
7	0	0	0	1	0	0	0	29	0	0	0	0	0
8	1	0	1	0	0	0	0	0	24	0	0	0	0
9	1	0	0	0	0	0	0	0	0	31	0	0	0
C	0	0	0	0	0	0	0	0	0	0	74	0	0
T	0	0	0	0	0	0	0	0	0	0	0	74	1
-	0	0	0	0	1	0	0	0	0	0	0	0	73
未辨識	1	0	4	2	0	0	1	1	1	0	3	3	1
總數	42	50	51	35	21	37	58	30	25	31	77	77	75
正確率 (%)	90.5	100	90.2	88.6	95.2	97.3	98.3	96.7	96.0	100	96.1	96.1	97.3

表 16　改良式 Region_based 船號辨識測試整體船號正確率

正確率 / 網路	整體船號正確率
v3	85.7%
v3-tiny	61.0%
v4	88.3%
v4-tiny	81.5%

從字元辨識率和整體船號正確率可以得知不管是何種 YOLO 網路，在校正船號區域圖片後可以更進一步提升船號辨識正確率，YOLOv3 和 v4 提升了 2 到 4%，v3-tiny 和 v4-tiny 提升了 5 到 6%。在所有 YOLO 網路中，又以 YOLOv4 有最佳正確率，從混淆矩陣來看，辨識錯誤的數量也有所減少。因此在本研究中以正確率

來看，使用 YOLOv4 的改良式 Region_based 船號辨識系統，會有最好的船號辨識準確率。

4.4 實驗結果討論

綜合上述實驗結果，將使用 4 種 YOLO 網路 v3、v3-tiny、v4、v4-tiny，比較 3 種系統:直接船號辨識、Region_based 船號辨識、改良式 Region_based 船號辨識，其正確率與效能比較如表 17 所示。以改良式 Region_based 船號辨識並以 YOLOv4 進行訓練不管是字元還是整體船號的正確率都最好，但運算量較其他 YOLO 網路大，因此會對硬體設備有較高的需求。若改使用 YOLOv4-tiny，雖然正確率略比 YOLOv4 低,但運算量大幅減少,裝置的運算能力要求就不需要這麼高，進而減少硬體成本。

表 17　三種辨識系統正確率比較

實驗方法 \ 正確率與效能		測試字元正確率	測試整體船號正確率
直接船號辨識	v3	44.8%	14.3%
	v3-tiny	12.6%	0%
	v4	44.8%	18.2%
	v4-tiny	3.9%	1.2%
Region_based 船號辨識	v3	88.8%	72.7%
	v3-tiny	74.5%	46.8%
	v4	92.9%	77.9%
	v4-tiny	87.4%	64.9%
改良式 Region_based 船號辨識	v3	92.3%	85.7%
	v3-tiny	80.5%	61.0%
	v4	95.7%	88.3%
	v4-tiny	92.4%	80.5%

圖 41 到圖 57 為測試資料中的三張影像使用每種辨識系統的辨識過程與辨識結果。深度學習網路使用 YOLOv4 進行，由於直接辨識辨識結果較小，因此將辨識結果放大附在原辨識結果旁。

範例一　船號:CT1-8176

<table>
<tr>
<td>直接辨識</td>
<td>


圖 41　範例一漁船影像未成功辨識結果


圖 42　範例一漁船影像結果(放大)。 8 辨識成 6，6 辨識成 9
</td>
</tr>
<tr>
<td>Region_based 船號辨識</td>
<td>

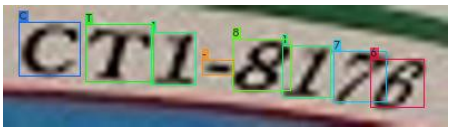

圖 43　範例一漁船影像成功辨識結果


圖 44　範例一漁船影像成功辨識結果
</td>
</tr>
</table>

範例二　船號:CT5-1567

<table>
<tr>
<td>直接辨識</td>
<td>

圖 45　範例二漁船影像辨識結果。 右舷船號沒辨識出來

圖 46　範例二漁船影像左舷成功辨識結果(放大)
</td>
</tr>
<tr>
<td>Region_based 船號辨識</td>
<td>

圖 47　範例二漁船影像成功辨識結果

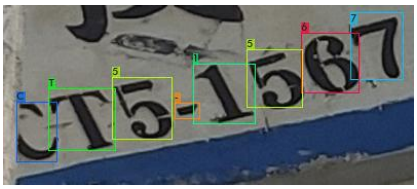



圖 48　範例二漁船影像左舷成功辨識結果

圖 49　範例二漁船影像辨識結果。 右舷 5 同時辨識成 5 和 3
</td>
</tr>
<tr>
<td>改良式 Region_based 船號辨識</td>
<td>

