資電領域優等(2)

摘要

作品名稱：AI 偵測火災警報系統

本研究想要使用 AI 視覺辨識來判斷火災，搭配 NTC、MQ2 及 CO2 感測器，發展出偵測、判斷、滅火及警訊四個模組。偵測模組觀察環境變化，透過 Jetson Nano 將數據上傳至雲端；判斷模組利用火焰的特徵發展模型，並將即時影像與模型做比對；警訊模組則根據感測器數值或影像辨識結果做不同程度的提醒；滅火模組則當判斷為火災時將會啟動灑水系統。

研究發現，透過不同演算法測試可以達到 98%以上的正確率。而傳統只用溫度、煙霧及二氧化碳的變化在時效上不如即時影像判斷。此外若確定火災發生時，可以在鄰近灑水系統先行動作，建立防火牆來降低財損。而本系統所發展的警報系統具有正確率高、即時判斷、容易與現有監視器結合，容易移植等優點。

專題內容

一、前言

根據內政部消防署資料顯示，107 年火災共發生 27922 次，死傷人數有 463 人，財物損失達 5 億 9592 萬元；108 年火災發生了 22866 次，死傷人數有 628 人，財物損失高達 14 億 4221 萬元，所以我們想發展一套智慧監控裝置，若確認有火災發生，可立即通報與處理。結合視覺辨識和原先的感測器，讓使用者透過手機 APP 及時得知火災發生。

為了達成上述目的，我們對於以下三個問題進行探討

(一)研究以視覺辨識偵測火災發生的可行性。

(二)研究不同場域所搭配的各式感測器，且發展火災偵測系統，去分析所有可能會碰到的困難與解決方法。

(三)探討本偵測系統可行之應用模式。

本研究使用研究設備與器材如下表 1

表 1 研究設備與器材

名稱	規格	數量	用途
Jetson Nano Board	Nano B01	1 個	執行 Python 程式
Camera 8MP CSI	800 萬畫素	1 個	拍攝環境畫面
煙霧感測器	MQ-2	2 個	偵測煙霧
二氧化碳感測器	DS-CO2-20	2 個	偵測二氧化碳
溫溼度感測器	NTC	2 個	偵測溫度

二、研究方法(過程)

(一)視覺辨識模型與運算的選擇

在訓練資料集中使用火災與正常畫面為各 900 張照片，而在驗證資料集中，火災與正常畫面各 100 張照片，將這些照片透過 CNN 演算法建立初步模型，最後再研究如何搭配不同的演算法，提高本系統正確率。

圖 1 火災視覺辨識訓練初步規劃

(二)警報系統初步功能規劃

我們針對辨識火源的部分運用了二氧化碳、煙霧及溫度感測器，搭配攝影機視覺辨識，希望能透過以上這四種方式精確的判斷是否有火災，以降低誤判的機率。以下是滅火警報器初步規劃的功能圖：

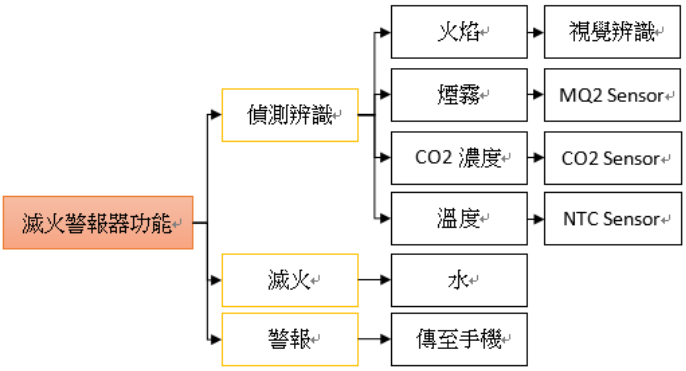


圖 2 滅火警報器初步功能架構圖

(三)即時傳遞資料:我們希望能將蒐集到的數據自動傳送到雲端，同時方便讓使用者收到即時資訊。因此我們建立 Google 雲端資料庫，使資料能同步到手機。申請金鑰與寫入 Google Spreadsheet 的流程如下圖 3。

