第二十一屆旺宏科學獎

成果報告書

參賽編號：SA21-080

作品名稱：運用影像辨識及機器學習改良打字系統與密碼

姓名：鄭兆宏

關鍵字：機器學習、影像辨識、打字習慣

研究題目：

運用影像辨識及機器學習改良打字系統與密碼

壹、 研究動機

隨著時代的改變，越來越多人必須使用電腦完成課業報告與專案計畫，但是研究者發現到許多人的打字速度非常慢，導致在做報告時常常都需要花上非常多的時間才能完成。為了提升打字的速度，許多人會去尋找網路上的打字系統練習，而目前的打字系統大多都是教導我們使用 10 指打字（圖 1），但是卻只能感應我們所按下的按鍵是否正確，並無法判斷打字者是否正確使用對應的手指按壓鍵盤。因此研究者決定改良現今的網路打字系統，製作出能夠辨識正確指法的系統。

而研究者在文獻探討時發現到其實 10 指打字並非是最快的打字方式，只要使用自己最舒服的打字習慣，照樣還是能夠打非常快。因此研究者認為可以運用大家都有不同的打字習慣進行加密，就算密碼不小心讓他人知道也不會那麼容易就解鎖。

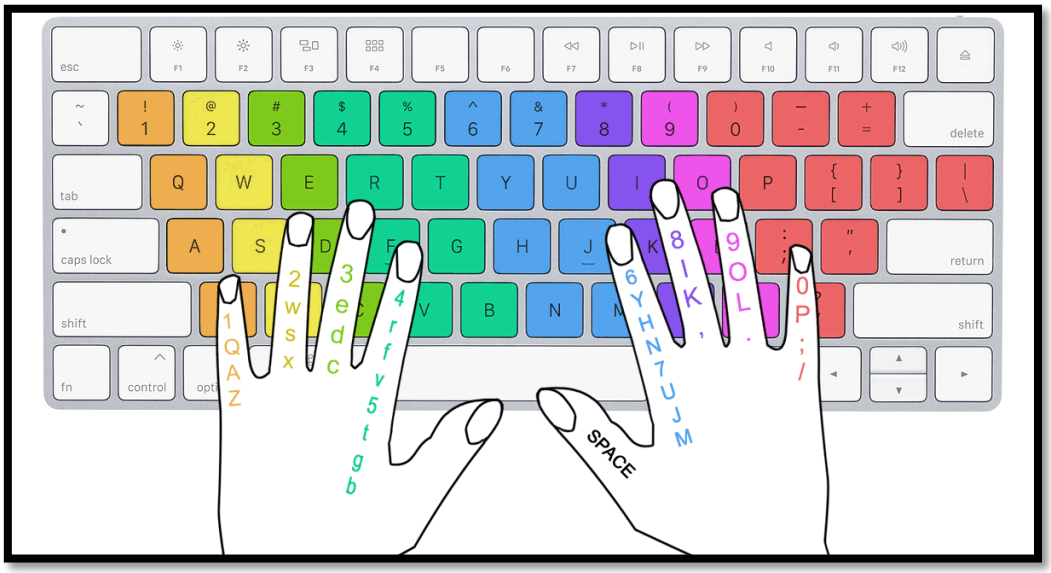


圖 1：網路打字系統的 10 指指法

貳、 研究目的

本研究想要改善現今的打字系統網站的不足，利用影像辨識判斷是否使用正確指法，讓使用者能夠學習到最標準的 10 指打字；也在打字過程中運用機器學習（Machine Learning），分析手部的動作找到打字的習慣，形成一種獨一無二的密碼。以下是本研究的研究目的：

一、 開發打字系統。

二、 運用影像辨識判斷手部位置座標。

三、 分析打字時字與字之時間習慣。

四、 分析打字時的座標習慣。

五、 比較不同機器學習模型差異。

六、 探討此生物辨識下的密碼最短長度。

七、 開發忘記密碼時的生物辨識辨識。

參、 研究方法

一、研究器材

（一）硬體：筆記型電腦、滑鼠、視訊鏡頭、腳架、鍵盤。

（二）軟體：Visual Studio Code、Python、Pygame、Opencv-python、Natural Language Tool Kit、MediaPipe、Unity、Anaconda、Jupyter Notebook、Scikit Learn。

二、研究流程

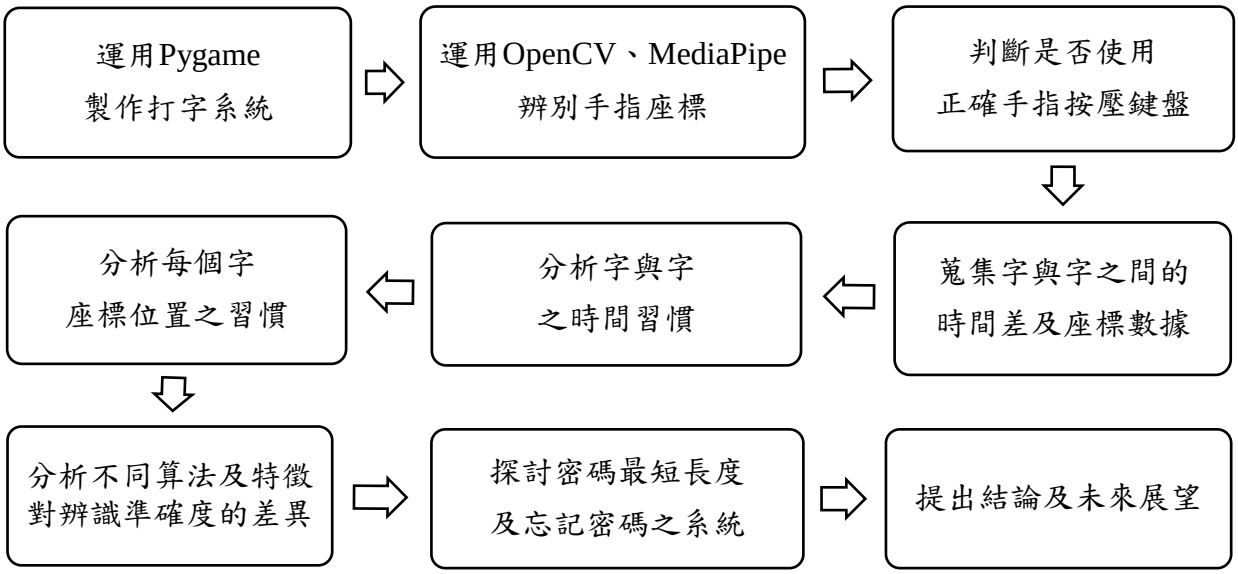


圖 2：研究流程圖

三、研究過程

（一）運用 Pygame 開發打字系統

Pygame 是 Python 製作遊戲的模組，能讓開發者加入文字、圖案、聲音，進行事件處理開發遊戲。因此研究者便決定使用 Pygame 開發打字系統（圖 3）。

圖 3：Pygame Logo

打字系統最重要的核心功能可分為三個部分：隨機生成文字、感應鍵盤、判斷輸入文字與生成文字是否相同。

首先是「隨機生成文字」。在經過了文獻查詢後，研究者發現了「Natural Language Tool Kit」，這是專門處理自然語言的工具，當中就約有多個英文單字，因此研究者只需要將單字記錄下來，並隨機顯示於打字系統畫面上即可（圖 4）。

圖 4：Natural Language Tool Kit 的單字到打字系統畫面

接下來必須處理感應鍵盤的問題。由於鍵盤上的每個按鍵按下後，電腦都會接收到訊息，得知現在是按壓到什麼按鍵。不過考量到使用者的鍵盤可能偶發感應不良情況，我們必須將打字系統畫面上的鍵盤在使用者輸入按鍵時將對應的按鍵亮起，以便於使者能夠知道自己是否有按下按鍵（圖 5）。

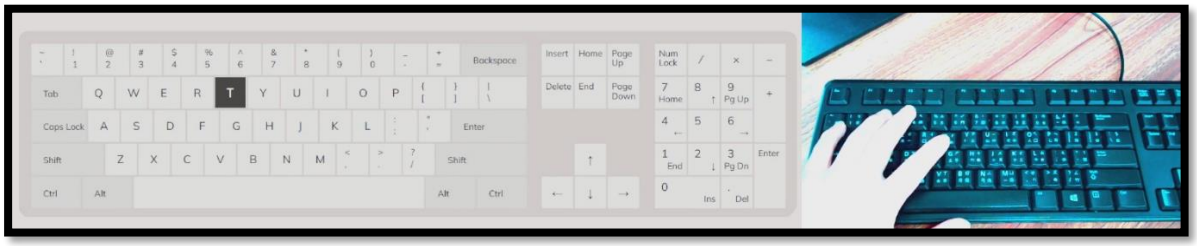


圖 5：按鍵按下後畫面上的按鍵顏色改變畫面

最後是判斷我們輸入的字元是否與系統產生的文字相同。在先前的生成文字部分研究者已把文字記錄下來，因此只要判斷我們按下的按鍵是否和文字相同即可。不過當我們輸入完一個字元後，我們必須告訴使用者現在打到第幾個字以及打得是否正確，研究者在文字底下增加底線，幫助使用者知道現在打到哪裡，並且如果使用者按錯鍵時，直到他按下正確的鍵後，字的背景會呈現紅色；若直接輸入正確，則會將背景顯示為綠色（圖 6）。

