屏東縣第 66 屆國中小學科學展覽會

作品說明書

科　　別：生活與應用科學科(一)

組　　別：國中組

作品名稱：　綠野蜥蹤

關 鍵 詞：YOLOv8、物件辨識、OpenCV

編號：**B6001**

摘要

本研究乃利用 YOLOv8 預設的模型權重訓練出能辨識綠鬣蜥的模型，進而在綠鬣蜥可能出沒的地點架設即時偵測的攝影機動態捕捉影像，當畫面中出現綠鬣蜥時，架設的電腦主機或是筆記型電腦能發出警告語音，並同時將偵測到的照片傳送電子郵件寄發給農民（或是捕蜥獵人）。

壹、 研究動機與文獻探討

一、研究動機

這幾年在校園內也頻繁地發現綠鬣蜥的蹤跡，甚至綠鬣蜥曾直闖一樓辦公室，老師和同學們都相當積極地在發現當下通報總務處，屏東縣政府農業處設置移除綠鬣蜥獎金。凡經訓練合格者，捕獲綠鬣蜥一隻視大小最少可兌換 100 元以上現金、等值禮券或農產品，如圖 1-1。某日夜間捕獵團隊在本校校園共捕獲 8 隻綠鬣蜥，如圖 1-2。

圖 1-1、2025 綠鬣蜥公定價

註記：取自《綠鬣蜥入侵 20 年 02》

圖 1-2、獵人於本校捕獲綠鬣蜥 114.10.14

註記：本校總務主任拍攝

而在鄰近學校的萬丹鄉種植紅豆的農民們在 9-10 月紅豆生長初期也是飽受綠鬣蜥啃食的禍害，綠鬣蜥為雜食性動物，喜好啃食嫩葉、芽、花與果實。而綠鬣蜥的作息時間和人類相近，老農民們甚至在正中午較易出沒時間點，拿著長形的農作工具來驅趕綠鬣蜥。

二、文獻探討

物件辨識的理論乃為深度學習中的「卷積神經網路」（簡稱 CNN），是一種監督式的學習，電腦透過標籤化的資訊中分析並做出預測，在學習的過程中比對誤差，一邊修正去做更精準的預測，目前主要的物件辨識主要分成 Two-Stage 和 One-Stage 兩種演算法。本研究採用的是 Ultralytics 公司提供的 YOLO（You Only Look Once）套件來發展出能夠偵測辨識綠鬣蜥的程式，YOLO 以 One-Stage 的方式亦即一個神經網路就能達成物件偵測與辨識。

而號稱「臺灣之光」的 YOLOv4，當時推出後便廣泛的應用在各個領域，尤其是偵測車輛、大卡車貨車的盲區、或是大型路口的人流等表現地相當優秀。本研究則採用相對穩定及多數用於物件辨識的 YOLOv8 來訓練偵測綠鬣蜥模型，YOLOv8 是以 PyTorch 函式庫來開發，然後在程式訓練深度學習中，需要大量的矩陣運算，若執行的電腦中的顯示卡配備 CUDA，搭配 CuDNN 針對 CUDA 做更高效能的加速，分配大量的矩陣運算任務給 GPU 的話，便能加速訓練模型。

目前國內的科學展覽中利用 YOLO 來探討的研究數量已相當可觀，郭毅尹等人（2024）透過 YOLOv8 辨識行人區的物件後，結合了 Arduino 控制 LED 燈來警示前方行人通過。林佑謙等人（2025）也是利用 YOLO 建立 LSTM 模型探討十字路口人車碰撞預測的情形以提升行人交通安全。此外，國內亦有碩士論文研究基於 YOLOv5 和 YOLOv8 以綠鬣蜥為目標偵測模型效能做比較（王竣彥，2025）。

藉由 YOLOv8 訓練辨識綠鬣蜥的模型，要了解訓練模型中的幾個名詞：

（一）IoU（intersection of union）：計算預測目標的框架和實際物件框架的重疊程度。其公式為預測框和實際物件框**交集**的面積除以預測框和實際物件框**聯集**的面積比值，如圖 2-1。

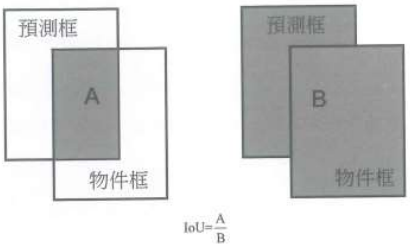


圖 2-1 IoU 的計算原理示意圖。註記：取自《台灣之光物件辨識最新 YOLO 原理精讀》

（二）AP（average precision）：預測為物件的平均精確度。會收集多種情形（例如只有偵測到綠鬣蜥的頭、尾巴或是爪子的框架，會導致 IoU 的值較小）的 Precision 數據後再做平均。同時會再計算出 recall（召回率）

（三）recall（召回率）：模型正確預測的目標數與實際目標數之比值。

（四）mAP（mean average precision）：衡量模型檢測性能的平均精確度，此為本研究所關注的重點之一，亦是大部分物件便是常用的指標，其數值越高，代表模型的辨識性能越好。

（五）mAP50（mean average precision at IoU 0.5 for Large Objects）：意即在 IoU 為 0.5 的情形下，針對較大目標計算的平均精確度的平均值。