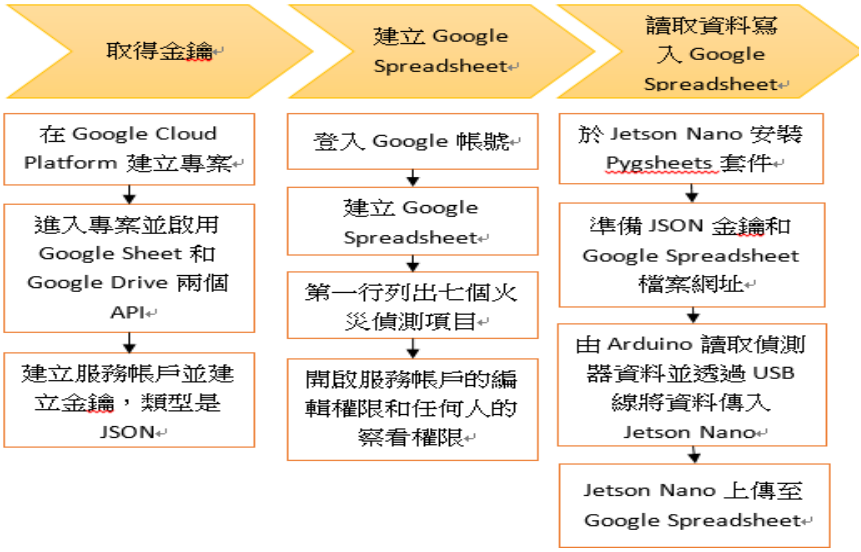


圖 3 申請與寫入 Google Spreadsheet 流程

(四)系統功能架構圖:本系統功能架構如下圖 4。

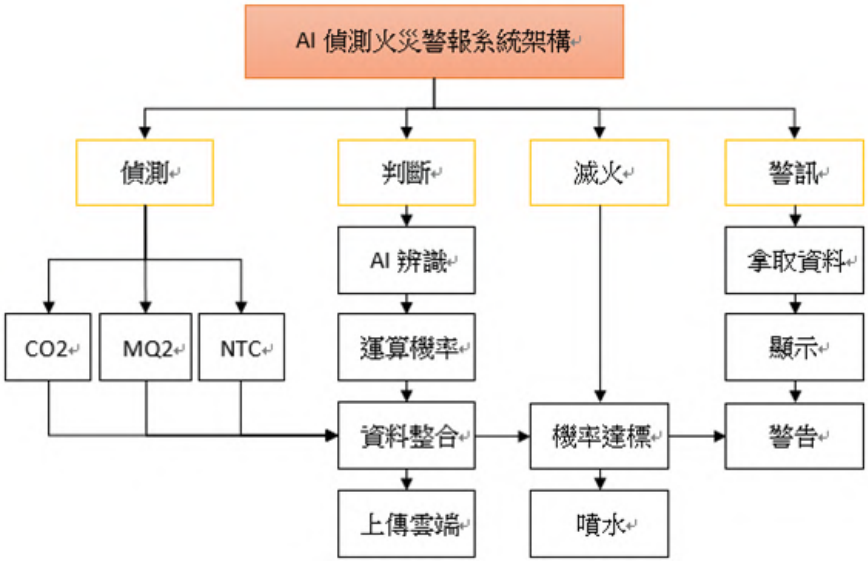


圖 4 AI 偵測火災警報系統架構圖

(五)建立硬體模型：我們使用厚紙板和感測器，組裝硬體模型如圖 5。

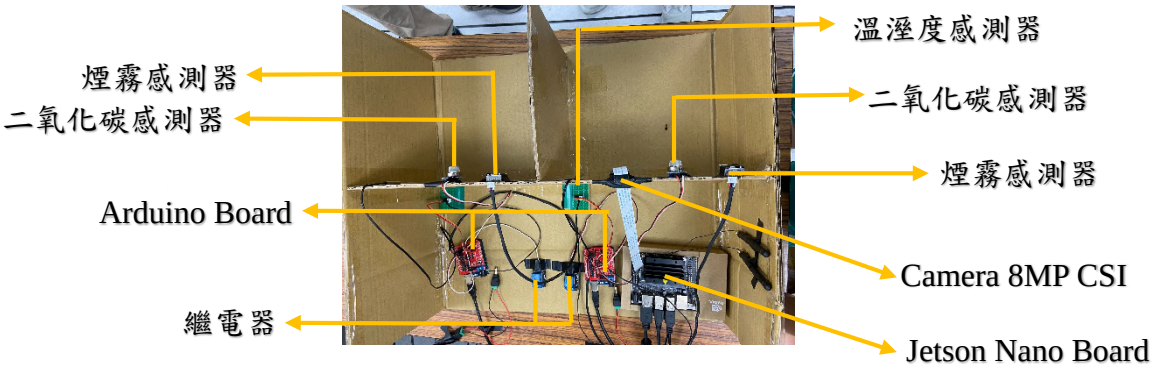


圖 5 硬體模型介紹

(六)程式設計與資料傳遞：

1. 視覺辨識演算法:我們使用的演算法是 InceptionV3，相較以前版本的 CNN，它是用一種叫做 Inception 的結構，所以經我們前測試過後就使用它作為訓練的演算法如下圖 6。
2. 遷移訓練:為了降低訓練時特徵提取時間，我們嘗試使用了遷移訓練方法，在第一次訓練好的模型關閉前幾個神經層，訓練後面判斷的部分，做第二次訓練，優化後面對照片的處理如下圖 6。

