中華民國第63屆宜蘭高級中學校內科學展覽會
作品說明書

高級中等學校組 電腦與資訊學科

科　　別：電腦與資訊學科

作品名稱：AI革新

組　　別：

作者：	指導老師：
高二 鄞永力	梁祐銘

關鍵詞：電腦視覺、虛擬人物、AI繪圖

摘要

人工智慧盛行在我們的時代，啟能悄悄從它身邊走過,偶然的接觸到人工智慧，竟然完全無招架之力的被那科技深深吸引，開啟短暫但我相信還有續集的探索之旅。

一、卷積神經網路(Convolutional Neural Network, CNN)：一種用於圖像辨識和分類的深度學習模型。MNIST 手寫數字辨識和 Cifar-10 圖片判斷都是基於卷積神經網路實現的。

二、YOLO 技術：這是一種物件檢測技術，可以實現快速而準確地檢測圖像中的物體。可製作判別貓狗位址和自動瞄準外掛都是基於YOLO實現的。

三、3D虛擬人物控制：MediaPipe 是 Google Research 開發的一個跨平台的機器學習框架，可以實現從相機、影像等輸入中擷取關鍵信息並進行處理。進一步使用 MediaPipe 模型操控3D虛擬人物。

四、神經風格轉換(Neural Style Transfer)：將一張圖像的風格轉移到另一張圖像上的技術。

五、生成對抗網路(Generative Adversarial Network, GAN)：這是一種利用兩個神經網路模型相互競爭來生成逼真的圖像的技術。GAN 的變體 CycleGAN 可以更加快速地實現風格轉換，GAN 這種技術在許多領域都有廣泛的應用。

壹、前言

一、研究動機

世代的進步和變革確實會帶來不少恐慌，但也是很多人成長和嶄新的契機。從農業時代到工業時代再到現在的人工智慧時代，每一個轉換都意味著某些企業和人的消亡，但同時也孕育著新的機會和崛起的企業和人才。現在正要邁向人工智慧的時代，如果想要跟上這波潮流，就必須加快步伐，不斷學習和創新，並且抓住機會，發揮自己的才華和創造力。我們正處在這樣一個轉捩點，對於我們每個人來說，這是一個非常激動人心的時刻，也是一個可以展現自己的舞台。

二、研究目的

(一) 研究人工智慧所擁有的有趣技術

(二) 辨識系統的應用

(三) 虛擬人物的控制

(四) AI繪圖原理

三、人工智慧 (Artificial Intelligence, AI)

1956年，人工智慧被確立為一門學科，半世紀間經過許多起起落落。如今電腦的運算能力約為 30 年前的 100 萬倍，且近10幾年大數據的快速發展，人工智慧重新活過來，許多先進的機器學習技術成功應用於社會中的許多問題。

人工智慧的定義可以分為兩部分，「人工」和「智慧」。「人工」即由人設計、創造，至於「智慧」還未有定論，我們還無法確定智慧的本質是什麼。自從發明電腦以來，人們就渴望創造出具有知覺、有自我意識的人工智慧，這方面就屬 強人工智慧(Strong AI)，至今市面上尚未有所重大進展。近年發展較多屬於 弱人工智慧(Weak AI)，看似擁有智慧，實際並無意識，也不理解執行的意義，只能處理特定目標，例如最近火紅的 ChatGPT、NovelAI皆是。

四、機器學習 (Machine Learning, ML)

而在AI底下有個分支，也就是這次的主題機器學習。機器學習是一門涵蓋電腦科學、統計學、機率論、博弈論等多門領域的學科，從 1980 開始蓬勃興起。機器學習之所以能興起，也歸功於硬體儲存成本下降、運算能力增強、大數據的發展。而機器學習中又有4類的學習方式。

(一) 監督學習 (Supervised Learning)

每筆資料都有對應的標籤(label)，讓電腦學習資料與標註的相關性。相對比其它種類精準，缺點需要人工自行標註，耗時耗力。

(二) 半監督學習 (Semi-Supervised Learning)

與監督學習目標差不多，差在只有少部分資料擁有正確標籤。最一般的策略是以尚有標籤得資料訓練出預測模型，再以此模型預測未標註資料。好處是不需要太多擁有標籤的資料，但預測模型可能不準。

(三) 無監督學習 (Unsupervised Learning)

數據集中只有未標註的資料，大多訓練方式是以 生成對抗網路 (Generative Adversarial Network, GAN)為主，簡單來說就是兩個(或多個)無知的人互相對抗，直到理解資料的相關性。

(四) 強化學習 (Reinforcement Learning)

以當下環境資料，算出下步動作，依情況給予獎勵或懲罰，最終以取得最大化的利益。就是讓電腦一直重複嘗試，其中給予正負評價，直到學會。

五、深度學習 (Deep Learning, DL)

深度學習又是機器學習的分支，且是其中成長最快的領域，其它還有 SVM、naive bayes、Decision、隨機森林、Decision Tree等電腦學習方式，但深度學習能自動提取資料特徵，其能力遠遠甩開其它演算法。深度學習其始祖為 人工神經網路(Artificial Neural Network, ANN)，後來經過種種事件，只要外層層數小於 3 的神經網路就稱 淺層神經網路(Shallow Neural Network, SNN)，反之稱為 深度神經網路(Deep Neural Network, DNN)，而深度神經網路也改名為我們所熟知的深度學習。

(一) 運作原理

神經網路架構靈感來源於人腦，人類的腦細胞，通常有500~1000億個神經元，以及超過100兆條突觸相連，形成一個複雜、高度互聯的網路，並相互傳送電訊號以協助人類思考。相同的，人工神經網路是由人工神經元組成，它們共同協作以解決問題。