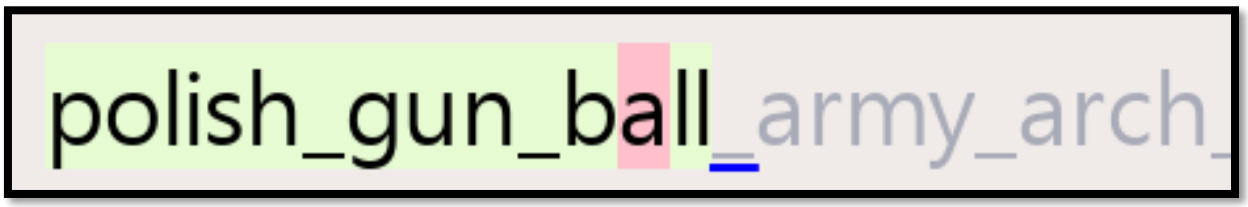


圖 6：按下按鍵後判斷是否為正確之打字系統畫面

（二）使用 OpenCV、MediaPipe 判斷手指位置

OpenCV 全名為「Open Source Computer Vision Library」（圖 7），是當今最知名的影像處理函式庫，可以讀取儲存圖片、視訊、影像處理等，可以用在物體追蹤、人臉辨識、動態視訊的影像處理等。因此研究者使用 OpenCV 幫助在影像處理的部分。

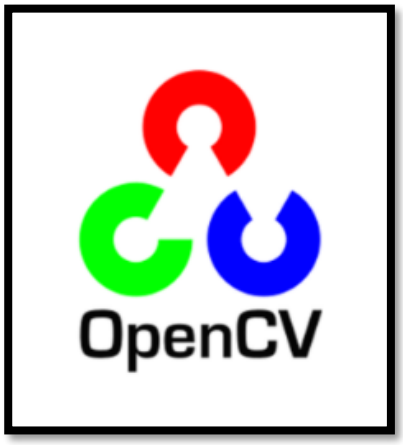


圖 7：OpenCV Logo

而 MediaPipe（圖 8）是一個整合 Media 與 Pipeline 的深度學習運行框架。不過此框架不負責優化模型訓練與模型推理，只專注於高速的 Pipeline 搭建。而到目前為止 MediaPipe 提供了 16 種 Pipeline 方案（圖 9），並開放模型下載使用。

圖 8：MediaPipe Logo

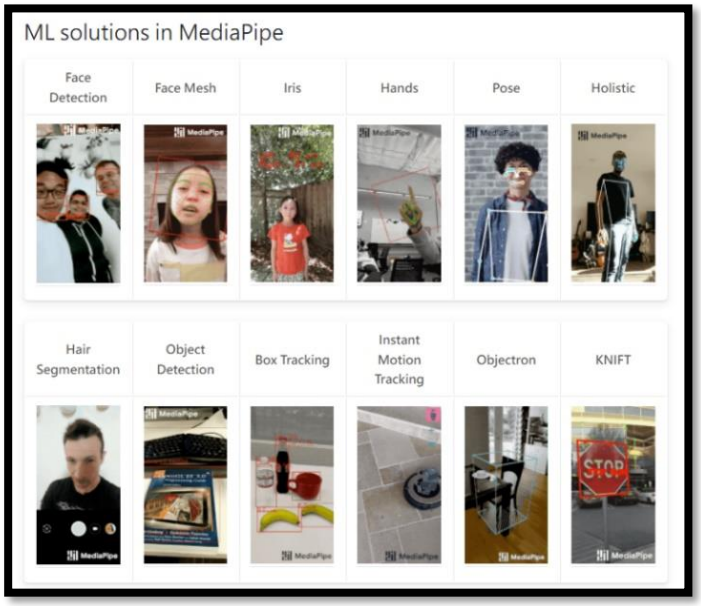


圖 9：MediaPipe 提供的模型

而此 16 個模型中，包含了偵測手指位置的模型。在這個模型主要分成兩部分：Palm Detection（手掌檢測）及 Hand Landmarks（手部座標）。手掌檢測的部分會先從影像中擷取整隻手，在找到手掌的位置。手部座標的部分在訓練模型時，在三萬多張的不同的手部照片中標示了 21 個關鍵位置的點位（圖 10）。因此從這兩個部分後，我們便可以順利取得影像中的 21 個點座標數值（圖 11、圖 12）。

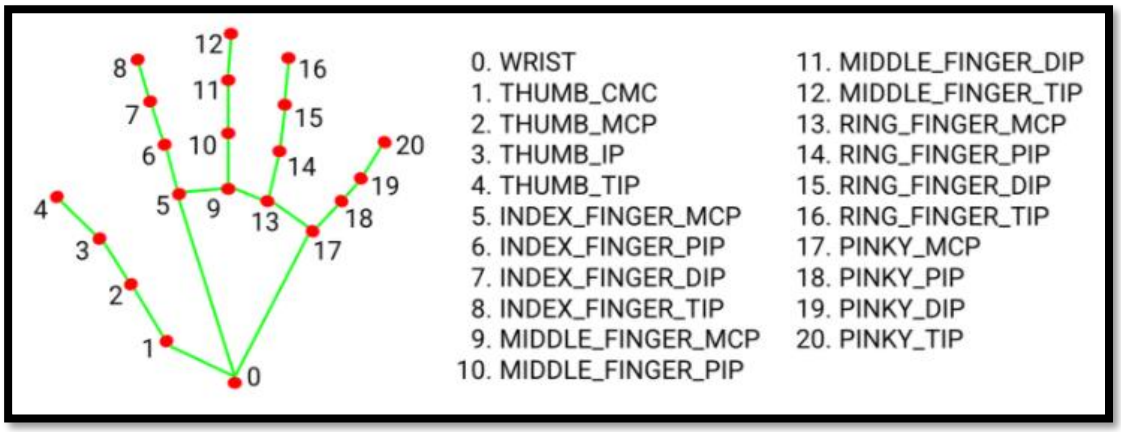


圖 10：Hand Landmarks（手部座標）

```
0 211 409
1 172 378
2 145 333
3 130 292
4 115 256
5 209 269
6 206 215
7 205 180
8 208 149
9 241 269
10 258 213
11 269 178
12 279 148
13 267 281
14 284 233
15 295 202
16 305 175
17 288 303
18 312 273
19 330 255
20 346 237
```

圖 11：21 個點位座標數值

圖 12：手部辨識畫面

（三）結合打字系統與手部辨識

為了能夠成功判斷手指位置並和打字系統結合，研究者將影像中鍵盤的每個按鍵位置予以轉換座標化（圖 13），如此一來，當我們的影像偵測到手指指尖的座標和按鍵座標有重疊時，就能夠知道我們用哪根手指按按鍵（圖 14）。

圖 13：鍵盤按鍵座標化

圖 14：手指指尖座標和按鍵座標重疊畫面

在能夠辨識出使用者使用哪根手指進行按壓後，研究者將此打字系統分成三類：若使用者使用正確手指打字，則文字背景顯示為綠色；若用了錯誤的手指但按下了正確按鍵，則會讓文字背景顯示為藍色；若是直接按錯了按鍵，則文字背景顯示為紅色（圖 15）。

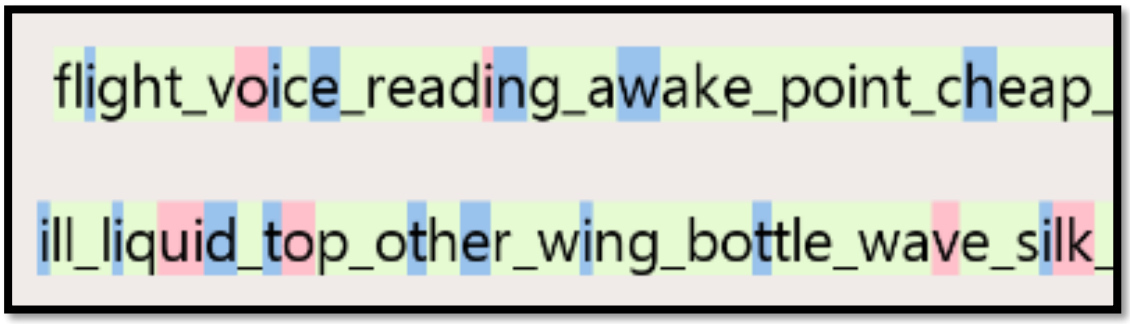


圖 15：打字系統的文字背景顏色畫面

（四）蒐集字與字之間的時間差及座標數據

研究者將「hand tracking password」當作電腦文字密碼（圖 16），當此密碼不小心外流時，其他人也沒有辦法能夠將研究者的密碼解鎖，因為他們並沒有研究者打此密碼的習慣。而研究者目前認為打字時的時間及手部座標可以觀察到每個人打字的習慣，因此開始蒐集研究者及其他人打「hand tracking password」時的每個字與字之間所需要花費的時間（圖 17）及每打一個字時，由 MediaPipe 辨識手部座標所得到的 21 個點之 XYZ 的座標數值（圖 18、圖 19）。