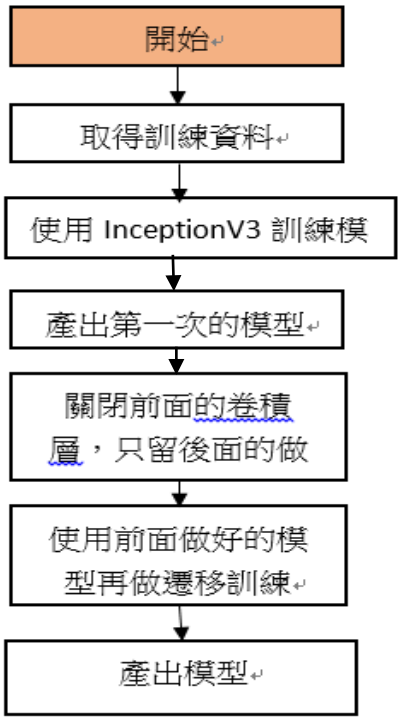


圖 6 模型訓練流程圖

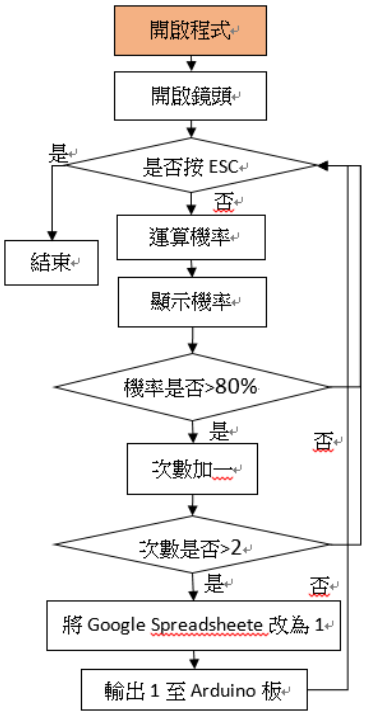


圖 7 視覺辨識邏輯流程圖

3. 視覺辨識邏輯流程:當發生火災的機率達到 trigger 條件(80%)，其間隔時間長短可以自訂，在一定秒數內有觸發超過一定次數(可以自訂)，將被系統認為有火災發生，會立即傳到用戶手機裡，本視覺流程圖，如上圖 7。本研究會透過