一般人工神經網路架構分為 輸入層(input layer)、隱藏層(hidden layer)、輸出層(output layer)。輸入層是資料進入系統的入口，而隱藏層是處理資訊的地方，隱藏層從輸入層或其他隱藏層取得輸入。人工神經網路可以有大量的隱藏層。每個隱藏層分析前一層的輸出，進一步處理，並將其傳遞給下一層，重複直到輸出層，而最終的計算結果就會顯現在輸出層，也就是預測結果。

(二) 前向傳播法 (Forward Propagation)

前向傳播法是從神經網絡的輸入層開始，逐漸往輸出層進行前向傳播。上一層的神經元對本層的神經元輸入(X)後，與上一層神經元對應的 權重(Weight)加乘，再加入 偏差值(Bias)後，最後在通過一個非線性函數(激活函數)，便完成了該節點的輸出(Y)。

(三) 激勵函數 (Activation Function)

在神經網路中使用激勵函數，主要是利用非線性方程式，例如ReLU、Sigmoid、Tanh等函數，解決非線性問題，若不使用激勵函數，神經網路即是以線性的方式組合運算，因為隱藏層以及輸出層皆是將上層之結果輸入，並以線性組合計算，作為這一層的輸出，使得輸出與輸入只存在著線性關係，則神經網路所訓練出之模型效果不佳。

(四) 損失函數 (Loss Function)

每次訓練輸出層(預測值)與真實數據的標籤的相差，我們稱為誤差(error)。接著會使用損失函數將誤差整合，常見的函數有 均方誤差 (Mean

square error, MSE)、平均絕對值誤差 (Mean absolute error, MAE)、交叉熵 (Cross Entropy)。

(五) 優化器 (Optimizer)

優化器目的調整神經網路中的權重、偏差值，達到損失函數最小化。以 梯度下降法 (Gradient Descent) 快速找出最小損失函數的方法，利用微積分的偏微分、鏈鎖律計算，其優化器有BGD、SGD、MBGD、AdaGrad、RMSProp、Adam 等等非常多種。優化器還是有些問題的存在，例如找到局部最小值而非全域最小值、梯度消失問題 (Vanishing gradient problem)、梯度爆炸問題 (exploding gradient problem)。

(六) 反向傳播 (Back Propagation)

基本就是把損失函數、優化器結合。先由損失函數算出 損失(loss)，將損失帶入優化器進行計算再乘 學習率 (Learning Rate)，最後更新整個神經網路的權重、偏差值，以此達到學習目的。反向傳播讓模型一再的讀取資料，一次又一次的錯誤中做出修正，慢慢找出權重與偏差值的正確組合，讓預測結果能夠趨近我們的理想情況的方法。

(七) 其他

期 (Epoch)：將資料集全部訓練過一次。
批次 (Batch)：將資料集分批送入神經網路學習，每次完成更新一次權重。批次越多優化器計算越精準，相反的耗時耗記憶體。
迭代 (Iteration)：資料集/批次。
學習率 (Learning Rate) : 控制模型中梯度下降的速度。學習率太大可能跳過損失最小值，相反需要收斂時間更多。

六、神經網路種類

(一) 前饋神經網路 (Feedforward Neural Network, FNN)

是最古老的神經網路之一，最簡單的神經網路模型，資料經由輸入層通過隱藏層到輸出層，神經元之間沒有連接迴路存在。

(二) 卷積神經網路 (Convolutional Neural Network, CNN)

卷積神經網路通常用於圖片辨識，模仿人類大腦的認知方式，觀察由細微的事物到整體特色。卷積層 (Convolution Layer) 使權重的減少、池化層 (Pooling layer) 壓縮圖片，以此更高效率的判斷圖片。

1. **卷積層** (Convolution layer)

將輸入的圖像劃分為若干個矩形區域，對每個子區域以相同權重運算，最後加上激勵函數。神經元運算中無須每個輸入都要一個權重，我們稱 共享權值 (Shared weights)，**可大幅減少權重數量，藉此減少運算時間。**

2. **池化層** (Pooling layer)

一個壓縮圖片並保留重要資訊的方法，取圖片範圍內最高或平均當**做輸出，常用的有最大池化** (Max pooling) **與 平均池化** (Average pooling)。

3. **扁平層** (Flatten Layer)

將多維的輸入壓扁為一維輸出，常用在從卷積層到全連接層的過渡。

4. **全連接層** (Fully Connected layer)

連接最基本的神經網絡。

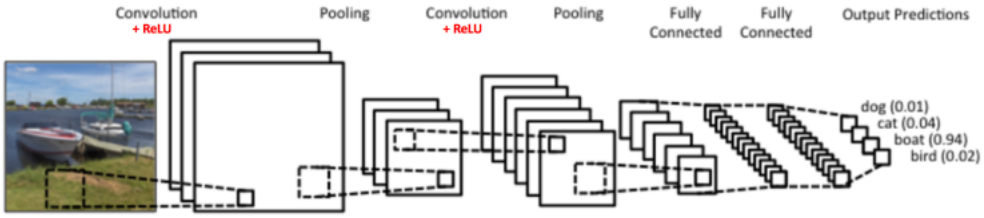

圖1 卷積神經網路架構 (圖片來源：https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/)

(三) **遞迴神經網路** (Recurrent Neural Network, RNN)

最常被用來處理時間和序列相關的問題。與使用前饋類神經網路不同的是，循環類神經網路具備前一層事件的「記憶」，並附加到目前層的輸出內容。缺點容易產生梯度消失、梯度爆炸、長期依賴 (Long-Term Dependencies) **等問題。**