貳、研究目的

外來種綠鬣蜥對人類的危害極大，造成農業損害，傳播細菌和寄生蟲，還破壞了生態平衡，本研究透過結合 YOLOv8 物件辨識技術與 OpenCV 影像處理技術之綠鬣蜥即時偵測與追蹤系統，提升野外生態監測的自動化與精準度。透過蒐集並標註綠鬣蜥影像，訓練客製化模型，使系統能在複雜自然環境中準確辨識目標並同步發送偵測通知電子郵件給相關者。期望本系統能應用於野外生態調查，降低人工觀察成本，協助農民或捕蜥獵人快速掌握蜥蜴出沒情形與數量變化，並為生態保育提供具體技術支持。

參、 研究設備及器材

一、軟體方面

程式語言	Python3.10
編譯器	PyCharm Professional 2021.3.3
影像標記	labelImg
影片編輯	Microsoft Clipchamp

Python 函式庫	OpenCV Ultralytics(8.3.240 版本) PyTorch(2.11.0+cu128 版本) pyttsx3 smtplib MIMEMultipart

二、硬體方面

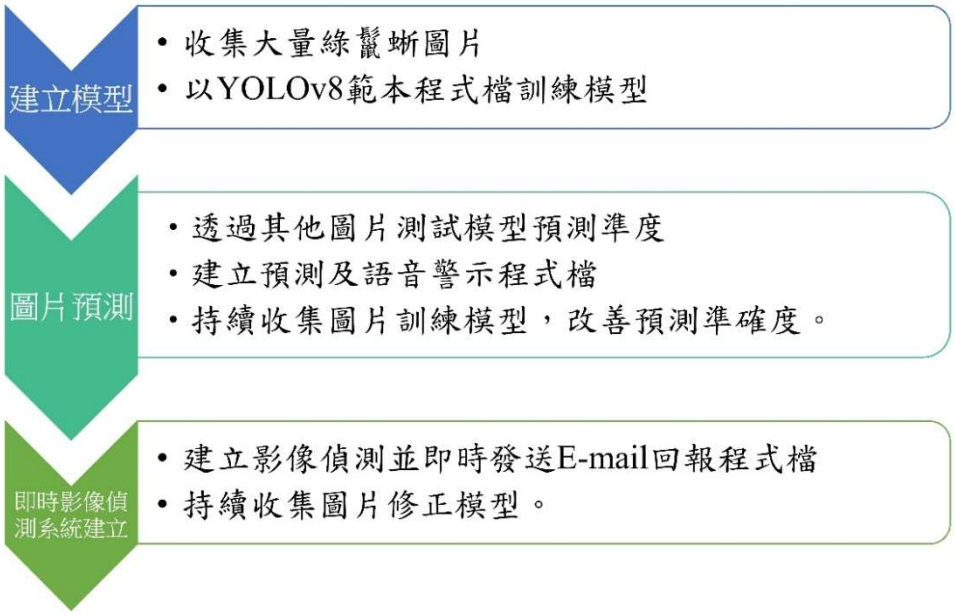
	
具備獨立顯示卡的筆記型電腦	Arducam 4K 8MP IMX219
	
4G LTE150Mbps 行動網路卡	Android 智慧型手機

三、筆記型電腦硬體規格：

CPU	Intel Core7 240H
顯示卡	NVIDIA Geforce RTX 5050 Laptop 8G
記憶體	16G
硬碟	SSD 500G

肆、 研究過程與結果

一、研究流程



二、圖片收集

綠鬣蜥的體型會隨著年紀而增長，紀錄甚至長達 2 公尺。尾巴以黑綠相間的顏色來辨別，繁殖期間雄性綠鬣蜥身體綠色的部分會呈現橘紅色。本研究的綠鬣蜥圖片拍攝處絕大多數來自於校園，以及鄰近本校的萬年溪周邊步道，少數圖片是取自網路新聞圖片、及臺南市綠鬣蜥通報平台臉書社群。

若為校園中拍攝者，則是本校老師們主動發現後，將檔案傳送給我們。此外我們也利用假日的中午期間，大約早上 11 時至下午 14 時是綠鬣蜥較容易出來曬太陽的時候，沿著萬年溪步道找尋綠鬣蜥的蹤跡。

屏東縣為農業大縣，其實身邊的親朋好友頻繁看見綠鬣蜥，也會主動提供給我們，因此圖片少部分來自是爺爺奶奶或是阿姨們在農田間或是大水溝旁發現拍攝的。我們將此類圖片或影像於後期模型較為成熟時檢測使用。

	
114.12.15 攝於本校大門口紅磚道	115.01.14 攝於萬年溪步道
	
114.10 攝於竹田鄉糶糴村	取自臺南市綠鬣蜥通報平台

三、模型訓練與圖片預測程式

將收集到約 200 張的圖片利用 LabelImg 軟體標記綠鬣蜥位置後，訓練的圖片數量及驗證的數量按照 2：1 分配，程式碼如圖 4。訓練出第一個模型版本 iguanaV1，模型訓練的準確度信心指數曲線以及 mAP50 走勢結果如圖 3。