圖 50　範例二漁船影像校正船號字體傾斜角度

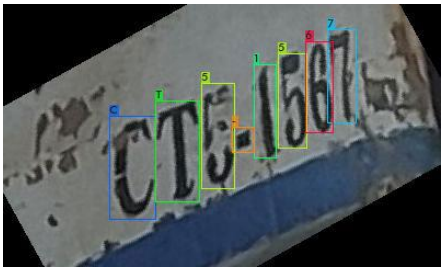



圖 51　範例二漁船影像右舷辨識成功結果
</td>
</tr>
</table>

範例三　船號:CT7-0012

<table>
<tr>
<td>直接辨識</td>
<td>

圖 52　範例三漁船影像未成功辨識結果

圖 53　範例三漁船影像辨識結果(放大)。 7、0、1、2 未辨識出
</td>
</tr>
<tr>
<td>Region_based 船號辨識</td>
<td>

圖 54　範例三漁船影像成功辨識結果

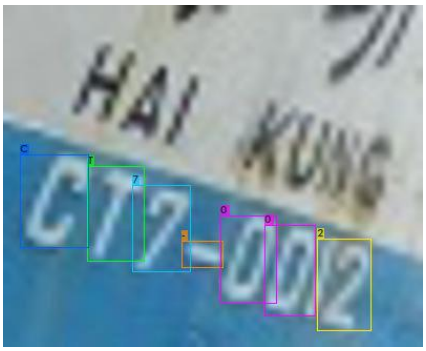



圖 55　範例三漁船影像辨識結果。 1 未辨識出來
</td>
</tr>
<tr>
<td>改良式 Region_based 船號辨識</td>
<td>

圖 56　範例三漁船影像校正船號字體傾斜角度

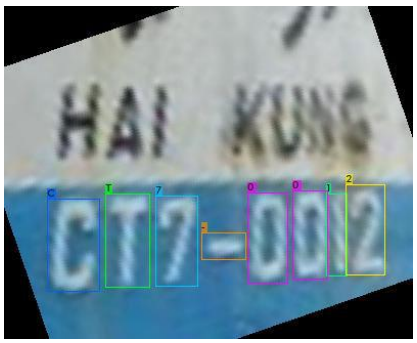



圖 57　範例三漁船影像成功辨識結果
</td>
</tr>
</table>

無法辨識圖片

圖 58 無法透過 YOLO 網路找出漁船上的文字區域，推測是檢測目標在圖片中所佔比例太小導致無法判別。為了判斷推測是否正確，本研究試著裁切圖片中一部份非漁船的背景，再將裁切後的漁船影像進行辨識，裁切結果如圖 59。實驗結果經過裁切後的漁船影像可以找出漁船上文字區域，且裁切的背景愈多，辨識結果的置信度就愈高。因此可以得知檢測目標須在檢測圖片中佔有一定比例，才能成功判別。

圖 58　無法找出船號區域之圖片

圖 59　無法找出船號區域之圖片經裁切後辨識結果。　成功辨識

圖 60 為 YOLO 網路無法辨識船號之船號區域圖片。上方應辨識之漁船統一編號未成功辨識，但圖片下方國際識別編號的數字卻可成功辨識，比較兩者差異可發現漁船統一編號字體顏色較淡也較模糊，且圖片經過旋轉切割後依舊無法辨識，辨識結果如圖 61 所示，因此不是字體歪斜的問題而是影像對 YOLO 網路來說太過模糊導致無法辨識。

圖 60　無法辨識船號圖片

圖 61　無法辨識船號圖片經旋轉切割辨識結果。　未成功辨識

第五章　結論與建議

5.1　結論

台灣是一個四面環海的海島，漁業更是台灣相當重要的經濟產業之一。在台灣，每天都有非常大量的漁船進出港，不過並沒有一個好的方法對於漁船進行監測。若以人力對於漁船進行監測，成本高且效率不佳。使用傳統影像處理方法，無法得到很好的辨識率。