(四) **長短期記憶網路** (Long term short term Memory, LSTM)

是進階的遞迴神經網路，解決許多問題。LSTM **會透過三個控制閥(輸入閥、遺忘閥、輸出閥)來決定將什麼資料保存(記憶)下來，而什麼記憶又**

該捨棄(遺忘)。看似不錯但也因為家入了許多內容導致參數變多，訓練難度提升了不少。

(五) 生成對抗網絡 (Generative Adversarial Network, GAN)

生成對抗網路是種非監督式學習，主要是兩個相互競爭的神經網路 生成網路 (Generative Network) 與 判別網路 (Discriminative Network)。生成網路生成圖片，目標騙過判別網路，判別網路判斷是否與資料相同，目標提升鑑定水準，這樣一來一回的對抗促使兩邊互相成長。

七、YOLOv8

YOLO (You Only Look Once) 是一種物件偵測方法，目前共推出8個版本。YOLO 的主要優勢是其快速的運算速度，能夠及時處理圖像。YOLOv8剛好在2023登陸，既然是最新版本，運算成本應該較低，因此選用 YOLOv8。

(一) one stage vs two stage

one stage：簡單講就是先將圖片拆成許多小塊，再預測修正物件位址、大小，全部只需要一個模型。較有名的偵測的方法：YOLO系列、SSD。

two stage：先找出圖片中物件，比如 Selective Search，再用卷積神經網路辨識。方法很直觀，但如果有大量物件，卷積神經網路就需辨識多次，時間上也就不允許及時運算了，相對精準。較有名的偵測的方法：R-CNN、fast R-CNN、faster R-CNN。

(二) 模型架構

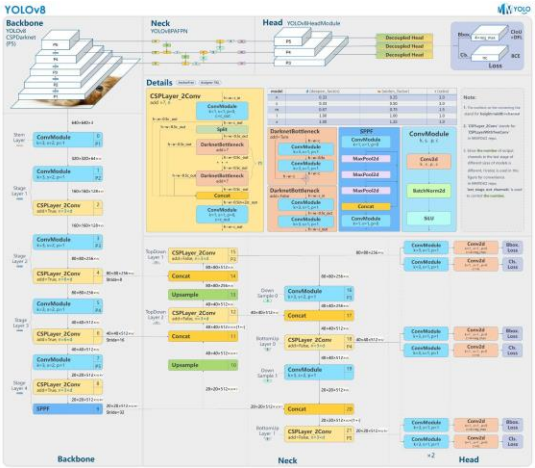

圖2 YOLOv8 架構 (圖片來源：https://github.com/ultralytics/ultralytics/issues/189)

(三) 模型種類

YOLOv8n、YOLOv8s、YOLOv8m、YOLOv8l、YOLOv8x。YOLOv8n 是最快和最小的，而 YOLOv8x 是其中最準確但最慢的。

貳、研究設備及器材

一、硬體設備

(一) 桌上型電腦

作業系統：Windows 10

CPU：Intel Core i7-12700K

GPU：NVIDIA GeForce RTX 3060

記憶體：32 GB

(二) Logitech C310 HD 網路攝影機

二、軟體工具

(一) Python 3.9：**程式語言**

(二) C#：**程式語言**

(三) Unity：**遊戲引擎**

(四) CSGO：**射擊遊戲(測試用)**

(五) Anaconda：**虛擬環境**

(六) Kaggle：**數據建模和數據分析競賽平台**

(七) Roboflow：**線上圖片標註**

三、Python 模組

tensorflow、keras、numpy、opencv、matplotlib、PIL、os、ultralytics、YOLO、roboflow、vgg19、KalmanFilter、glob、time、math、mouse、keyboard、simple_image_download、sklearn、pylab、ctypes、win32api、pygetwindow、pyautogui、pydirectinpu、socket、gdown、zipfile、pandas、seaborn、functools、enum

參、 研究過程及結果

一、機器辨識 - 天眼在我家

大家想到大數據總是又愛又恨，愛的是他的功能強大，只要有足夠的數據，能在幾毫秒得到所需答案，比過去數萬人、同時作業數年的效果更快更精準，相對的也可能涉及到隱私問題。機器視覺中的人臉辨識已是最基礎的，各種生物、物品的辨識所造成的人力精簡及時間的省節，都可藉由深度學習來達成，且準確度更不會有人為的疏失！

(一) MNIST 辨識

MNIST 是一種黑白的手寫數字資料集，分類為 10 類 (0~9)，在踏入CNN這領域時，大多數人都會使用這資料集當作練習，可以將解決 MNIST 視為深度學習的 "Hello World!"。

1. 資料集

包含了 60000 張訓練與 10000 張測試圖片，每張大小為 28*28*1 (28*28是圖片像素，1則是因為黑白圖片 1層)。

2. 模型製作

製作模型時可以依照前饋神經網路的做法，全用最基礎的密集層(dense)連結。假設只有一層100個神經元的密集層，輸入特徵有 28*28*1 個，權重數量 = 28*28*100(權重) + 100(偏差值) = 78500。反觀看卷積神經網路的卷積層，假設用100個3*3的卷積神經元，因為權重共享的關係，權重數量 = 3*3*100(權重) + 100(偏差值) = 1000。

這兩種作法比較權重差了將近80倍，而且是在只有一層神經網路、圖片大小只有 28*28*1 的狀況下，運算速度可想而知也差別巨大，前饋神經網路這做法也可能造成過度適配等問題。這就是為什麼要使用CNN技術。