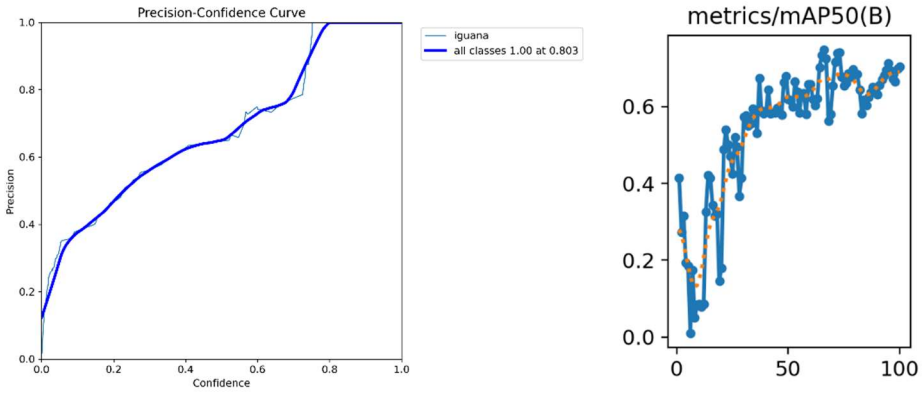


圖 3 iguanaV1 模型訓練的 Precision-Confidence Curve 和 mPA50 曲線圖

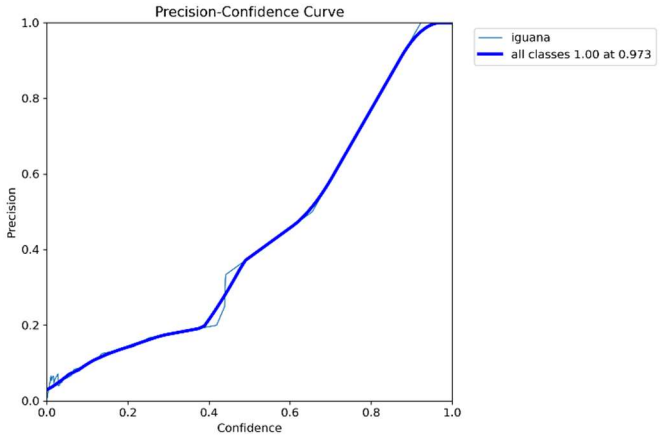
```python
import ultralytics
from ultralytics import YOLO
import multiprocessing

if __name__ == '__main__':
    multiprocessing.freeze_support()

    model = YOLO('models/iguanaV[illegible].pt')
    results = model.train(
        data="yaml/20251218.yaml",     #指定訓練任務檔
        imgsz=640,                     #輸入影像大小
        epochs=50,                    #訓練世代數
        patience=10,                   #等待世代數，無改善就提前結束訓練
        batch=10,                      #批次大小
        project='yolov8_iguanaV[illegible]',     #專案名稱
        name='exp1228'          )          #訓練實驗名稱
```

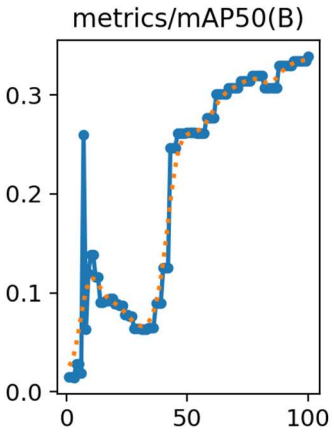
圖 4 模型訓練的 Python 程式碼

iguanaV1 的模型訓練的數據結果都不錯，接著開發出圖片預測及利用 Python 內建 pyttsx3 語音播放模組來讓電腦發出偵測播報訊息，程式碼如下：

```python
from ultralytics import YOLO
from PIL import Image
import cv2
import pyttsx3#語音撥放模組
def speakword(word):#定義語音播放啟動程式
    text = word
    action = pyttsx3.init()
    action.setProperty('rate',150)
    action.say(text)
    action.runAndWait()
model = YOLO("./iguanaV[illegible].pt")
im1 = Image.open("image2.jpg")
results = model.predict(source=im1, device='cuda')
for result in results:
    result.show()
    boxes = result.boxes
    if len(boxes) == 0:
        print("未偵測到iguana")
        speakword("未偵測到綠鬣蜥")
        continue
    class_name = model.names[0]
    print("-" * 20)
    print("偵測到共有",len(result.boxes),"隻",class_name)
    speakword("綠鬣蜥出現了出現了")
```

然而我們陸續使用了從沒訓練過的圖片讓 iguanaV1 模型預測，效果不盡理想，當背景較為複雜，例如增加了水泥地、女兒牆後，綠鬣蜥並沒有被檢測出來。因此我們持續收集了約 9 張綠鬣蜥圖片依舊按照 2：1 的比例訓練及驗證，用 iguanaV1 訓練後的最佳模型 best.pt 當訓練範本後得到 iguanaV2 的模型。其訓練指標如下圖：





iguanaV2 的 mAP50 值雖然整體是上升的曲線，但在前 50 世代數（epochs）訓練時有突然下降後來又上升。

將兩張未放到訓練圖片資料夾的圖像讓 igaunaV2 執行預測判斷，結果如下：

	
兩個框架出現誤判	未偵測到綠鬣蜥

我們參考 iguanaV2 的 YOLOv8 模型訓練後的混淆矩陣（Confusion Matrix），如圖 6，發現右上角（False Positive）數字為 5，代表著 valid 的 9 張圖片中有 5 張是誤認成綠鬣蜥，而矩陣中的左下角（False Negative）數字為 3，代表有 3 張有綠鬣蜥但是漏抓。由這兩個數據可知 iguanaV2 模型對新照片或是新的背景的綠鬣蜥檢測成效不佳。