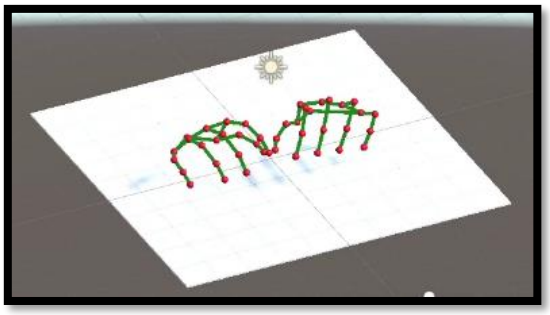
圖 16：研究者分析習慣之密碼

	A	B	C	D	E	F	G	H	I	J	
1	0.1110246	0.1483331	0.1900692	0.1890624	0.2370453	0.2520568	0.1470339	0.2400544	0.2508833	0.2880816	0.28
2	0.1520264	0.0970292	0.1490335	0.2000449	0.3370693	0.6216443	0.0970228	0.2430611	0.5009553	0.3420668	0.2
3	1.2902923	0.0990057	0.1520343	0.2871664	0.3910978	0.4841158	0.0970211	0.1930358	0.3520811	0.1950505	0.24
4	0.1520677	0.1830337	0.0980341	0.1930685	0.2500496	0.2450724	0.0950179	0.1900506	0.2940927	0.217062	0.28
5	0.104023	0.099031	0.149034	0.2836528	0.4428325	1.0224476	0.0950141	0.1470504	0.2990766	0.2430549	0.43
6	0.1530416	0.1450329	0.1410253	0.1931584	0.1390388	0.3820813	0.2410769	0.3014679	0.3820791	0.1980519	0.25
7	0.1030107	0.0920205	0.0990157	2.4822104	0.3820884	0.148031	0.0990303	0.1990447	0.5317254	0.1390302	0.30
8	0.8662004	0.1425653	0.2110476	0.4771469	0.1948512	0.3016219	0.0480108	0.1980474	0.337086	0.0940361	0.15
9	0.1020303	0.1430321	0.0990224	0.2360461	0.2420552	0.3800921	0.0970225	0.1920424	0.4388285	0.1440432	0.24
10	0.1180248	0.1090326	0.0940135	0.1440413	0.2850556	0.0980225	0.0970604	0.1590536	0.1930516	0.1460245	0.23
11	0.1040232	0.1450248	0.1910508	0.1440325	0.2920735	0.0990148	0.1020257	0.1480227	1.6503661	0.7612014	0.34
12	0.1180198	0.1440327	0.1441979	0.1390331	0.3803155	0.0950265	0.0440114	0.1920373	0.2890699	0.1430402	0.23
13	0.1000285	0.1410382	0.1550272	0.1410336	0.3365827	0.1015341	0.0960133	0.2030597	0.2881763	0.144032	0.19
14	0.1020226	0.1450338	0.0980289	0.1430321	0.2805223	0.1121984	0.0990255	0.1900537	0.4760954	0.3541658	0.52
15	0.1070223	0.1020157	0.1950431	0.1430328	0.3050764	0.0940177	0.1000259	0.1930435	0.6311388	0.1470311	0.19
16	0.1090317	0.1450248	0.0980222	0.1930432	0.2420549	0.098022	0.1110249	0.1510351	0.2480555	0.1470337	0.19
17	0.1020215	0.0970218	0.1870425	0.1901853	0.2432032	0.1620307	0.1400301	0.18905	0.2410939	0.144032	0.20
18	0.0990188	0.095031	0.1420307	0.1880412	0.3300996	0.0950286	0.0450113	0.1920421	0.243058	0.1540315	0.19
19	0.1020229	0.1000226	0.1989896	0.1492288	0.2892964	0.0970223	0.0950181	0.1900411	0.2540205	0.1420457	0.19
20	0.1030307	0.096014	0.1450405	0.1960442	0.3390682	0.095022	0.0480113	0.202065	0.2440481	0.1400363	0.1
21	0.1010225	0.1440396	0.096025	0.1870391	0.2200413	0.1430326	0.096031	0.2320423	0.1890433	0.1420319	0.14

圖 17：字與字之間時間差部分數據

	Q	R	S	T	U	V	W	X	Y	Z	AA	AB	AC	AD	AE
1	612	-143	440	473	-188	474	392	-202	500	329	-206	301	579	-132	349
2	603	-119	470	474	-159	509	394	-171	536	333	-176	318	572	-112	362
3	608	-159	430	469	-209	462	383	-224	486	315	-229	293	575	-146	334
4	621	-132	449	478	-177	479	394	-192	503	330	-197	318	586	-125	366
5	609	-141	462	475	-185	499	394	-197	528	331	-202	312	576	-134	359
6	605	-129	474	471	-171	506	388	-182	531	324	-185	334	577	-119	364
7	607	-156	430	454	-203	473	372	-209	506	316	-204	284	573	-143	334
8	630	-128	486	503	-168	530	426	-178	562	368	-181	324	592	-121	388
9	602	-106	466	467	-143	503	388	-155	531	327	-160	317	565	-102	371
10	620	-147	447	480	-191	490	405	-197	522	351	-196	289	584	-136	354
11	607	-147	447	464	-190	481	380	-199	505	318	-198	305	573	-133	344
12	592	-124	476	448	-163	513	369	-170	538	311	-170	327	554	-118	382
13	619	-124	463	479	-163	502	401	-171	530	343	-171	311	580	-119	374
14	613	-129	450	479	-170	488	399	-181	515	338	-184	303	577	-119	356
15	605	-145	446	461	-188	486	382	-195	513	325	-194	295	571	-136	352
16	614	-129	457	480	-171	497	399	-183	527	337	-186	308	576	-120	358
17	608	-132	455	465	-171	491	384	-179	518	323	-179	304	574	-124	355
18	613	-119	464	474	-157	503	395	-166	531	337	-168	315	577	-112	370
19	604	-104	462	466	-142	501	388	-152	531	331	-155	315	566	-102	373
20	613	-125	426	475	-164	466	398	-172	494	340	-172	274	576	-117	338
21	609	-123	452	470	-165	488	388	-178	516	325	-182	312	573	-117	362
22	610	-129	460	474	-170	499	393	-181	528	330	-183	311	578	-121	358
23	601	-114	463	464	-151	499	383	-158	527	323	-159	314	565	-109	368
24	615	-171	460	471	-220	494	390	-229	520	328	-229	313	586	-161	361
25	603	-134	463	473	-178	506	397	-190	538	338	-194	314	566	-124	366
26	622	-126	456	480	-166	500	403	-174	533	348	-174	301	582	-120	371
27	608	-138	465	468	-182	500	385	-194	526	321	-197	320	575	-131	371
28	614	125	446	466	-163	482	396	-168	508	320	-166	200	580	115	351

圖 18：每一個字的打字手部座標部分數據



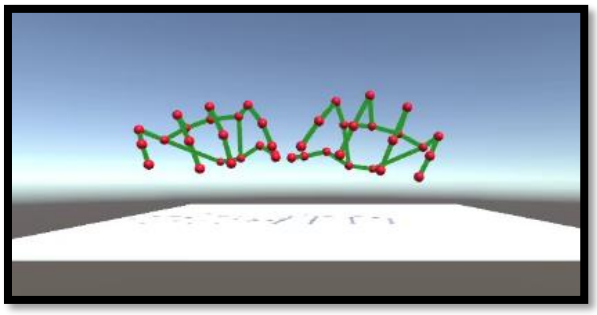


圖 19：MediaPipe 所得到 21 個點座標的 3D 圖形

（五）機器學習決策樹演算法（Decision Tree）

決策樹模型有許多類別，而本研究使用到的是分類樹（Classification Tree）。此模型為監督式模型的一種，看起來就像是一顆樹，每個節點都是一個特徵，根據特徵一步一步的細分，讓我們可以分出不同類別的資料。