3. 實際執行

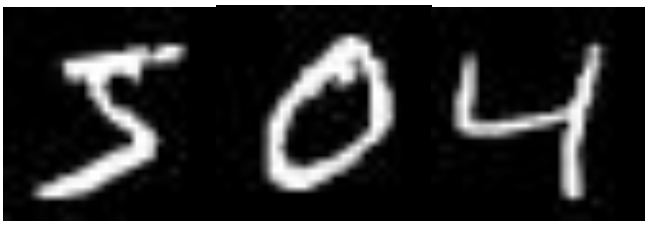

圖3 偵測為5、0、4

4. 結論

第一次辨識成功後，我非常驚訝只用數學運算就可辨識物件，難怪現在深度學習如此有人氣，想不到未來這技術能有多強大，真的很佩服想到深度學習的人。

(二) Cifar-10 辨識

MNIST 手寫數字辨識完成後，換一個相較難一點點的資料集 Cifar-10，資料集內含 10 種類別的圖片，分別是飛機、汽車、鳥、貓、鹿、狗、青蛙、馬、船、卡車，且屬於彩色圖片。

1. 資料集

這資料集有 50000 張訓練用以及 10000 張測圖片，圖片放大到 32*32*3 (彩色圖片有紅、綠、藍 3層)。

2. 資料增強 (Data Augmentation)

不過只是圖片大小改變，修一下手寫數字模型就好了，但如果想追求更高正確率的話、不產生 過度擬合 (over-fitting)、彌補資料量不足，可以用資料增強這技術。簡單講就是稍微改變資料，讓資料多樣化。改變的方法有很多，像是上下左右移動、放大縮小、偏移、旋轉、飽和度·等等非常多種。不過要記得符合資料可能的狀況，像是如果要訓練判別貓狗，資料增強時不會吧貓狗圖片完全顛倒，因為現實很少出現這種狀況。

3. 實際執行

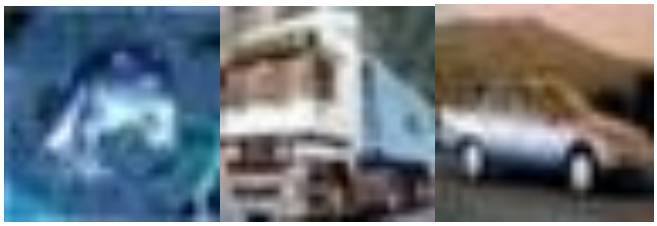

圖4 偵測為青蛙、卡車、汽車

(三) 貓狗偵測

試驗 YOLOv8 技術，採用 yolov8l-seg 訓練模型，較講求精準。

1. 圖片爬蟲

使用 simple_image_download 模組，可以自動在 google images 查找相關圖片並下載。容易尋找圖片，但有許多重複又或者無法使用的圖片，需自行刪除修改。

2. Roboflow

物件偵測的資料集網站，可以自製，也有別人現成的資料。透過爬蟲取得圖片後，將圖片於 Roboflow 中標記物件位址，可使用方框或分割，這非常耗時，貓與狗分別各60張，花費將近3小時(使用分割)。

3. 訓練結果

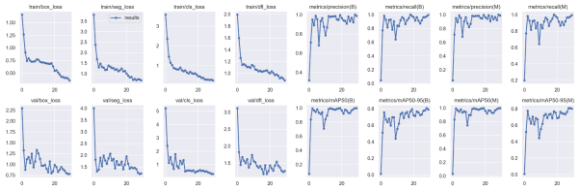

圖5 訓練過程

4. 實際執行

圖6 貓狗

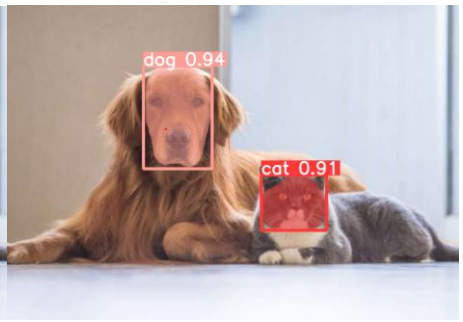

圖7 貓狗偵測

5. 結論

製作資料集時手真的差點抽筋，非常痛苦的三小時，其餘都相對簡單。意外的是，只需120張圖片就可做出正確率如此高的辨識模型，讚嘆YOLO。

(四) CSGO 自動瞄準

在逛 Roboflow 網站時，意外找到 CSGO(射擊遊戲) 人物偵測的資料集，馬上意識到這資料集是為了外掛而生。使用 YOLO 預測人物位置，再移動鼠標至人物，加點隨機數，這不就是個無法防範的外掛(一般外掛以破解遊戲內部資料為主)。

1. 資料集

Robotflow 中找到名為"csgo models"以方框標記的資料集。

2. 鼠標移動

python 中有許多可控制滑鼠的模組，例如：mouse、ctypes、win32api、pyautogui、pydirectinput...等，但多數已無法運用在遊戲中(遊戲有防)，pydirectinput 實測後唯一可在 CSGO 中使用。

3. 訓練結果

使用 yolov8n 訓練模型，希望以最快速度找到人物。圖8為訓練過程，並不是太成功，猜測其原因為 CSGO人物與地圖顏色過於相近、資料集不太完善、訓練不足。

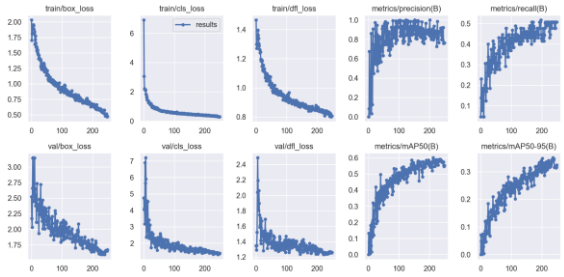

圖8 訓練過程

4. 實際執行

準備就緒後，開一場電腦場(只有我一個玩家)，不出意外的話就要出意外了。偵測不給力、時間差、容易偵測到隊友、鼠標移動，種種問題浮現，主要還是偵測太慢且不准。輸入到偵測完成，敵人早以移動到下個位置了，如果能一直偵測到，這問題就可以用預測的方式解決，但這偵測模型只要距離稍微遠一些，就完全偵測不到了。