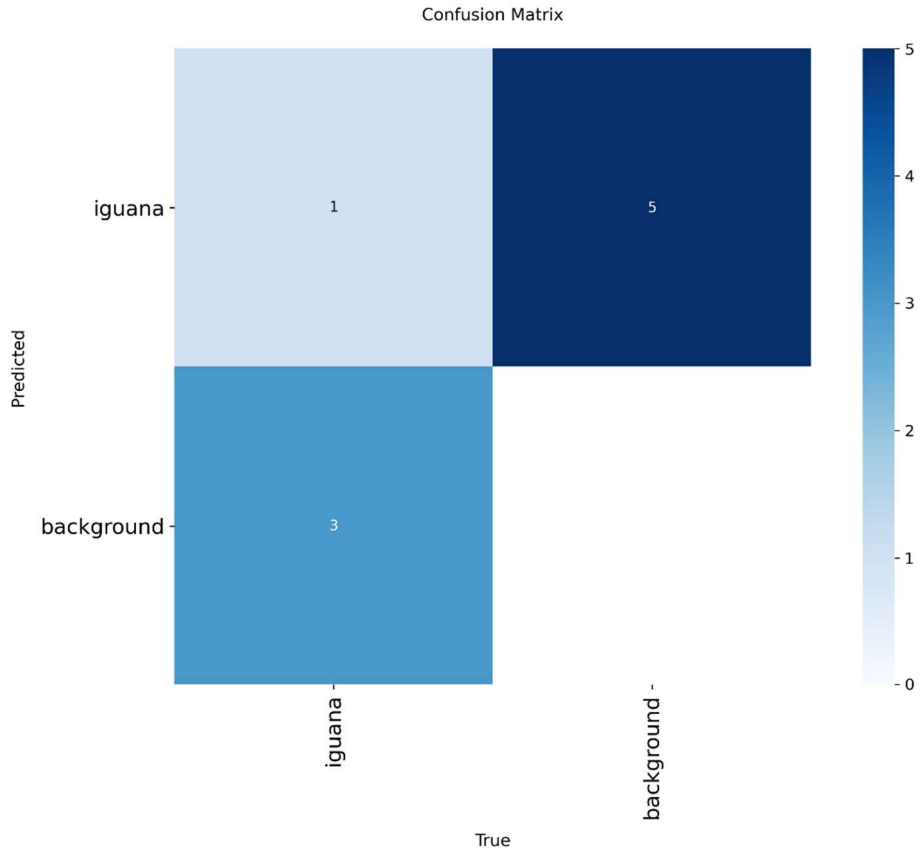


圖 6 iguanaV2 的混淆矩陣圖

出現上述可能原因為：樣本訓練圖片數量仍然太少，即使模型發展到 iguanaV2 的圖片訓練已達 250 張，以及沒有給予負樣本（即畫面中沒有綠鬣蜥）的訓練。而訓練資料太少，導致模型學習速度過快，我們決定減少訓練世代數至 50（V1 和 V2 的訓練世代數皆為 100）。

四、影像偵測及發送偵測結果至電子信箱程式

Python 內建的 smtplib 只可以發送 ASCII 編碼的郵件，考量模型可能誤判而發出錯誤的偵測訊息，我們需要在通知檢測結果的郵件中附帶偵測到的圖片檔案，MIME 模組可以在

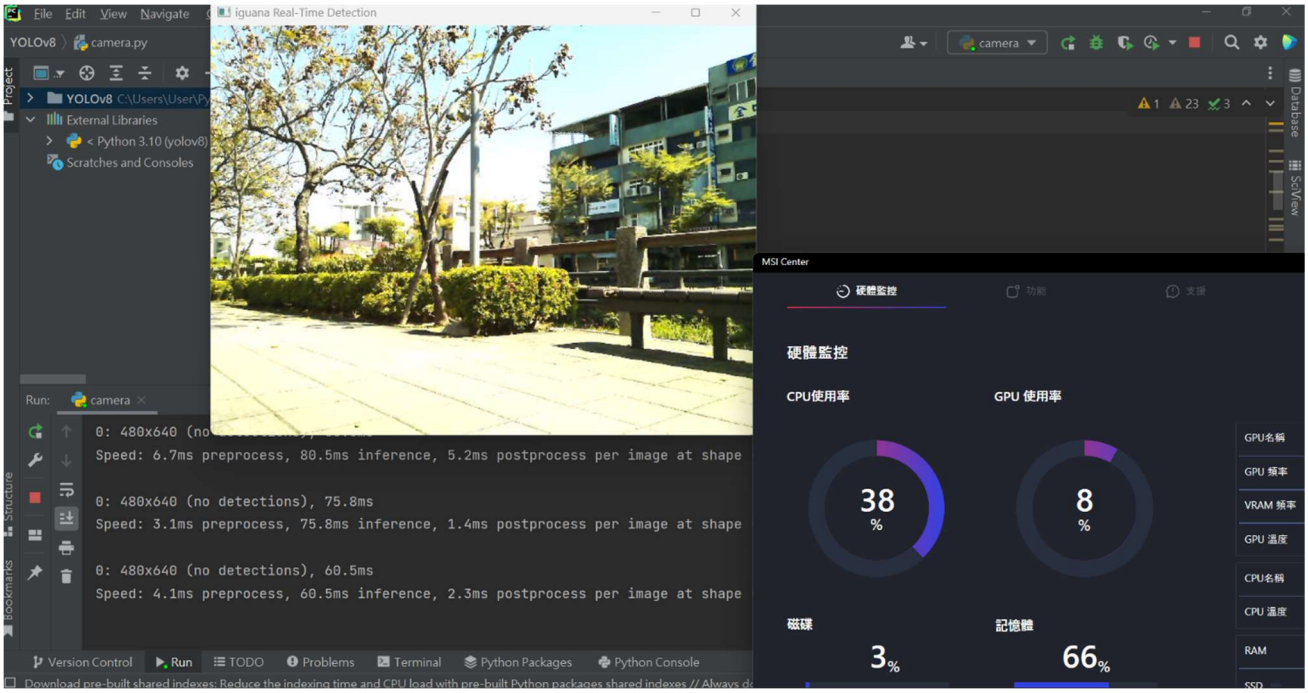
Python 程式中傳送圖片、音訊、視訊的電子郵件，本研究最後的檢測發送通知程式碼入下：

```python
from ultralytics import YOLO
from PIL import Image
import cv2
import pyttsx3#語音撥放模組
import smtplib
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
from email.mime.image import MIMEImage

def speakword(word):#定義語音播放啟動程式
    text = word
    action = pyttsx3.init()
    action.setProperty('rate',200)
    action.say(text)
    action.runAndWait()
model = YOLO("./iguanaV3.pt")
cap = cv2.VideoCapture(1)
if not cap.isOpened():
    print("Cannot open Camera")
    exit()
while True:
        print("Cannot receive frame")
        break
    results = model.predict(source=frame, conf=0.5)
    catch_frame = results[0].plot()
    cv2.imshow("iguana Real-Time Detection", catch_frame)

    if len(results[0].boxes.xyxy) > 0:#如果偵測到物件
        for result in results:
            result.show()#顯示偵測到的畫面
            result.save()#將此畫面儲存
            speakword("綠鬣蜥出現了出現了")

        with open("results_image0.jpg","rb") as file:
            filecontent = file.read()
        mime = MIMEMultipart()
        mime["Subject"] = "Detection Notify"
        mime["From"] = "[redacted]@gmail.com"
        mime["To"] = "[redacted]@gmail.com"
        mime["Cc"] = "[redacted]@gmail.com,[redacted]@gmail.com"

        mime.attach(MIMEText("找到了綠鬣蜥","plain"))
        image = MIMEImage(filecontent)
        image.add_header("Content-Disposition",'attachment', filename="results_image0.jpg")
```