而決策樹的運作原理可以分成兩個部分。首先是「Entropy」，代表一個不確定性的量度（圖 20）。當 Entropy 越小，代表越穩定的狀態，若 Entropy 越大，則為最混亂且無序的狀態（圖 21）。因此此模型目的就是要讓 Entropy 漸漸減少到趨近於 0。

$$Entropy = H(S) = \sum_{i=1}^{C} -p_i log_2 p_i$$

圖 20：Entropy 計算公式

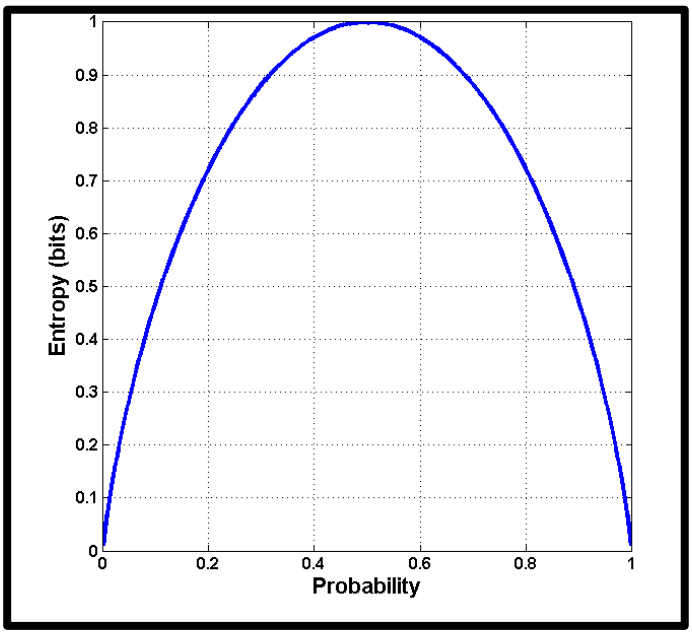


圖 21：Entropy 數值示意圖

那要怎麼讓 Entropy 越來越小呢？這時候就需要「Information Gain」（圖 22），是將每個節點跟上一個節點去做比較，若數值越大，則此特徵越有貢獻，反之則貢獻較少（圖 23）。換句話說，即為不同狀態時 Entropy 的差值。如此一來，我們就能找到最合適的特徵讓 Entropy 的值逐漸趨近於 0。

$$Information\ Gain = Gain(S, A) = H(S) - \sum_{i} \frac{|S_i|}{|S|} H(S_i)$$

圖 22：Information Gain 計算公式

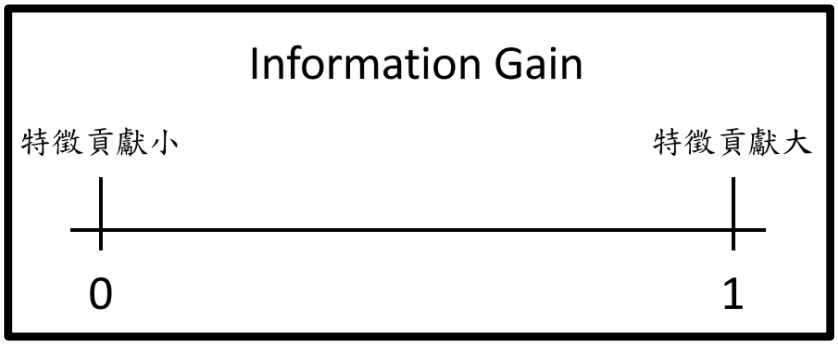


圖 23：Information Gain 數值示意圖

（六）機器學習隨機森林演算法（Random Forest）

隨機森林其實就是進階版的決策樹，也就是眾多樹組成的樹林，是集成學習 Bagging 及抽取資料方法 Bootstrap 所建構的演算法。在每個決策樹中只能看見部分特徵，並利用多數決集合每顆樹的預測，防止模型容易發生 overfitting。要生成一個好的隨機森林模型分類器，我們會將訓練集中抽取n'筆資料，並在這n'筆資料中挑選k個特徵做樣本。在重複m次後會產生m顆決策樹，最後在進行多數投票制進行預測。

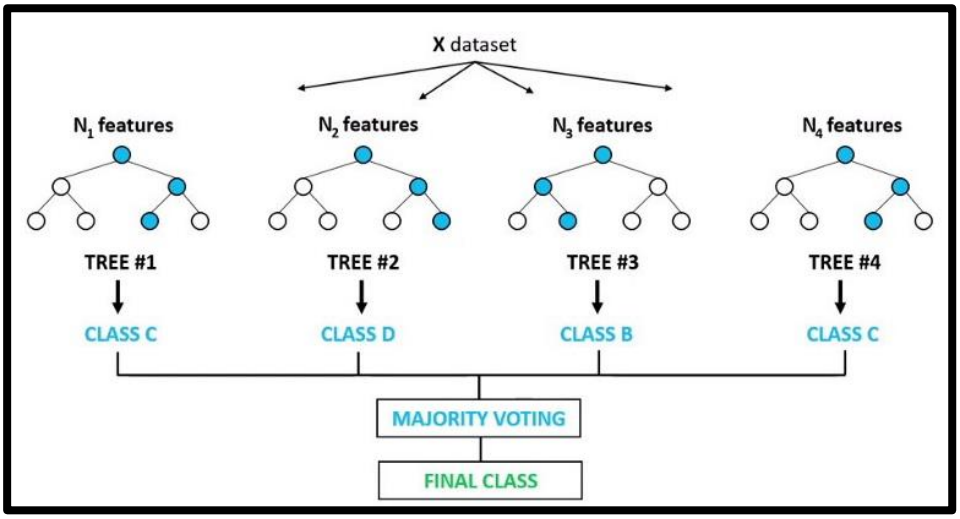


圖 24：Random Forest 模型建構流程

（七）在打字時間差及手部座標的生物辨識下密碼最短長度探討

現代密碼的設置多為八碼設置，有些密碼還會要求要大寫英文字母等，為了就是能夠降低密碼被猜到的機率。但長度越長的密碼，也會越容易記錯，因此就開始探討在此生物辨識下最短的密碼長度為多少。

研究者決定了三組隨機的八碼數字，並將每組數字的每個長度都輸入 50 次（圖 30 說明了資料約只需 50 筆便能分辨）。舉例來說，現在有一組密碼：12345678，那麼我們就必須將輸入 1 達到 50 次，再將輸入 12 到 50 次，依此類推，最後就是將 12345678 輸入 50 次。有了資料集後，就能夠看出到達什麼樣的密碼長度時，模型準確率會一直都很高。

（八）忘記密碼時生成文字的生物辨識判斷系統

現代忘記密碼的系統都是寄 mail 或著是簡訊讓忘記密碼的人能夠重設密碼，但是有個缺點就是我們必須要多開另外一個系統來確認是否為真正的使用者。為了能夠不用那麼麻煩的打開 mail 或著簡訊，我想到運用此生物辨識技術可以讓我們不用這麼麻煩的就開啟我們忘記密碼的網站。

當忘記密碼時，首先將原密碼拆分成n個字串，並找到有哪些字是和這n個字串是有共同子序列的，並給使用者進行打字。舉例來說，假設密碼現在為 12321567，那我們可以讓系統生成兩個字串分別為 12345678、654321，並讓忘記密碼者輸入此兩字串，紀錄打字特徵（時間差或手指座標）。這時我們只需要擷取打第一個字串的 123 及 567 時的打字習慣，還有第二個字串打 21 時的打字習慣，並將這三個部分的結合，如此一來，我們便有了原密碼 12321567 的打字習慣。因此研究者會先隨機選擇三組數字密碼，並將這些數字密碼拆分成n個字串，比較不同n值時的分辨情況。

肆、 研究結果

一、 改良後可判斷使用者打字手指之打字系統

在經過了上述的開發設計與測試精進過程，總算改善現今網路上的打字系統的缺點，也就是沒有辦法知道使用者是否是用正確的手指進行打字（圖 25）。使用者在一開始會進入初始介面（圖 26），按下任意鍵後即開始打字練習（圖 27），而打完後會在詢問使用者是否要再一次進行練習（圖 28）。

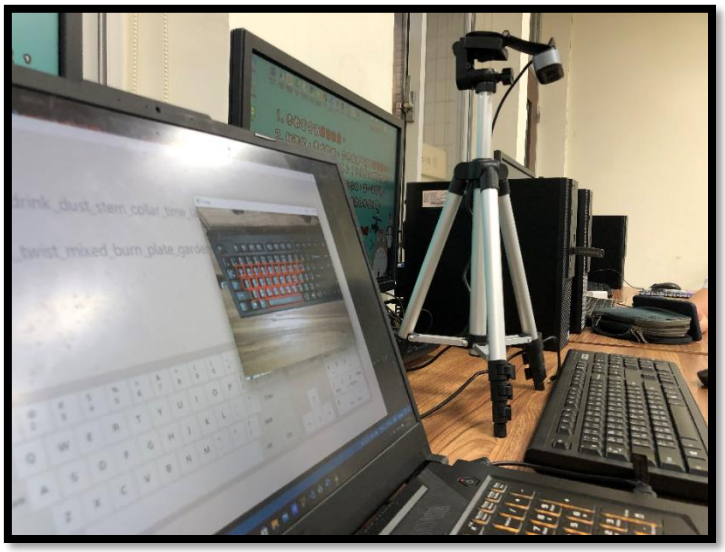


圖 25：打字系統裝置圖

圖 26：打字系統初始介面

圖 27：打字練習畫面

圖 28：重新開始畫面

二、 運用決策樹分析打字習慣

研究者目前已蒐集了 130 次 Person A 及 Person B 打「hand tracking password」時的時間差及手指座標位置。

首先使用 Person A 及 Person B 的打字時間差所建構的決策樹樹狀圖如下圖 29。可以發現在第六個數值為主要影響機器判斷是 Person A 還是 Person B 正再打字，也就是 Person A 與 Person B 的字母 t 到字母 r 的時間是非常不一樣的，而這就是我們的打字習慣差別。當我們將資料以不同比例分為訓練資料及測試資料後的結果如下圖 30。可以發現到當我們訓練資料只有10%時，準確度居然還有高達96.58%，而在訓練資料大於50%後，機器便能夠100%的辨別是 Person A 還是 Person B 在打 hand tracking password 這串密碼。