本研究使用 YOLO 深度學習網路對漁船編號進行辨識，以 700 多張白天拍攝任意角度的台灣漁船影像進行訓練。YOLO 深度學習網路快速且準確度高，但直接對漁船影像進行辨識，會因為辨識目標船號太小而無法有很好的效果，因此本研究使用 Region_based 船號辨識:將辨識過程分為找出船號區域與辨識船號二階段，再更進一步進行改良，旋轉歪斜船號圖片，減少因圖片歪斜無法辨識船號圖片。

根據實驗結果顯示，以 YOLOv4 並用改良式 Region_based 船號辨識的系統不管是字元還是整體船號都會有最好的效果。字元正確率有 95.7%，整體船號的正確率也有 88.3%。若把運算量也加進考量，YOLOv4-tiny 用改良式 Region_based 系統雖然正確率較 YOLOv4 低，但運算量卻少 YOLOv4 非常多。雖然因為分成找出區域和辨識兩步驟，運算時間會比直接辨識多，運算量也會增加，但正確率卻有很大的提升，旋轉歪斜船號區域圖片也能有效減少因為歪斜漁船船號無法辨識的圖片。整體而言，如果沒有硬體的限制，透過 YOLOv4 並且使用改良式 Region_based 系統對漁船船號辨識有最佳的表現，若要減少硬體上的成本或設置在計算能力不高的裝置上，YOLOv4-tiny 也有不差的表現，字元正確率有 92.4%，整體船號的正確率也有 80.5%。

5.2　建議

在本研究中，只有使用 700 多張影像進行訓練，船號的種類也只有 13 種。若能增加更豐富更多船號種類且漁船統一編號清楚的漁船影像，對於船號的辨識也能更精準。如果能增加足夠數量且包含國際識別編號的漁船影像，甚至可建立一個同時辨識國際識別編號與漁船統一編號的辨識系統，增加系統的實用性。

參考資料

[1] Ferreira J.C., Branquinho J., Ferreira P.C., Piedade F. Computer vision algorithms fishing vessel monitoring—identification of vessel plate number. In: De Paz J., Julián V., Villarrubia G., Marreiros G., Novais P., editors. Ambient Intelligence—Software and Applications—8th International Symposium on Ambient Intelligence (ISAmI 2017); Advances in Intelligent Systems and Computing. Vol. 615. Springer International Publishing; New York, NY, USA, 2017.

[2] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” in CVPR, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2005

[3] J. Redmon, S. Divvala, R. Girshick, A. Farhadi “You Only Look Once: Unified, Real-Time Object Detection” Computer Science, arXiv 1506.02640, 2015

[4] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016

[5] J. Redmon and A. Farhadi. YOLO9000: better, faster, stronger. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 7263– 7271, 2017.

[6] J. Redmon and A. Farhadi, “Yolo9000: better, faster, stronger,” Computer Science,arXiv:1612.08242, 2016

[7] J. Redmon and A. Farhadi “YOLOv3: an incremental improvement,” Computer Science, arXiv 1804.02767, 2018

[8] A. Bochkovskiy, C.Y. Wang, H..Y. Mark Liao “YOLOv4: Optimal Speed and Accuracy of Object Detection” Computer Science, arXiv 2004.10934, 2020

[9] H. Gong, H. Li, K. Xu and Y. Zhang, "Object Detection Based on Improved YOLOv3-tiny," 2019 Chinese Automation Congress (CAC), 2019, pp. 3240-3245, doi: 10.1109/CAC48633.2019.8996750.

[10] C.Y. Wang, A. Bochkovskiy, H.Y. Mark Liao”Scaled-YOLOv4: Scaling Cross Stage Partial Network” Computer Science, arXiv 2011.08036,2021

[11] T. Y. Lin, P. Dollar, R. B. Girshick, K. He, B. Hariharan, and S. J. ´ Belongie, “Feature pyramid networks for object detection,” in CVPR, 2017.

[12] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, “The pascal visual object classes (voc) challenge,” International Journal of Computer Vision, vol. 88, Jun 2010.