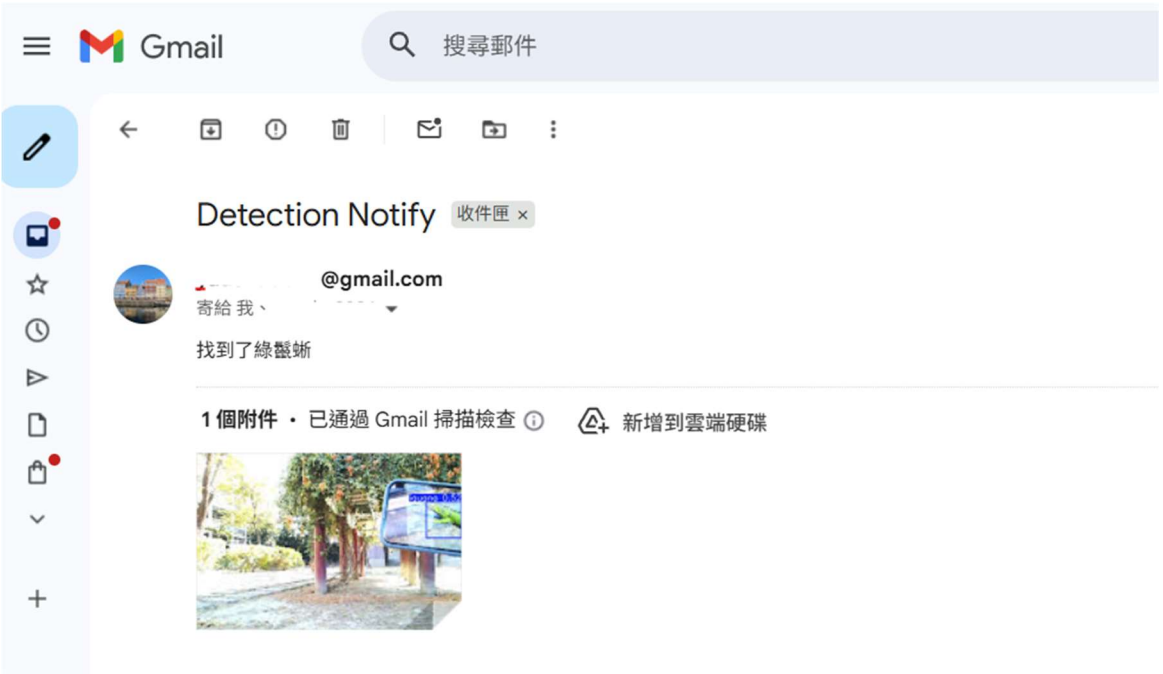
接下頁：

```
        image = MIMEImage(filecontent)
        image.add_header("Content-Disposition", 'attachment', filename="results_image0.jpg")
        image.add_header("Content-Type","application/octet-stream")
        mime.attach(image)
        msg = mime.as_string()
        smtp = smtplib.SMTP('smtp.gmail.com',587)
        smtp.ehlo()
        smtp.starttls()
        smtp.login('[redacted]@gmail.com','[redacted]')
        from_addr = '[redacted]@gmail.com'
        to_addr = '[redacted]@gmail.com'
        status = smtp.sendmail(from_addr, to_addr, msg)
        if status == {}:
            print("郵件傳送成功")
        else:
            print("郵件傳送失敗")
        smtp.quit()
        break
    if cv2.waitKey(1) & 0xff == ord("q"):#按q退出視訊畫面
        break
cap.release()
cv2.destroyAllWindows()
```