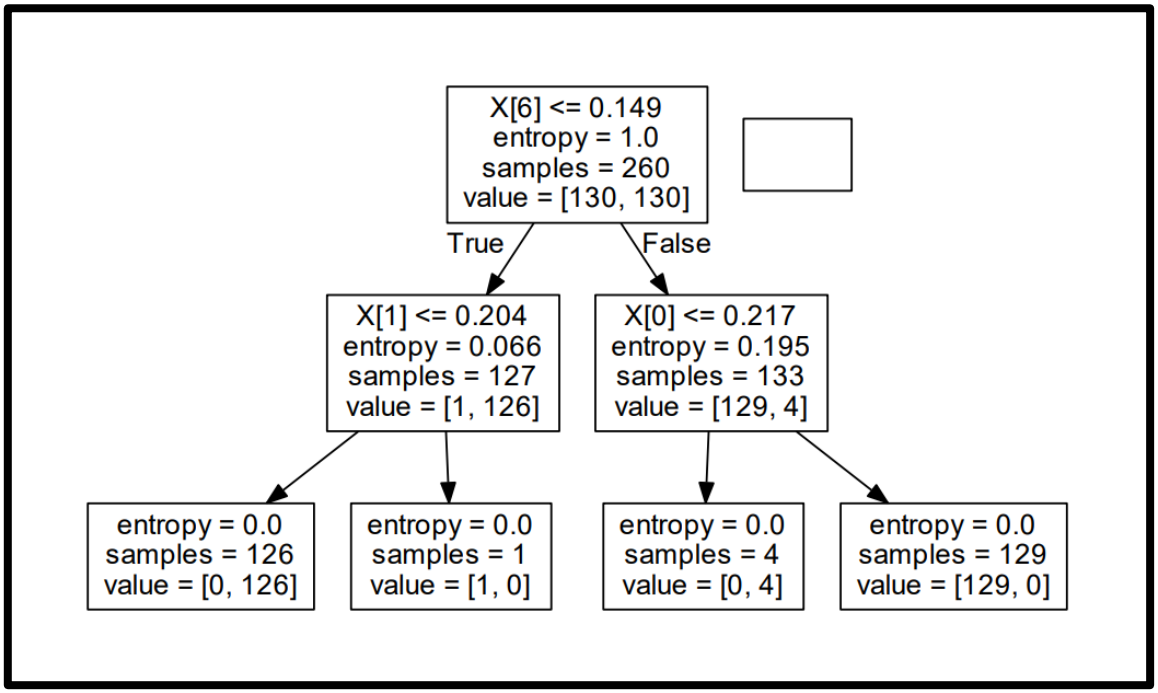


圖 29：Person A 及 Person B 的打字時間差之決策樹樹狀圖

圖 30：不同訓練資料量機器分辨 Person A 還是 Person B 所打字的準確率

由於訓練資料量如此少時，機器的準確度還有96.58%的準確率，因此只要可知只要兩人的習慣差很多，機器就能夠分辨得出來是誰在打字。但為了驗證機器也能夠分辨出此人是否為可以解開密碼的人，研究者將 Person A 打字的資料假設為兩組，並將兩組的標籤設為 Person A 與 Person a 帶入模型當中訓練。若分類模型訓練後準確度非常低，就代表了模型其實分類

不太出來這兩人，也就是他們應該比較有可能為同一人。

圖 31 為 Person A 與 Person a 的打字時間差所建構的決策樹樹狀圖，可以看到此樹狀圖已經分到第 7 層，代表 Person A 與 Person a 的資料非常相近。而研究者也將在不同比例下的訓練資料與測試資料之準確度與 Person A 和 Person B 的準確度做比較（圖 32），可以發現到就算使用了90%的訓練資料，準確度也只有77.78%，且和 Person A 及 Person B 的準確率做比較，準確率也有明顯下降許多。因此如果是預設可以為解開密碼的人，那麼機器很大的機率可以判斷的出來。

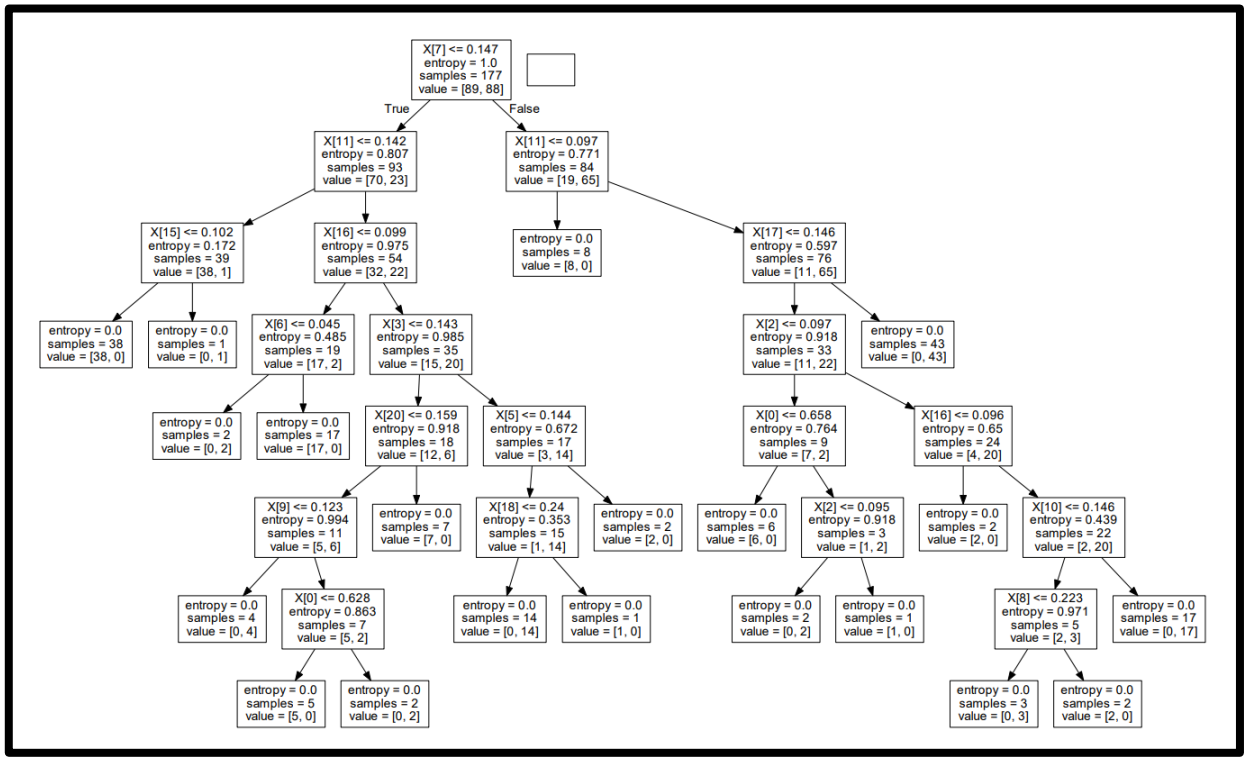


圖 31：Person A 及 Person a 的打字時間差之決策樹樹狀圖

圖 32：不同訓練資料量 Person A、Person a 模型準確率

與 Person A、Person B 的模型準確率比較

接下來研究者也嘗試每次打字時 Person A 與 Person B 的手指座標。圖 33 為 Person A 與 Person B 的每個字的手部座標當作特徵所建構的決策樹樹狀圖。由於我們的特徵是每次打字時的左右手的 XYZ 座標，所以每打完一次 hand tracking password 共有 $21 \times 2 \times 3 \times 22 = 2772$ 個數值，要在這 2772 個數值中，找到特定的習慣是非常有可能的，因此決策樹的樹狀圖才不會那麼多層。