Google Spreadsheet 接收所有偵測到的資料，若發現有火災，會在火災的格子上會由 0 變成 1。

(七)本系統運作流程：

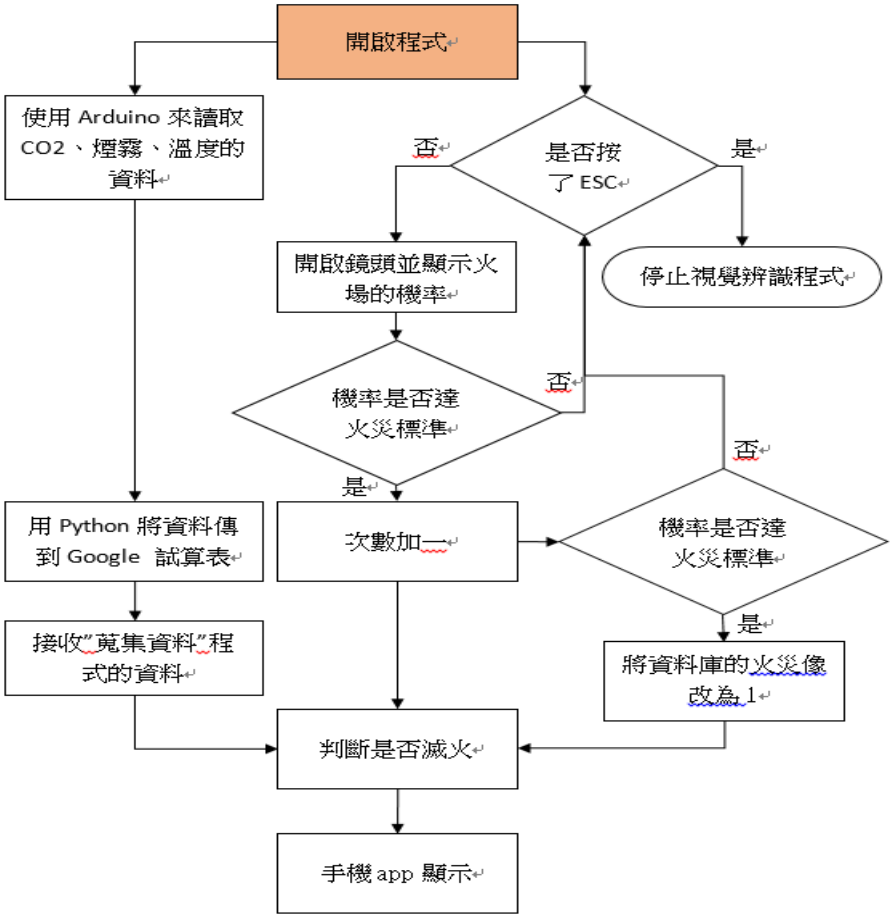


圖 8 系統運作流程圖

三、研究結果

使用線香產生的煙霧實驗 MQ2 及 CO2，火災的部分則使用火災影片，步驟如下：

(一)開啟程式以初始化:鏡頭拍攝到的畫面未有火災發生，如圖 9，故發生火災的機率只有 0.01%。而在 Google Spreadsheet 裡的 D2 格子顯示 0 如圖 11，因此手機 App 的火焰識別顯示”警戒”如圖 10。

圖 9 未有火災時的影像(平常情況)

圖 10 APP 顯示畫面

	A	B	C	D	E	F	G
1	co2-master	smoke-master	temprature-master	fire-master	co2-slave	smoke-slave	temprature-slave
2	961	0	24.83	0	853	0	21.44

圖 11 Google Spreadsheet 顯示畫面

(二)點燃線香並使用手機播放火焰的影片置於”客廳”：

圖 12 火災發生

圖 13 APP 顯示火災

	A	B	C	D	E	F	G
1	co2-master	smoke-master	temprature-master	fire-master	co2-slave	smoke-slave	temprature-slave
2	1056	0	24.33	1	853	0	21.44

圖 14 Google Spreadsheet 顯示畫面

多次交叉測試發現 CO2 濃度感測器附近點燃線香一段時間後， CO2 濃度增加到 1056，相對鏡頭拍攝到火焰的瞬間，視覺辨識就認為火災發生，機率高達 99.54% 如圖 12。Google Spreadsheet 的 D2 改為 1 如圖 14，因此手機 App 的火焰識別顯示”火災發生”如圖 13，這證明了視覺辨識比偵測器收到火災的資訊更即時。

(三)將點燃線香靠近客廳偵測器，偵測 CO2 及煙霧濃度

圖 15 偵測到煙霧及 CO2

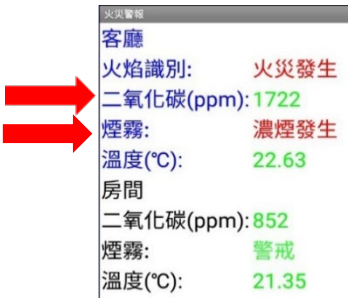


圖 16 App 顯示濃煙發生

	A	B	C	D	E	F	G
1	co2-master	smoke-master	temprature-master	fire-master	co2-slave	smoke-slave	temprature-slave
2	1722	48091	22.63	1	852	0	21.35

圖 17 Google Spreadsheet 顯示畫面

點燃線香後 CO2 及煙霧的濃度迅速上升，偵測到的煙霧濃度已超過系統設定如圖 15，因此 App 上顯示”濃煙發生”如圖 16，CO2 感測器數值以傳尚上 Google Spreadsheet 如圖 17，但時間較前述實驗有點延遲。

(四)客廳之繼電器已被觸發如圖 18，以表示觸發後續的滅火裝置。

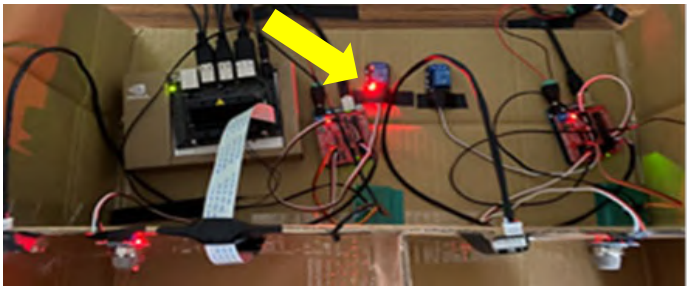


圖 18 客廳繼電器已被觸發

(一)遷移訓練對模型的影響

我們使用遷移訓練來二次加強模型。將前 249 個神經層關閉，只訓練 249 層之後的部分，嘗試調整全連階層(Dense)數量，發現在 dense1=200，dense2=50 時模型訓練精確度(acc)最高且損失值較低，測試結果如下圖 19，最終精確度都在 94.5% 以上，成功達到提升泛化能力和精確度，再透過 optimizer 函式最佳化，發現 Nadam，Adagrad 可以很快達到 98%以上的正確率如圖 20。