在執行偵測模型時，電腦啟動了 GPU 加速偵測，其 GPU 使用率在本研究中最大可達到 15%。但在正中午太陽底下執行環境使得 CPU 和 GPU 的工作溫度都超過了 50℃，架設後使用電腦執行程式的畫面如下：



由於若有偵測到物件需要發送電子郵件，考量器材放置地點多在戶外且不容易有無線網路，我們採用 4G LITE 無線網路卡，以便能在偵測到綠鬣蜥的同時能夠發出訊息。觀察者或是補蜥獵人便能同步接收訊息並前往所在地找尋綠鬣蜥。接收端得到的電子郵件如下圖：



五、增加負樣本（背景圖）訓練模型 iguanaV3

有鑑於 iguanaV2 模型太過敏感，正所謂的「看到黑影就開槍」而頻繁誤判綠鬣蜥，且之前兩個版本訓練的圖片缺乏負樣本（YOLO 官方稱為背景圖），加上我們持續在假日於萬年溪畔步道捕捉到大、小隻、或是斷尾的綠鬣蜥身影。以有綠鬣蜥的圖片和沒有綠鬣蜥的圖片數約為 3：1 的情況下，添加共 50 張的負樣本數量並按照比例分配至訓練和驗證的資料夾，再利用 iguanaV2 的模型檔，訓練出本研究目前的模型版本 V3。其訓練結果如圖 7；混淆矩陣如圖 8。

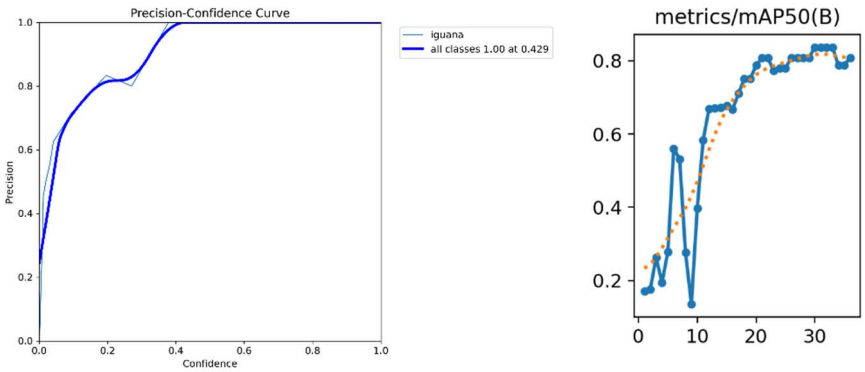


圖 7 iguanaV3 的 Precision-Confidence Curve 和 mAP50 趨勢圖

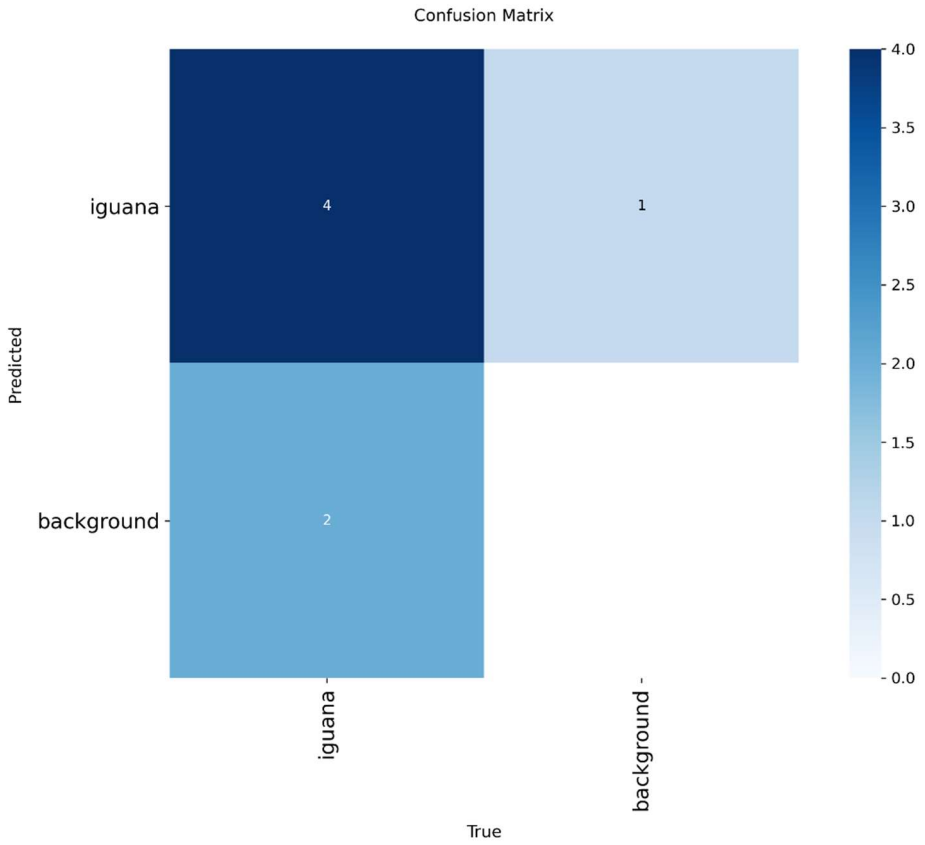


圖 8 iguanaV3 的混淆矩陣圖

六、iguanaV3 模型實際檢測

由圖 7 和圖 8 可知 iguanaV3 對於綠鬣蜥的誤判降低相當多，我們開始試著於校園各角落（如圖 a）以及萬年溪步道（如圖 b）測試 iguanaV3 模型的檢測性能，將各種測試場景整理成以下表格，圖 9 和圖 10、圖 11、圖 12 是在校園涼亭架設器材，攝影鏡頭為 IMX219，偵測模型執行約 10 分鐘期間，有老師經過及鳥類徘徊於畫面中模型未誤判，之後我們透過手機顯示各種綠鬣蜥照片（未加入模型訓練圖片集），包含只有綠鬣蜥頭部、綠鬣蜥半身等出現在畫面中，模型成功偵測。