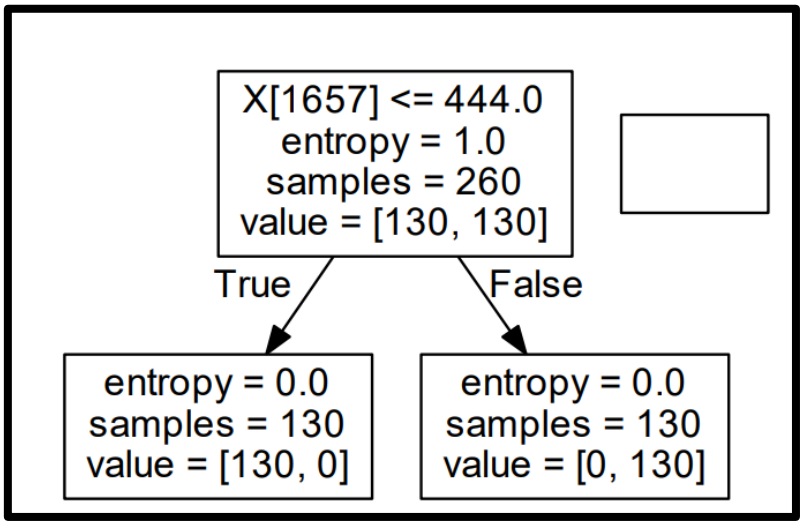


圖 33：Person A 及 Person B 的手部座標之決策樹樹狀圖

接下來我們觀察在不同比例的訓練資料以及測試資料的結果（圖 34）。由圖中可以發現相對於打字時間差在 50%以後準確率都是100%以上，手指座標在大於40%以上後準確率就全都是100%了。由此可知，要在座標中找到兩人不一樣的習慣是非常容易的，而在將資料一半以上去訓練模型很有可能準確度會達到100%。

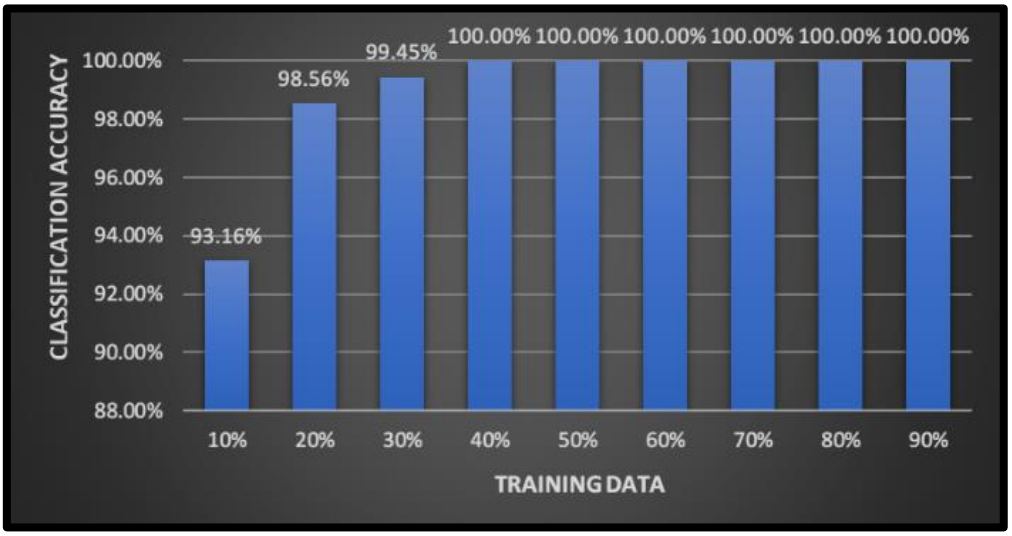


圖 34：不同訓練資料量機器分辨 Person A 還是 Person B 所打字的準確率

既然這麼好分別出不同人，那是否無法輕易判斷出允許開啟密碼的人呢？因此研究者將 Person A 打字的手指座標資料分成兩組，並將兩組的標籤設為 Person A 與 Person a 帶入模型當中訓練。圖 35 即為 Person A 與 Person a 的手部座標樹狀圖，雖然 Person A 與 Person a 的樹狀圖層數比 Person A 和 Person B 的層數還要多，代表資料還是較為相近，不過相對於時間差時 Person A 與 Person a 的樹狀圖來說層數是有減少的狀況。因此若排除資料量不夠的情況，或許時間差會比手部座標的判斷還要準確。那在看到不同比例的訓練及測試資料的部分（圖 36），同一個人數據訓練出來的模型沒有辦法達到很高的準確率，而且隨著數據量的增加，分類準確率也沒有明顯的上升趨勢。但很有意思的是，當同一個人只有20%的數據時，模型分類準確率和不同人準確率的差別就更小了。我想這也是因為我們有更多的坐標值，所以模型很容易找到差異。這也意味著，如果我們要使用手部坐標進行生物識別，數據量必須多一點，這樣更容易區分不同的人。

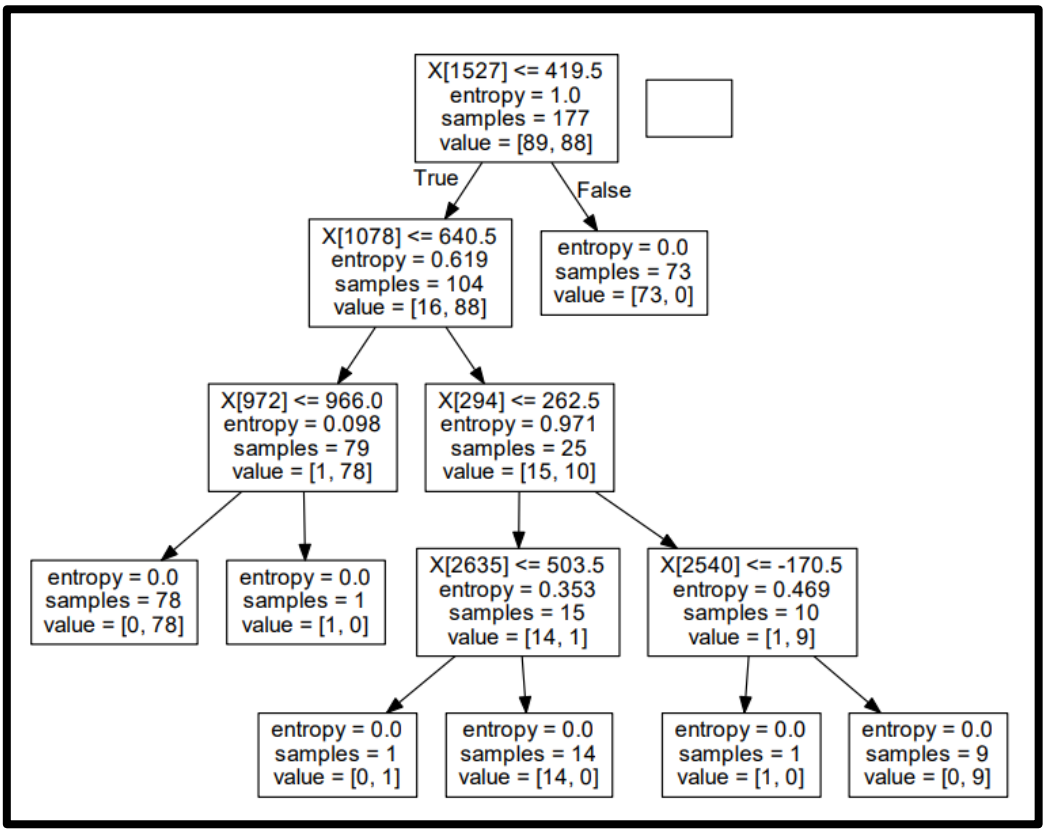


圖 35：Person A 及 Person a 的手部座標之決策樹樹狀圖

圖 36：不同訓練資料量 Person A、Person a 模型準確率

與 Person A、Person B 的模型準確率比較

三、 運用隨機森林分析打字習慣

隨機森林模型是將決策樹模型進行優化，透過生成多棵決策樹進行投票預測。首先我們先看到時間差為特徵，建構 5 顆決策樹的 Person A 與 Person B 隨機森林模型（圖 37）。從圖 38 我們可以看到在決策樹數量只有 5 顆的時候，我們 training data 的準確率並沒有到100%，這也就代表著我們的模型有防止到 overfitting 的情況，而在 testing data 的部分，準確率也有達到99.51%。而在更多數量的決策樹所建構的模型中，training data 與 testing data 的準確率皆達到 100%。而在不同比例的訓練資料下，研究者也發現到隨機森林比決策樹模型的準確度還要來的高，甚至在更少資料的情況下，也能夠有很高的準確度，而決策樹就相對沒有那麼出色了。由此可知，我們的隨機森林演算法所建構出來的模型確實有優化原本決策樹所建構出來的模型。