5. 結論

可說是完全的失敗，不過我相信只要優化過偵測模型，還是能成為很好的外掛。但為什麼這種外掛模式不怎麼聽過，很明顯效益不高，與傳統外掛相比，傳統外掛直接竊取內部資料簡單、快速、準確、更強。相反這種外掛還需要看到敵人才能有所反應，各項技能皆差距傳統外掛一大截，唯一好處是難被抓包，不過遊戲端應該也可反用AI抓這種外掛。

二、3D虛擬人物 - 玩弄虛擬人物於股掌之間

藉由程式設計，我創造出自己的虛擬人物，這個虛擬人物就可以隨我控制，一個人體結構，就可以有40個控制點，著實令人興奮不已，過去在 虛擬實境 (Virtual Reality, VR) 的遊戲裡，控制虛擬人物大多都以VR穿戴裝置實現，總是需要手把或其他工具來操作這個虛擬人物。使用 MediaPipe 後，發現是否只用一台攝影機，我個人的任何動作都可控制這個虛擬人物，不需任何的手把或工具。並且我相信仍有更多使用的空間。

(一) MediaPipe

MediaPipe 是 Google Research 發表的開源專案，可支援多種語言，擁有許多辨識功能，這次實驗主要使用 臉部網路 (Face Mesh)、人體偵測 (Pose)、手部偵測 (Hands)。

這些偵測模型可抓出身體各部位，只使用一個鏡頭，並且輸出三維位置。人體偵測原理是訓練時以三維當標籤，臉部網路則是偵測幾個點後再將3D圖形套上。

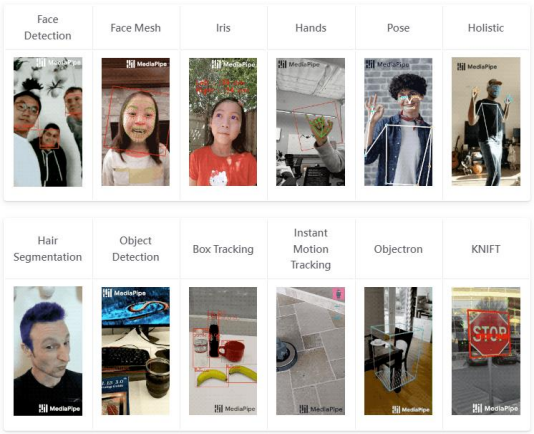

圖9 MediaPipe專案類別 (圖片來源：https://google.github.io/mediapipe/)

(二) 人體動作偵測

只能控制虛擬人物太無聊了，因此設計動作偵測，創造互動式小遊戲。希望做出特定動作，角色就可發射子彈，攻擊目標。

1. 資料集

既然是動作，網路上就相對少有資料集，因此需自製。自製方法也相當簡單，也相當耗時耗力，使用 MediaPipe 偵測點位，而動作是需要時間完成的，所以需要在相同時間內完成動作，並將時間內所有偵測到的點當作資料集。

收集的資料集為，發射動作與無動作。

圖10 發射資料

圖11 無動作資料

2. 模型製作

剛好動作與時間序相關，正好可使用長短期記憶網路，可記憶以前重點事件並輸出給下個神經元，最後以密集層連接。

3. 實際執行

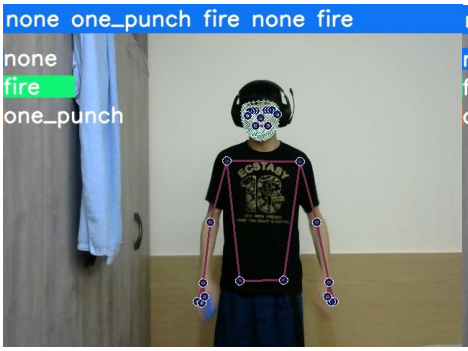

圖12 發射偵測

圖13 無動作偵測

(三) 歐拉角 (Euler angles)

物體在三維空間旋轉的方法，三個旋轉軸分別為翻滾(Roll)、俯仰(Pitch)、偏擺(Yaw)。運用臉部網路偵測點，推算頭部翻滾、俯仰、偏擺。

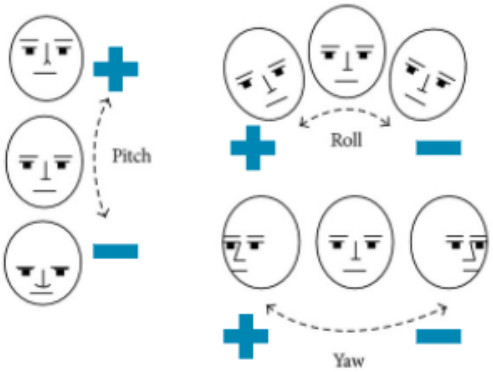

圖14 翻滾、俯仰、偏擺示意圖 (圖片來源：https://github.com/jerryhouuu/Face-Yaw-Roll-Pitch-from-Pose-Estimation-using-OpenCV)

(四) 卡爾曼濾波器 (Kalman Filter)