輸入層 1	輸入層 2	遷移訓練前				遷移訓練後			
dense1	dense2	loss	acc	val_loss	val_acc	loss	acc	val_loss	val_acc
100	24	0.031	0.9886	0.2449	0.949	0.0597	0.9823	0.0466	0.9898
100	50	0.0551	0.9814	0.1686	0.9592	0.0886	0.9782	0.0951	0.9796
200	100	0.0482	0.987	0.2655	0.9592	0.0652	0.981	0.0976	0.9796
200	50	0.017	0.9942	0.0877	0.9898	0.0857	0.9779	0.086	0.9796
500	200	0.0427	0.9879	0.1446	0.949	0.1219	0.9574	0.095	0.9694
1000	500	0.0233	0.992	1.3658	0.7245	0.0449	0.9903	0.1442	0.9388
1500	700	0.0466	0.9883	0.157	0.9592	0.09	0.9751	0.122	0.9694
2000	1000	0.0787	0.9754	0.1635	0.9388	0.0661	0.975	0.149	0.949
3000	1500	0.1397	0.9829	0.0448	0.9898	0.0905	0.972	0.1086	0.9796
4000	2000	0.0469	0.9818	0.2721	0.9388	0.0651	0.9828	0.1083	0.9592
1000	200	0.1768	0.9762	0.0203	0.9898	0.08	0.9744	0.1115	0.9592

圖 19 遷移訓練前後對比表

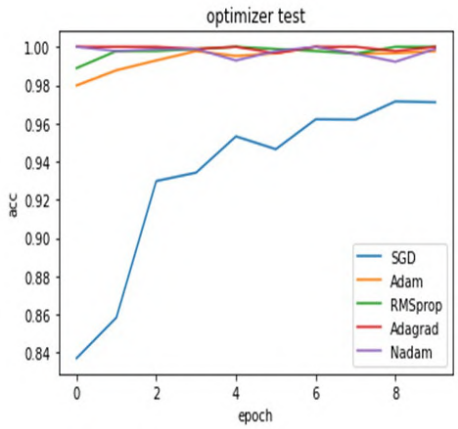


圖 20 不同演算法執行結果比較圖

四、結論

根據以上研究結果，證明本模型可以有效地利用 AI 和感測器來辨別火災，並在偵測到火災時立即發出警訊，讓使用者即時知道火警發生，以下是本系統特色：

(一)火災判斷迅速且正確率高：我們在視覺辨識上讓電腦針對火焰特徵做判斷，而非僅辨識顏色。在經過訓練後，其成功判斷機率能達 94.5%以上，若再經過 optimizer 函式優化後，就可以達到 98%以上的正確率。

(二)能即時傳資料至雲端並做後續處理：本研究裡的 NTC、MQ2、CO2 感測器數值透過 Jetson Nano 傳至雲端，再傳至手機 App 裡，讓使用者能即時看到最新資訊，但我們發現感測器偵測與上傳到網路有其速度限制，若能在確認的情形下，先行灑水建立防火牆或通知消防局，相信能減少更多的財物損失。

(三)可和現有系統結合：現今到處都有監視攝影機，而監視畫面大都透過網路，直接

存入資料庫。如果能分流監視攝影機的影像，再將其寫入資料庫，同時做出即時的視覺辨識，就可減少重新安裝的成本。

(四)本研究的模型容易移植，通用性高：本系統訓練使用常見的 Python 程式及套件，產生的模型檔大小約為 136~150MB，未來仍可以持續增加圖片繼續訓練以提高正確率，在使用上若不想重新訓練，也可直接載入模型檔案使用，非常方便。

(五)偵測鏡頭角度與死角問題：本研究使用 1 個鏡頭做偵測，未來可考慮使用多鏡頭在不同角度、位置做辨識。除了避免偵測死角外，也可嘗試建立不同特徵模型做比較分析。

2021全國高中職專題製作競賽

資電領域

壹、研究動機

由於火災頻繁的發生，造成死傷及財物損失慘重，所以我們想發展一套能智慧監控的裝置，若確認有火災發生即可立即通報與處理。正好近年來流行視覺辨識，所以我們想結合視覺辨識和原先的感測器，除了讓使用者能透過手機APP及時得知火災發生，並研究如何結合感測器數值作後續滅火處理。