圖 13、14 則是架設於學校一樓走廊，午休期間仍有不少老師、學生通過，模型亦未出現誤判，相同地在執行程式約 10 分鐘後，開啟斷尾的綠鬣蜥圖片（未加入模型訓練圖片集）緩慢地從鏡頭左側移動，模型成功偵測。圖 15、16 則是在萬年溪步道透過智慧型手機拍攝到的影片，改成讓程式執行檢測影片檔，此時模型出現誤判（如圖 16），雖有綠鬣蜥在畫面中，但模型的判斷框架卻是左下角的一堆落葉。

	
圖 a 偵測器材架設於校園涼亭	圖 b 偵測器材架設於萬年溪步道
	
圖 9 攝影鏡頭 IMX219	圖 10 攝影鏡頭 IMX219
	
圖 11 攝影鏡頭 IMX219	圖 12 攝影鏡頭 IMX219

<table>
<tr><td> 
</td><td> 
</td></tr>
<tr><td>圖 13 攝影鏡頭 IMX219</td><td>圖 14 攝影鏡頭 IMX219</td></tr>
<tr><td> 
</td><td> 
</td></tr>
<tr><td>圖 15 智慧型手機拍攝</td><td>圖 16 智慧型手機拍攝</td></tr>
<tr><td> 
</td><td> 
</td></tr>
<tr><td>圖 17 智慧型手機拍攝</td><td>圖 18 筆記型電腦內建鏡頭拍攝</td></tr>
</table>

圖 17 則是利用智慧型手機若透過 TypeC 傳輸線連接電腦也可以作為 Webcam，且具有三段變焦（2.0、1.0、0.5）的功能，但產生了誤判。圖 18 則是同學們掃地時間通知我們綠鬣

蜥在此處出沒，但已爬上其中一棵樹。我們還是選擇在此處架設器材後，執行程式偵測了 25 分鐘，最後將公車車頂和校園圍欄組成的顏色相間誤判為綠鬣蜥尾巴。

將學校老師拍到綠鬣蜥闖進辦公室的拍攝影片，因為手機拍攝的影片畫面大小對於 OpenCV 的 640x640 過大影響模型辨識，我們利用影片編輯軟體 Microsoft Clipchamp 調整大小並且解析度降為 720p 後，執行辨識結果如圖 19，判斷的信心指數（confidence）設定為 0.5，造成模型辨識錯誤。再調整 confidence 設為 0.6，如圖 20，模型再度把鐵窗框辨識成綠鬣蜥。

圖 21 和圖 22 是直接將影片畫面縮小但沒有調降解析度也沒有填滿畫面，模型皆沒有沒有辨識出任何物件，推測很可能是畫面在沒有填滿的情況下過多的黑色背景是之前訓練模型沒有學習過的場景。

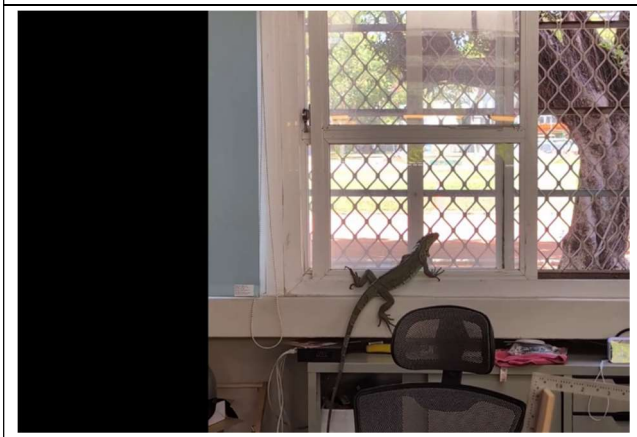
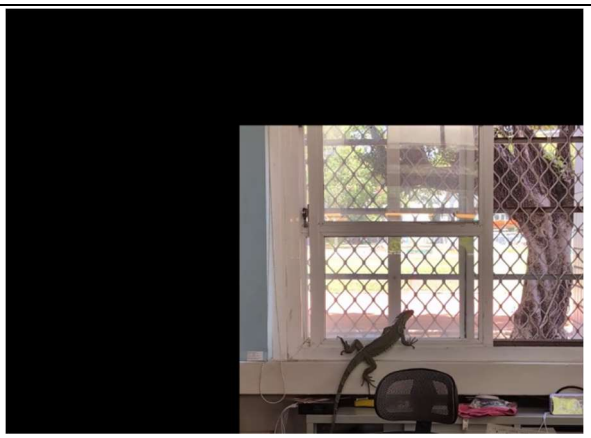
<table>
<tr><td> 
</td><td> 
</td></tr>
<tr><td>圖 19　720p 填滿 conf=0.5</td><td>圖 20　720p 填滿 conf=0.6</td></tr>
<tr><td>  </td><td>  </td></tr>
<tr><td>圖 21　畫面縮小沒有填滿</td><td>圖 22 畫面縮小沒有填滿</td></tr>
</table>