是一種高效率的遞歸濾波器，能夠從包含雜訊的測量中，排除雜訊。MediaPipe 偵測中很難完全無雜訊，卡爾曼濾波器這時就可很好的發揮其作用。

(五) 傳輸控制協定 (Transmission Control Protocol, TCP)

將 MediaPipe 偵測完人體位置後，Python 傳送至 Unity 中的所需工具。傳輸控制協定會在兩個端點間建立連線確保雙方的溝通順暢，其中要求位置(IP)、連接埠(Port)。

(六) Unity

最初不知要使用何種方式呈現虛擬人物，一度嘗試用 Python 建3D模型，但難以執行，後來發現 Unity，簡直與我的需求完全符合。Unity 為2D和3D的遊戲引擎，語言為 C#(完全沒碰過 全部重頭學起)。

(七) 3D虛擬人物模型

大部分虛擬人物皆需要錢，Unity 官方有免費釋出一個人物模型 Unity-Chan，有免費肯定用啊。

圖15 Unity-Chan

(八) 物理骨 (PhysBones)

由 VRChat 開發，在 Unity 中模擬頭髮、衣物、配件物理飄動功能。

(九) 製作過程

版本1：先在 Python 中使用 MediaPipe 人體偵測，再用傳輸控制協定將偵測資料傳給 Unity，使每個傳輸資料控制一個小方塊，完成後就可簡單看出人體架構了。

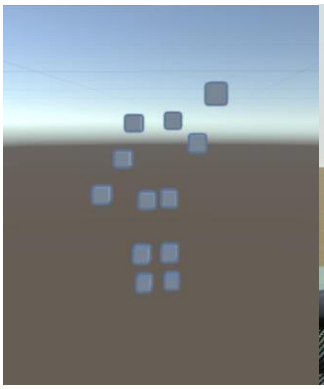

圖16 Unity影像

圖17 現實影像

版本2：將 Unity-Chan 人物模型套入，但3維位置是無法控制角色關節活動的，Unity 中有個指令可使一個3維位置指向另一個，藉此完成角色控制，最後推算並套用頭部翻滾、俯仰、偏擺。實測發現全身有嚴重震動，卡爾曼濾波器加入後優化許多，但移動速度就相對慢一拍。


圖18 Unity影像　　圖19 現實影像

版本3：只有控制角色就稍微無趣些，因此加入第一人稱視角的小型射擊遊戲，遠處放上些許目標物，偵測人體動作判斷是否射擊，藉此擊倒目標物。手指可以表達許多事物，因此也將手指偵測位置套入，但手指相對精細，距離稍遠可能偵測不完全。


圖20 Unity影像　　圖21 現實影像

(九) 實際執行

第一人稱射擊小遊戲展示。

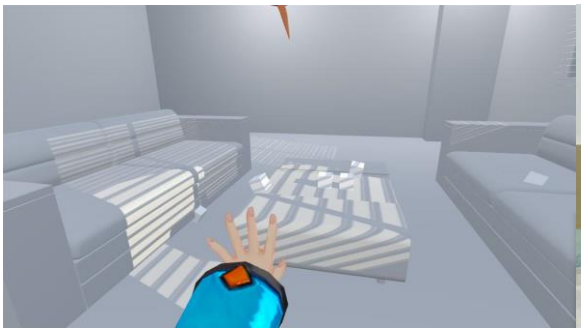


圖22 Unity影像　　圖23 現實影像

(十) 結論

真的很訝異我能夠做出控制虛擬人物程式，以前以為遙不可及的，現在卻在我手中。過程中訪查了上百上千個網站來學習，為了達到這個功能，為此還特別學了一種程式語言。雖然這技術還有需多可改進的地方，像是偵測的準確度，可以使用雙攝影機加上自己設計的模型，也許就可判斷更加準確。

三、藝術 - 很抱歉，AI 讓藝術家失業了！

透過AI的協助更可擴大藝術家、非藝術家的創意發想，只要想得到的，不只都能創作出來，且可發揮比想到的更大更精準的效果，並引領創作者達到另一個層次與想像！換句話說，也不再需要藝術家了，人人都可以透過此設計就比藝術家更厲害了。

(一) 神經風格轉換 (Neural Style Transfer, NST)

過去幾個月流行將現實圖片轉換成動漫風，而我也被這技術驚豔到，因此發現神經風格轉換。

1. 原理

採用判別模型來對原圖及風格圖提取特徵，生成一張新的圖片同時擁有原圖的內容以及風格圖的風格。

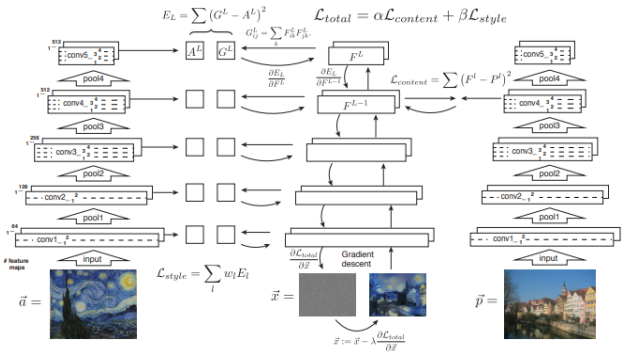

圖24 神經風格轉換 (圖片來源：https://rn-unison.github.io/articulos/style_transfer.pdf)

2. VGG-19

為卷積神經網路判別模型(16個卷積層及3個全連接層)，以預先訓練過判斷1000種類別，但我們並不需要能夠辨識這1000種類別的能力，以原先的卷積層當初始值，加以訓練。