[13] Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. Rectifier nonlinearities improve neural network acoustic models. In Proceedings of International Conference on Machine Learning (ICML), vol 30, pp. 3, 2013

[14] Diganta Misra. Mish: A self regularized nonmonotonic neural activation function. Computer Vision, arXiv 1908.08681, 2019.

[15] K. He, X. Zhang, S. Ren, and J. Sun, “Spatial pyramid pooling in deep convolutional networks for visual recognition,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 37, no. 9, pp. 1904–1916, 2015.

[16] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. Path aggregation network for instance segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 8759–8768, 2018.

[17] Zhaohui Zheng, Ping Wang, Wei Liu, Jinze Li, Rongguang Ye, and Dongwei Ren. Distance-IoU Loss: Faster and better learning for bounding box regression. In Proceedings of the AAAI Conference on Artificial Intelligence (AAAI), 2020.

[18] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. DropOut: A simple way to prevent neural networks from overfitting. The journal of machine learning research, vol 15,no.1,pp. 1929–1958, 2014.

[19] Golnaz Ghiasi, Tsung-Yi Lin, and Quoc V Le. DropBlock: A regularization method for convolutional networks. In Advances in Neural Information Processing Systems (NIPS), pages 10727–10737, 2018.

[20] Youngwan Lee, Joong-won Hwang, Sangrok Lee, Yuseok Bae, and Jongyoul Park. An energy and GPU-computation efficient backbone network for real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshop (CVPR Workshop), 2019.

[21] YOLOv3-引入：FPN+多尺度偵測(目標檢測)(one-stage)(深度學習)(CVPR 2018)
https://blog.csdn.net/Gentleman_Qin/article/details/84350496

[22] TF-YOLO: An Improved Incremental Network for Real-Time Object Detection(PDF)
https://www.researchgate.net/publication/335043703_TF-YOLO_An_Improved_Incremental_Network_for_Real-Time_Object_Detection

[23] CSPNET: A NEW BACKBONE THAT CAN ENHANCE LEARNING CAPABILITY OF CNN (PDF)
https://arxiv.org/pdf/1911.11929.pdf

[24] 【darknet】darknet——CSPDarknet53 網路架構圖（YOLO V4 使用）

https://blog.csdn.net/Jaredzzz/article/details/108560087
[25] MaskHunter: real-time object detection of face masks during the COVID-19 pandemic(PDF)
https://www.researchgate.net/publication/349558817_MaskHunter_real-time_object_detection_of_face_masks_during_the_COVID-19_pandemic
[26] Video-Based Human Action Recognition Using Spatial Pyramid Pooling and 3D Densely Convolutional Networks(PDF)
https://www.researchgate.net/publication/329147074_Video-Based_Human_Action_Recognition_Using_Spatial_Pyramid_Pooling_and_3D_Densely_Convolutional_Networks
[27] PANet 算法筆记
https://blog.csdn.net/u014380165/article/details/81273343
[28] YOLOv4
https://zhuanlan.zhihu.com/p/138510087
[29] 睿智的目標檢測 35——Pytorch 搭建 YoloV4-Tiny 目標檢測平台
https://www.javaxks.com/?p=75661
[30] 深入淺出 Yolo 系列之 Yolov3&Yolov4 核心基礎知識完整講解 (Mosaic)
https://kknews.cc/zh-hk/tech/9vl23r8.html
[31] NIPS 2018｜Quoc Le 提出卷積網絡專屬正則化方法 DropBlock
https://kknews.cc/tech/jebvrap.html
[32] An Architectural Multi-Agent System for a Pavement Monitoring System with Pothole Recognition in UAV Images(PDF)
https://www.researchgate.net/publication/345212328_An_Architectural_Multi-Agent_System_for_a_Pavement_Monitoring_System_with_Pothole_Recognition_in_UAV_Images
[33] YOLO 演進 — 4 — Scaled-YOLOv4
https://medium.com/ching-i/yolo%E6%BC%94%E9%80%B2-4-scaled-yolov4-c8c361b4f33f
[34] 洪銘鴻,改良式 YOLOv3 深度學習網路應用於船舶影像分類,國立臺灣海洋大學碩士論文,2020
[35] 洪銘鴻、張麗娜，“應用深度學習於船舶影像分類,” 中華民國系統工程研討會, Jun. 2020