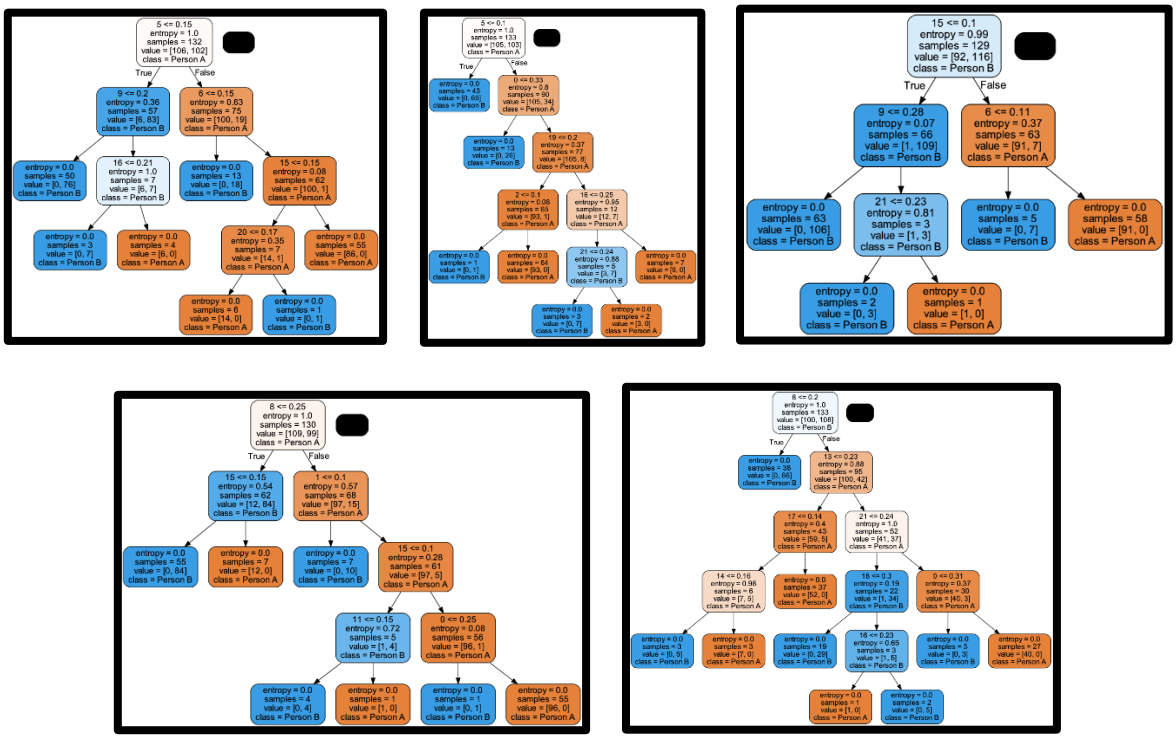


圖 37：Person A 與 Person B 時間差的隨機森林模型

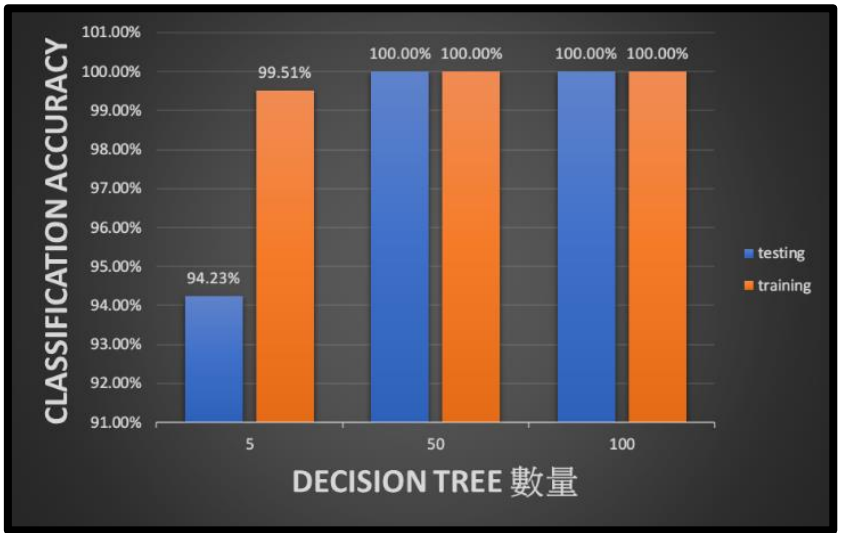


圖 38：不同決策樹數量的隨機森林模型準確率

伍、 討論及應用

一、 討論

1. 不是用對的手指按壓，系統卻判定為正確手指按壓正確按鍵

從圖 39 與圖 40 中可以看到使用者用左手按壓按鍵 H，然而實際上應該是使用右手打，因此這是一個用錯誤手指按下正確按鍵的事件，不過系統卻判定使用者為正確按鍵及正確手指。而這是因為目前此系統僅判斷 XY 座標是否重疊，並沒有將 Z 座標考慮進去。因此研究者認為要解決此問題必須要增加另外一個鏡頭，判斷手指的 Z 座標，若和鍵盤離太遠則判定不正確。

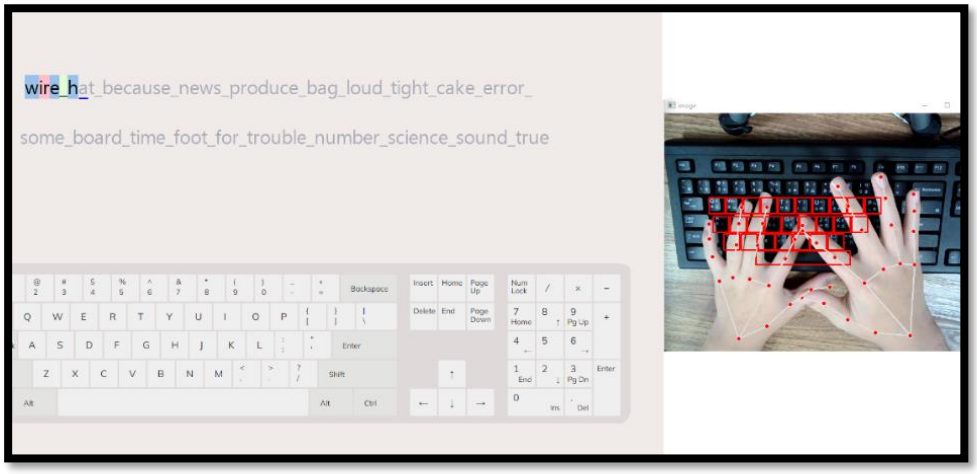


圖 39：使用錯誤手指按壓但系統判定正確按壓畫面

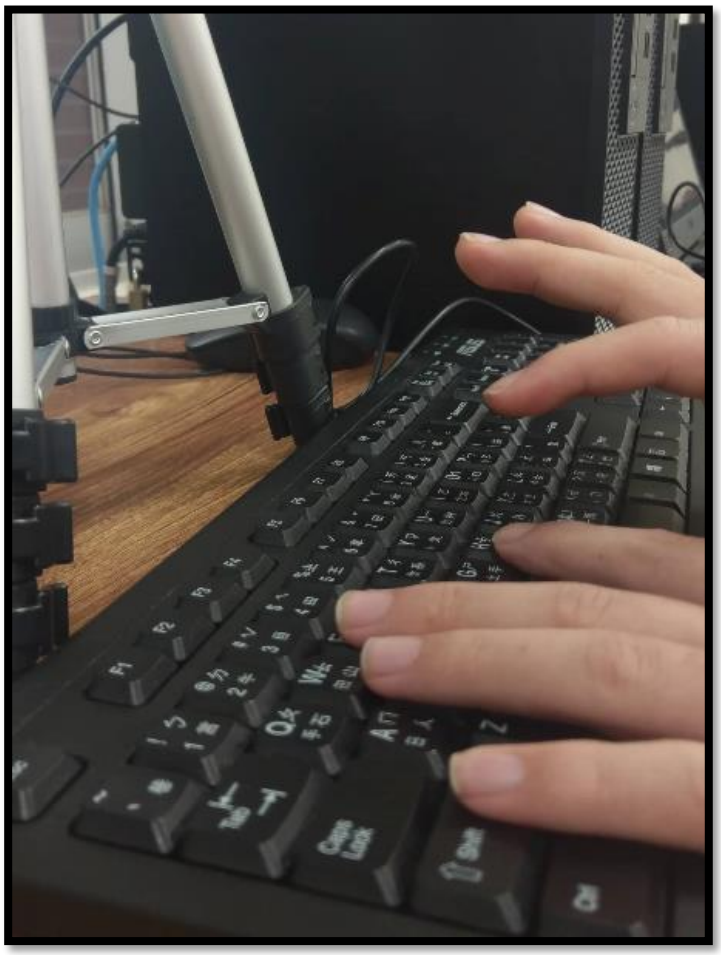


圖 40：右手離鍵盤有一段距離導致系統誤判情況

2. 使用正確手指及按鍵，但系統判定沒有使用正確手指按壓

於圖 41 中可以看到右側手指中，使用者按壓了英文字母 I 按鍵，但系統判定我們使用錯誤的手指，仔細觀察後發現，辨識手部的動作時，中指指尖的座標並非位於按鍵 I 上，因此系統會認定使用者並沒有用正確的手指。研究者會有此問題的發生有可能是因為 MediaPipe 所提供的內建模型還有優化的空間，所以有時候還是有點誤差值。若要解決這個問題，那就必須要優化模型，讓模型的準確性更高。

圖 41：使用正確手指打字系統卻判定錯誤

3. 調整取得影像的時間點

目前的打字系統在按下按鍵時會拍攝一張照片（圖 42），但是想必從按下按鍵到拍攝照片這兩個動作之間，肯定是會有一些時間差的，而這短短的時間，我們的手或許就有位移的動作，導致我們拍攝出來的手部座標並非那麼的準確。因此要解決這個問題，就必須要計算這兩個動作之間的時間差，並且回朔到過去拍攝的照片中，找到那張按下按鍵時的即時影像。