3. 實際執行

選擇本人照片當內容圖，而使用 **梵谷《星夜》**(The Starry Night)、**康丁斯基《第七號構圖》**(Composition VII)、**葛飾北齋《神奈川沖浪裏》**(The Great Wave off Kanagawa) **為風格圖。**

圖25 風格轉換

4. **結論**

神經風格轉換為神經網路影像風格轉換的開山始祖，但每次要生成一張新的風格化影像需要重新訓練一次 VGG-19，**效率頗低，且完全不能即時轉換。**

(二) DCGAN

發現神經風格轉換無法實現即時轉換後，嘗試找出何種方式可即時轉換。雖然當時沒找到，但找到生成對抗網絡這有趣技術，只要給無標註的圖片，就可隨機生成出同類別圖片。

1. **原理**

由生成網路與判別網路互相比較，直到雙方完成學習。先隨機產生**亂數給生成網路生成假圖片，再由判**別網路判斷越接近實際越好，但判別網路不會平白無故知道他要判別的事物，所以也要給予實際圖片訓練**判**別網路。完成訓練後，生成網路就成為模仿大師，仿冒出人也無法判**別的圖片。**

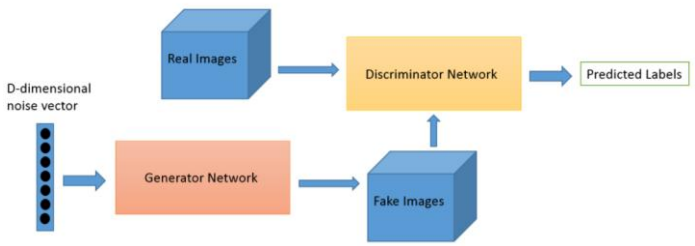

圖26 GAN 原理 (圖片來源：https://github.com/jonbruner/generative-adversarial-networks/blob/master/gan-notebook.ipynb)

2. 反卷積網路 (Deconvolution Network)

正好與卷積網路完全相反，卷積網路目的是想盡量保留特徵的壓縮，反卷積網路則是嘗試完整解壓縮，反捲積 (Deconvolution)、上採樣 (UNSampling)、上池化 (UnPooling)。

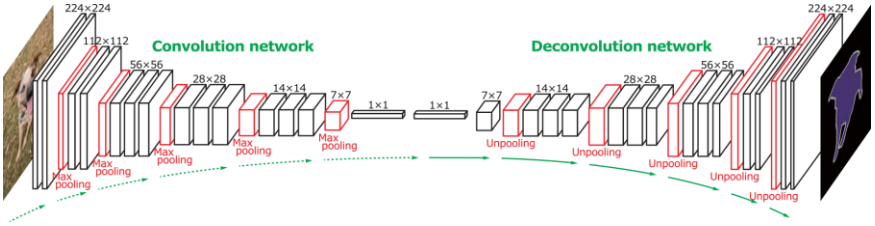

圖27 卷積網路與反卷積網路 (圖片來源：https://www.cv-foundation.org/openaccess/content_iccv_2015/papers/Noh_Learning_Deconvolution_Network_ICCV_2015_paper.pdf)

3. 資料集

Kaggle 中找到的資料集，其中包含63632張動畫人臉。

4. 實際執行

共做兩個版本，第一版成效並不好，第二版模型、損失函式大改後好了許多，甚至有眼鏡的出現。可以看出是張人臉，但要成功騙過人，這樣的成果還差強人意。

圖28 版本一 大小64*64

圖29 版本二 大小64*64

圖30 版本一 大小256*256

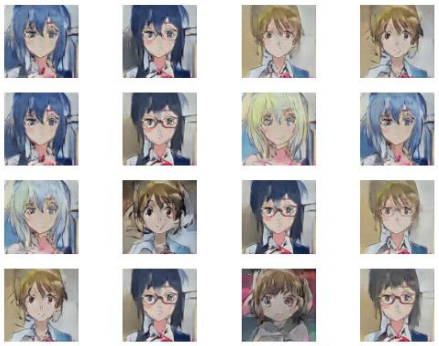

圖31 版本二 大小256*256

5. 結論

只能說待加強，感覺模型還可再增強、學習率也可再調，生成對抗網絡是由兩個模型互相比較而來，兩個模型要一起成長、進步，不能其中一個的能力強過多於另一個，因此微調其中參數也是很重要的一環。

(三) CycleGAN

生成對抗網絡有非常多變體，多到以英文字頭+GAN取名的變體都快佔滿了，當中我無意看中 CycleGAN，希望藉此達成即時轉換功能。

1. 原理

類似於兩對 GAN 相互圖片轉換，原理可參考以下圖。

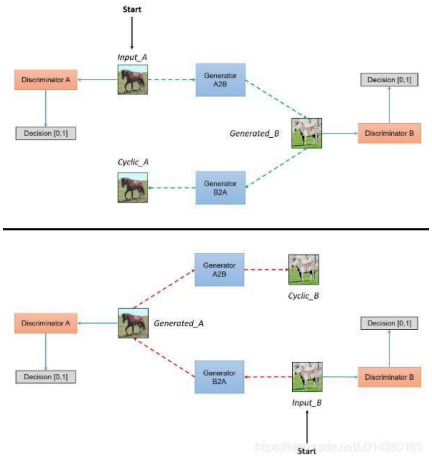

圖32 CycleGAN 原理 (圖片來源：https://hardikbansal.github.io/CycleGANBlog/)