貳、研究目的

一、研究以視覺辨識偵測火災發生的可行性。

二、研究在不同場域搭配各式感測器，發展火災偵測系統，並分析可能碰到之困難與解決。

三、探討本偵測系統可行之應用模式。

參、研究設備及器材

表1、研究器材一覽表

名稱	規格	數量	用途
Jetson Nano Board	Nano B01	1個	執行Python程式
Camera 8MP CSI	800萬畫素	1個	拍攝環境畫面
煙霧感測器	MQ-2	2個	偵測煙霧
二氧化碳感測器	DS-CO2-20	2個	偵測二氧化碳
溫溼度感測器	NTC	2個	偵測溫度
繼電器	5V高電位觸發	2個	控制幫浦

肆、研究過程或方法

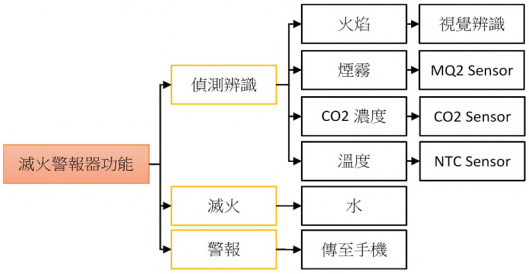
一、視覺辨識模型與運算的選擇

本組在訓練資料集中使用火災與正常畫面為各900張照片，而在驗證資料集中，火災與正常畫面各100張照片，將這些照片透過CNN演算法建立初步模型，最後再研究如何搭配不同的演算法，提高本系統正確率如圖1所示。

圖1　為火災視覺辨識訓練初步規劃

二、警報系統初步功能規劃

我們針對辨識火源的部分運用了二氧化碳、煙霧及溫度感測器，搭配攝影機視覺辨識，希望能透過以上這四種方式精確的判斷是否有火災，以降低誤判的機率。以下是滅火警報器初步規劃的功能圖：



三、擬定感測器模組並作前測

針對本研究所需的感測器分別做了功能前測如下圖，分別是MQ2(煙霧測試)，NTC負溫度係數熱敏電阻器(溫度測試)，DS-CO2-20(二氧化碳測試)：

圖2　為各感測器測試畫面

四、即時傳遞資料

我們希望能將蒐集到的數據自動傳送到雲端，同時方便讓使用者收到即時資訊。因此我們建立Google的雲端資料庫，使資料能同步到手機裡。以下是申請金鑰與寫入Google Spreadsheet的流程：

圖3　為申請與寫入Google Spreadsheet流程

五、火災偵測與辨識說明

如果只使用感測器判斷火災，其發出警報的時效令人懷疑。因此我們在火焰辨識上使用鏡頭作視覺辨識，只要空間裡一有火燒起來即可馬上辨識，系統預計在辨識的結果中連續觸發3次所設定的標準(張數、間隔時間可自訂)，即認為應該發生火警，隨即傳送警告訊息與滅火模組。本組希望能訓練出於各種場合皆可使用的AI模型，兼顧即時性和應用性。

伍、研究結果

一、系統功能架構圖：本研究所發展的系統架構如圖4所示。

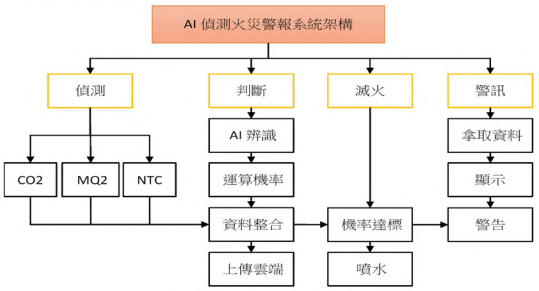
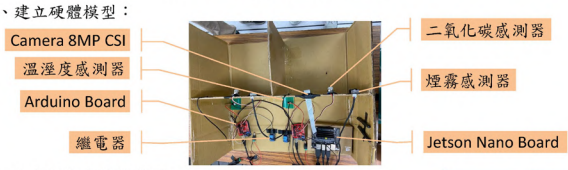


圖4　為AI偵測火災警報系統架構圖

二、建立硬體模型：



三、程式設計與資料傳遞：

1. AI模型訓練：

(1)視覺辨識演算法：

我們使用的演算法是InceptionV3，相較以前版本的CNN，它是用一種叫做Inception的結構，所以經本組前測過後就使用它作為訓練的演算法。

(2)遷移訓練：

為了讓成功率提高，我們使用了遷移訓練這種方法，在第一次訓練好的模型關閉前幾個神經層，訓練後面判斷的部分，做第二次訓練，優化後面對照片的處理。此外，本組發現在訓練模型階段，使用Jetson Nano搭配視覺辨識雖可完成模型，但是隨著運算層變多轉移到google colab運算效能更快。