	
圖 21　1080p 填滿畫面 conf=0.5	圖 22　1080p 填滿畫面 conf=0.6

上方圖 21 和圖 22 則是將影片解析度調成 1080p 並且調整填滿畫面下，不論是判斷物件的 confidence 設定成 0.5 或是 0.6，模型再度將鐵窗框判定為綠鬣蜥。可以推測 iguanaV3 的模型對於在室內場景或是非戶外的背景，即使有綠鬣蜥也沒有判斷出來。

下方圖 23 是下課期間在走廊上同學發現的綠鬣蜥，當時緊急利用筆記型電腦的內建鏡頭開啟偵測，結果 iguanaV3 模型未辨識成功。圖 24 則是身形及尾巴顏色分布亦是相間的鬆獅蜥，iguanaV3 未誤認成綠鬣蜥。

	
圖 23　誤闖校園走廊的綠鬣蜥	圖 24 鄰居阿姨豢養的鬆獅蜥

伍、討論

一、模型訓練方面：

本研究初期一直給予模型正樣本的圖片訓練，在完全沒有負樣本的學習下，V1 和 V2 的

模型在靜態時連天花板的日光燈、冷氣室外機上面的棚架都會誤認為是綠鬣蜥，動態更不用說人來人往時每個人的臉龐都是綠鬣蜥。直到找尋相關資料後才了解到負樣本（或稱背景圖）對模型訓練也同樣相當重要。而訓練綠鬣蜥的圖片數量截至報告繳出大約 250 張，加上負樣本的 50 張，總和 300 張的數量對深度學習而言屬於極小樣本，因此訓練出來的模型還是不穩定。考量著作權我們還是決定自行收集綠鬣蜥的各種照片，假日只要有時間便會到萬年溪沿岸找尋。

二、偵測器材方面

本研究非使用雲端運行環境（例如 Google Colab）來訓練 YOLO 模型，而是採用具備入門獨立顯示卡（Nvidia GeForce GTX-5050）筆記型電腦運行深度學習的程式，其離線運作的方便性和時間運用上還是相對方便。然而研究中原使用的 2 種檢測攝影鏡頭：筆電內建鏡頭、USB 介面的 Webcam，在透過 OpenCV 開啟檢測畫面時，解析度不高亦無法變焦。之後我們加入了智慧型手機後當成是攝影鏡頭，可以變焦的話使模型更有機會檢測到綠鬣蜥。

三、器材架設環境方面

架設檢測設備主要有筆記型電腦、傳輸線以及鏡頭、還有無線網路卡，因綠鬣蜥會在中午出來曬太陽活動筋骨，除了架設在學校涼亭時並沒有讓筆電溫度飆升外，在萬年溪步道架設時，即使正值 12 月或 1 月，屏東正中午的氣溫還是會到 28 至 30 度，若遇太陽直曬則會使的筆記型電腦燙手，根據觀察執行模型辨識綠鬣蜥的程式 1 小時會電量下降約 50%。此外在大太陽環境下，也會造成拍攝畫面逆光導致畫面太暗而影響模型檢測效能。

陸、結論

深度學習的相關研究和應用在這幾年如雨後春筍般出現，Python 程式引領我們接觸 YOLOv8 在物件辨識的領域，若將本研究實際交由農民們或是捕獵的獵人們操作，在整個開啟辨識綠鬣蜥的模型程式介面以及架設器材過程還是不夠親民，還有筆記型電腦架設於戶外沒有源源不絕的電力來源供應。未來若有機會還想透過樹莓派（RaspberryPi）主機板來取代筆記型電腦執行 Python 程式，但模型檢測是否會因運算需求而減低辨識結果，是必須考慮的。不過用樹莓派主機板就可以增加模型辨識硬體的機動性，利用行動電源也可以克服電力

需求。

本研究實地架設偵測綠鬣蜥模型多次雖然未真正遇見綠鬣蜥，但若利用手機顯示綠鬣蜥圖片出現在鏡頭前，模型多數能辨識成功；然而誤判的情形還是相當可觀。相信若再蒐集更多的綠鬣蜥照片（總數希望達千張）加以訓練模型，一定能提高辨識準確率以及降低誤判率。

柒、參考文獻

Samuel（2025）。第 25 個冬天，**【YOLOv8】物件偵測與識別測試**。

https://coolmandiary.blogspot.com/2025/07/yolov8.html

Tommy Huang（2018）。**深度學習-物件偵測:You Only Look Once (YOLO)。**

https://chih-sheng-huang821.medium.com/

小狐狸事務所（2018）。**Python 學習筆記 ： 以 Gmail 寄送郵件的方法**。

https://yhhuang1966.blogspot.com/2018/10/python-gmail_17.html?m=1

劉亭妤、陳信安、顏吟竹（2025）。窩窩，**【綠鬣蜥入侵 20 年 02】移除量屢創新高，犧牲 24 萬綠鬣蜥為何還不見「控制」成效？。**

https://wuo-wuo.com/topics/widlife/20th-dragon/2236-iguanaintaiwan02

林佑謙、黃政嘉、李啟龍（2025）。**十字路口人車碰撞預測之研究**。全國中小學科展作品。

高正龍、邱俊瑋、彭冠綸、郭懿尹（2024）。**行人警示系統之開發研究。**全國中小學科展作品。

楊建華、李瑞峰（2024）。**台灣之光物件辨識最新 YOLO 原理精讀＋實戰。**台北市：深智數位股份有限公司。

張德豐（2022）。**一本書秒殺電腦視覺最新應用 80 個 Python 大師級實例**。台北市：深智數位股份有限公司。