2. 資料集

採用 Kaggle 中莫內的畫作300張(風格圖)，以及現實風景7038張(內容圖)。

3. 實際執行

圖33 現實轉莫內圖片

4. 結論

GAN 都如此難訓練了，更別說 CycleGAN 了。原理是很有趣、巧妙，但產生的圖形缺乏多樣化，訓練時損失震盪巨大，成效差強人意，並不是太有效的風格轉換方式。

肆、結論

用數學運算功能，加上程式語言，就能達到很多意想不到的答案。原來程式設計只是邏輯的運用，說穿了，每件事都可以邏輯的運用及分析。程式語言只是工具，它的功能強大，在於如何去運用它、活化它、創作它。真是令人深深著迷。

一、機器辨識

透過最基礎的卷積神經網路技術的測試以及進一步的應用，可以發現其在辨識與定位方面的應用潛力非常大。除了手寫數字辨識、貓狗辨識等基礎應用外，還可以應用在許多重要的場景中，例如自駕車、軍事、失踪人口協尋等等。然而，應用卷積神經網路技術時也必須注意遵守相關法律法規以及道德規範，確保技術應用的合法性與合理性。總體而言，卷積神經網路技術的應用前景廣闊，將對未來的生活、工作、社會等方面產生深遠的影響。

二、3D虛擬人物

這種技術的應用對多個領域帶來了實際的改進。從VR到元宇宙，這種精確控制的程式為人們提供了更方便、更簡單、更直觀的虛擬體驗。隨著這些技術的進一步發展和普及，可以期待看到更多的人從中受益。

三、藝術

使用 GAN 技術進行影像風格轉換是一種很有前景的應用，它可以讓我們將一張圖片的風格轉換成另外一種風格，例如將印象派畫作的風格應用到一張照片中。儘管成果目前並不十分樂觀，但這個技術仍有很大的發展空間，特別是隨著AI技術的不斷進步，未來我們可能能夠更好地掌握影像的細節，進一步提高影像風格轉換的效果。

這項技術對美化世界有很大的潛力，它可以幫助藝術家以更快、更有效的方式創作出令人讚嘆的作品，同時也可以創造出更多具有價值的商品。除此之外，影像風格轉換還可以應用在影視、遊戲等領域，讓觀眾享受到更多美感和豐富的視覺體驗。總之，這項技術有很大的潛力，值得我們繼續關注和發展。

伍、參考文獻資料

[1] Tommy Huang（2018年9月27日）。機器/深度學習: 基礎介紹-損失函數(loss function)。取自：https://chih-sheng-huang821.medium.com/%E6%A9%9F%E5%99%A8-%E6%B7%B1%E5%BA%A6%E5%AD%B8%E7%BF%92-%E5%9F%BA%E7%A4%8E%E4%BB%8B%E7%B4%B9-%E6%90%8D%E5%A4%B1%E5%87%BD%E6%95%B8-loss-function-2dcac5ebb6cb

[2] GGWithRabitLIFE（2018年8月5日）[機器學習ML NOTE]SGD, Momentum, AdaGrad, Adam Optimizer。取自：https://medium.com/%E9%9B%9E%E9%9B%9E%E8%88%87%E5%85%94%E5%85%94%E7%9A%84%E5%B7%A5%E7%A8%8B%E4%B8%96%E7%95%8C/%E6%A9%9F%E5%99%A8%E5%AD%B8%E7%BF%92ml-note-sgd-momentum-adagrad-adam-optimizer-f20568c968db

[3] WenWei Kang（2019年3月14日）。Back-propagation。取自：https://medium.com/ai-academy-taiwan/back-propagation-3946e8ed8c55

[4] Sumit Saha (2018,Dec 16). A Comprehensive Guide to Convolutional Neural Networks－the ELI5 way. Retrieved from https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

[5] 甜不辣馬拉松（2020年12月9日）。[Tesorflow Keras 學習筆記]新手一定要玩的MNIST手寫數字辨識。取自：https://sweetornotspicymarathon.medium.com/tesorflow-keras-%E5%AD%B8%E7%BF%92%E7%AD%86%E8%A8%98-%E6%96%B0%E6%89%8B%E4%B8%80%E5%AE%9A%E8%A6%81%E7%8E%A9%E7%9A

%84mnist%E6%89%8B%E5%AF%AB%E6%95%B8%E5%AD%97%E8%BE%A8%E8%AD%98-9327366cc838

[6] Sovit Rath (2023,Jan 10). YOLOv8 Ultralytics: State-of-the-Art YOLO Models. Retrieved from https://learnopencv.com/ultralytics-yolov8/

[7] oxxo（2022年9月25日）。(Day 14) 使用 MediaPipe。取自：https://ithelp.ithome.com.tw/articles/10297967

[8] 湯沂達（2021年3月6日）。類神經影像藝術風格轉換系列筆記－基礎。取自：https://medium.com/ai-academy-taiwan/%E9%A1%9E%E7%A5%9E%E7%B6%93%E5%BD%B1%E5%83%8F%E8%97%9D%E8%A1%93%E9%A2%A8%E6%A0%BC%E8%BD%89%E6%8F%9B%E7%B3%BB%E5%88%97%E7%AD%86%E8%A8%98-%E5%9F%BA%E7%A4%8E-e5a02d07c5c0

[9] Leon A. Gatys, Alexander S. Ecker, Matthias Bethge (2016). Image Style Transfer Using Convolutional Neural Networks. Retrieved from https://rn-unison.github.io/articulos/style_transfer.pdf

[10] mikechenx（2018年1月7日）。Day 28：小學生談『生成對抗網路』(Generative Adversarial Network，GAN）。取自：https://ithelp.ithome.com.tw/articles/10196257

[11] Jun-Yan Zhu, Taesung Park, Phillip Isola, Alexei A. Efros (2023,Jan 10). Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. Retrieved from https://junyanz.github.io/CycleGAN/