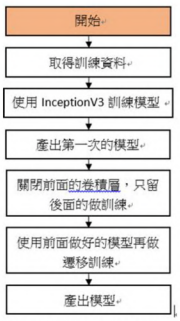
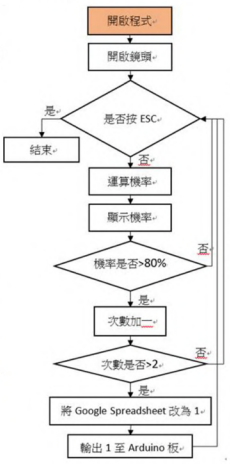


圖5　為模型訓練流程圖



2. 模型載入與執行：

當發生火災的機率大於我們設定的值(80%)，就達到trigger條件，其間隔時間長短可以自訂，當在一定秒數內有觸發超過一定次數(可以自訂)，則將被系統認為有火災發生，會立即透過網路傳輸到用戶手機裡顯示火災警報。

簡而言之，我們設定的各項數值(機率、秒數、觸發次數)並不是設定數值越高代表判斷火災越精準，而是要針對訓練模型的結果去設定合適的數值。本視覺辨識的流程如圖6所示。

圖6　為視覺辨識邏輯流程圖

本研究會透過Google Spreadsheet接收所有偵測到的資料，若發現有火災，在Google Spreadsheet火災的格子上會由0變成1，如圖7、8所示。

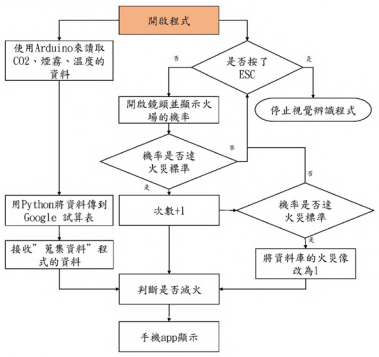
	A	B	C	D	E	F	G
1	co2-master	smoke-master	temprature-master	fire-master	co2-slave	smoke-slave	temprature-slave
2	708	51072	22.14	0	641	29386	22.54
3							

圖7　未有火災時數字為0

	A	B	C	D	E	F	G
1	co2-master	smoke-master	temprature-master	fire-master	co2-slave	smoke-slave	temprature-slave
2	714	1	21.94	1	670	0	21.25
3							

圖8　判斷有火災時數字為1

四、本系統運作流程



系統運作流程圖

五、實測結果

(一)使用線香產生的煙霧實驗MQ2及CO2，火災的部分則使用火災影片，步驟如圖9~11：

開啟程式以初始化

圖9 未有火災時的影像

圖10 APP顯示畫面

圖11 Google Spreadsheet顯示畫面

鏡頭拍攝到的畫面未有火災發生，故發生火災機率只有0.01%，Google Spreadsheet裡的D2格子顯示0，因此手機App的火焰識別顯示“警戒”。

(二)點燃線香並使用手機播放火焰的影片置於”客廳”，如圖12~14

圖12 火災發生

圖13 APP顯示火災發生

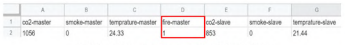


圖14 Google Spreadsheet顯示畫面

多次交叉測試發現，在二氧化碳濃度感測器附近點燃線香一段時間後， 二氧化碳濃度增加到1056，相對於鏡頭拍攝到火焰的瞬間，視覺辨識就認為火災發生，機率高達99.54%，Google Spreadsheet的D2改為1，因此手機App的火焰識別顯示” 火災發生”，這證明了視覺辨識比偵測器收到火災的資訊更加即時。

(三)將點燃線香靠近客廳偵測器，偵測二氧化碳及煙霧濃度，如圖15~17

圖15 偵測到煙霧及CO2

圖16 App顯示濃煙發生

圖17 Google Spreadsheet顯示畫面

點燃線香後二氧化碳及煙霧的濃度迅速上升，煙霧的濃度已達標，因此App上顯示”濃煙發生”，但時間仍然較前述實驗有點延遲。

(四)觀察客廳之繼電器是否被觸發，如圖18

圖18 客廳繼電器已被觸發

當客廳場域被判斷為火災時，因此客廳的繼電器被觸發，灑水系統啟動，而房間因為沒有偵測到煙霧，因此繼電器未被觸發。

(五)將點燃的線香靠近房間偵測器，偵測二氧化碳及煙霧濃度，如圖19~21：

圖19 App顯示濃煙發生

圖20 Google Spreadsheet顯示畫面

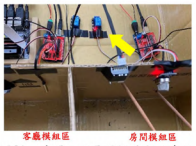


圖21 房間之繼電器觸發顯示畫面

由於房間少了鏡頭，所以當煙霧及二氧化碳濃度都達標準，Arduino就會啟動繼電器，視同啟動警報系統及灑水迴路。

由於一般大樓灑水迴路有其相關法規規定，本實驗也是互為獨立，然未來若火警判斷的正確率提高時，可以嘗試當確定火警發生時，在鄰近灑水迴路取得控制權，先行灑水建出一條防火牆，以降低財物損失，這在程式上是完全可行的。