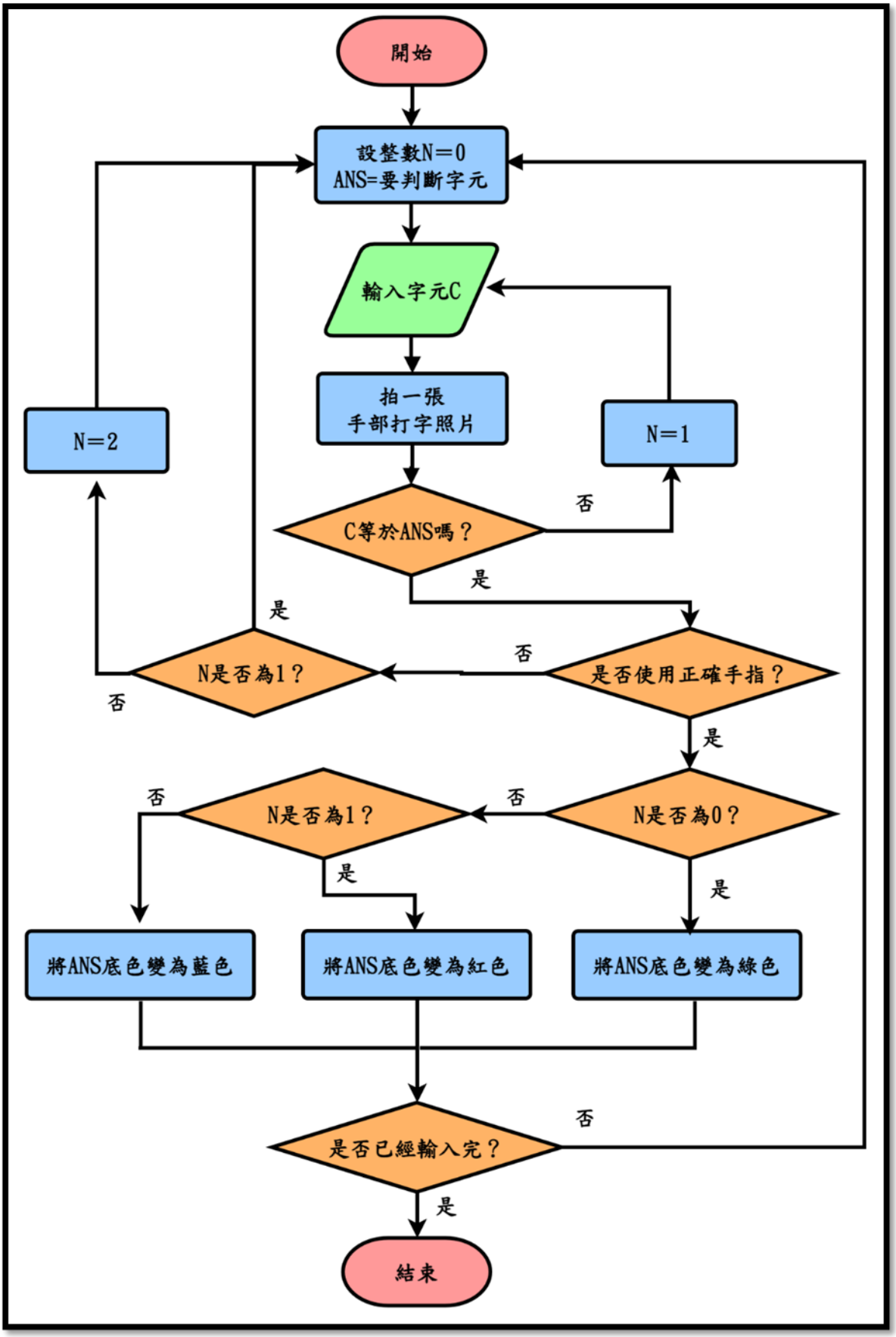

開始
設整數N＝0
ANS=要判斷字元
輸入字元C
拍一張
手部打字照片
N=2
N=1
C等於ANS嗎？
否
是
是
N是否為1？
否
否
是否使用正確手指？
否
是
否
N是否為1？
否
N是否為0？
是
是
將ANS底色變為藍色
將ANS底色變為紅色
將ANS底色變為綠色
否
是否已經輸入完？
是
結束

圖 42：打字系統流程圖

4. 密碼輸入者過少

由於本研究需要使用影像辨識來觀察打字者的手部動作習慣，需要有同一視角的拍攝才能夠有最準確的座標點位，因此無法像時間差做個網頁去讓他人打字，以便更輕易的蒐集更多資料。

而這導致目前研究只限於在 Person A 和 Person B 兩人的打字習慣以及 Person A 和 Person a 的判斷，如果想要更確定能夠使用時間差以及手部座標當作打字習慣進行生物辨識的話，那就需要再增加更多使用者來增加資料集的數量，確保是能夠廣泛使用此技術的。

二、 未來展望

本研究首先改善現今打字系統無法判斷使用者是否使用正確的手指按壓鍵盤，並透過打字時間差與手指座標建構決策樹和隨機森林模型，分析 Person A 與 Person B 的打字習慣。

未來將嘗試給大眾使用此打字系統，測試是否對於 10 指打字是否有幫助與改善，並將上述問題逐一解決並提高模型準確度，也會蒐集更多人輸入「hand tracking password」密碼時的時間差及手部座標並進行分析，也將嘗試使用更多不同的演算法來提升模型的準確度。而在最短密碼及忘記密碼的輸入系統中，目前都在進行資料的蒐集，預計在決賽報告前完成資料的蒐集並且訓練模型並觀察結果。

本研究最值得的地方是能夠運用在 ATM 上。只要將本研究的系統架設好，便可以進行時間差或著手指座標的打字習慣判斷。若有人知道他人的提款帳號，但他也沒有辦法在 ATM 成功領取，因為他並沒有密碼擁有者的打字習慣。未來也希望能夠使用 Deep Learning 去判斷打

字習慣，依照拍攝的照片進行訓練後看能否更有效率及準確度的進行分辨的動作。而目前我們都只是使用 hand tracking password 進行打字習慣的分辨，未來也會嘗試往在日常使用電腦時就紀錄習慣，並在輸入密碼時，可以預測使用者應該會如何打這串密碼。

陸、 參考資料

一、Clay（2019 年 7 月 30 日）。英文自然語言處理的經典工具 NLTK。https://clay-atlas.com/blog/2019/07/30/nlp-python-cn-nltk-kit/

二、Iqra Anwar（2021 年 12 月 30 日）。MediaPipe Python Tutorial [How to Install + Real-Time Hand Tracking Example]。https://reurl.cc/6EeVWV

三、Moxa ken（2020 年 9 月 9 日）。[30 天 Unity 大破解] 2:準備進入 unity 的世界。https://ithelp.ithome.com.tw/articles/10235005

四、Murtaza's Workshop - Robotics and AI（2021 年 3 月 25 日）。Hand Tracking 30 FPS using CPU〔影片〕。YouTube。https://www.youtube.com/watch?v=NZde8Xt78Iw

五、onlinetyping.org(2016 年 5 月 7 日)。Touch typing online lesson。https://onlinetyping.org/typing-lessons/touch-typing-online-lesson.php

六、PyInvest（2019 年 7 月 15 日）。[機器學習首部曲] 決策樹模型介紹 Decision Tree。https://pyecontech.com/2019/07/15/decision_tree/

七、Rice Yang(2021 年 4 月 12 日)。用 MediaPipe 快速搭建 Hand Tracking。https://reurl.cc/Ope7e3

八、TechBrige 技術共筆部落格（2019 年 10 月 19 日）。使用 Python 和 PyGame 遊戲製作入門教學〔部落格文章〕。https://reurl.cc/akzpNZ

九、自學成功道（2022 年 1 月 14 日）。9 個免費英文打字練習網站：讓你寫程式、回信件超快速。https://selflearningsuccess.com/typingpractice/

十、作者不詳（2021 年 8 月 10 日）。利用 MediaPipe 的手部追蹤来控制您的 Mirru 機械手。https://picture.iczhiku.com/weixin/message1628544487144.html

十一、作者不詳（2020 年 8 月 7 日）。MediaPipe 跨平臺機器學習應用開發框架。

https://codertw.com/%E7%A8%8B%E5%BC%8F%E8%AA%9E%E8%A8%80/741938/

十二、洪錦魁（2022）。OpenCV 影像創意邁向 AI 視覺王者歸來。台灣：深智數位。

十三、緯育 TibaMe（2020 年 3 月 23 日）。你的 AI 會看圖嗎?Open CV 介紹〔部落格文章〕。https://blog.tibame.com/?p=15141

十四、賴屹民（2017）。OpenCV 3 學習手冊。台灣：歐萊禮。