陸、討論

一、遷移訓練對模型的影響

我們希望此模型能廣泛應用在各個場所，所以本組將前249個神經層關閉，只訓練249層之後的部分做嘗試，測試結果如表2：

表2、遷移訓練前後對比

輸入層1	輸入層2	遷移訓練前				遷移訓練後			
dense1	dense2	loss	acc	val_loss	val_acc	loss	acc	val_loss	val_acc
100	24	0.031	0.9886	0.2449	0.949	0.0597	0.9823	0.0466	0.9898
100	50	0.0551	0.9814	0.1686	0.9592	0.0886	0.9782	0.0951	0.9796
200	100	0.0482	0.987	0.2655	0.9592	0.0652	0.981	0.0976	0.9796
200	50	0.017	0.9942	0.0877	0.9898	0.0857	0.9779	0.086	0.9796
500	200	0.0427	0.9879	0.1446	0.949	0.1219	0.9574	0.095	0.9694
1000	500	0.0233	0.992	1.3658	0.7245	0.0449	0.9903	0.1442	0.9388
1500	700	0.0468	0.9883	0.157	0.9592	0.09	0.9751	0.122	0.9694
2000	1000	0.0787	0.9754	0.1635	0.9388	0.0661	0.975	0.149	0.949
3000	1500	0.1397	0.9829	0.0448	0.9898	0.0905	0.972	0.1086	0.9796
4000	2000	0.0469	0.9818	0.2721	0.9388	0.0651	0.9828	0.1083	0.9592
1000	200	0.1768	0.9762	0.0203	0.9898	0.08	0.9744	0.1115	0.9592

根據上表顯示，經過Transfer Learning後大部分精確度呈現上升狀況，可能是原先的模型已過度擬合，但經遷移訓練後改善了這個缺點，最終精確度都在94.5%以上，成功達到提升泛化能力和精確度。

本研究發現在添加遷移訓練後，能配合不同場所成功判斷的機率極高，若未來要移植至監視系統也不用擔心有場地的疑慮。

二、嘗試各種優化器演算法將模型最佳化

本組嘗試自行撰寫程式，將不同演算法透過運算後全部繪製到同一個圖上，learning_rate從0.001到0.01做測試，透過紀錄其history然後將其繪製如圖22，比較不同演算法的差異性。

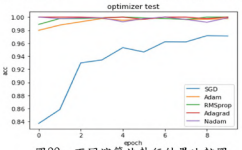


圖22 不同演算法執行結果比較圖

透過比較，本系統在learning_rate=0.001時，我們發現Nadam，Adagrad可以很快達到0.98以上的正確率，經過epoch=40次以上時，各演算法都可以達到穩定的結果，適合本模型之所需，而未來若想要移植到不同環境重新編譯，也可以透過此法快速找出最佳的演算法。

柒、結論

本模型可以有效地利用AI和感測器來辨別火災，讓使用者即時知道火警發生，希望能在防災上可以做到一點幫助。本作品優點如下：

一、火災判斷迅速且正確率高：我們在視覺辨識上讓電腦針對火焰特徵做判斷，而非僅辨識顏色，經過訓練後，其成功判斷機率都能達94.5%以上，若再經optimizer函式優化，可以達到98%以上的正確率。

二、能即時傳送資料至雲端並做後續處理：本研究裡的感測器數值透過Jetson Nano上傳至雲端，再傳至手機App裡，讓使用者能即時看到最新資訊，此外若能在確認火警的情形下，先行灑水建立防火牆或通知消防局，相信更能減少民眾的財物損失。

三、可和現有系統結合：現今監視攝影機的監視畫面大都透過網路直接存入資料庫，如果能分流監視器的影像，同時做即時的視覺辨識，則可以減少重新安裝成本。

四、本研究的模型容易移植，通用性高：本系統訓練產生的模型檔大小約為136~150MB，未來仍可以持續增加圖片繼續訓練以提高正確率，也可以直接載入模型檔使用，非常方便。