

電腦視覺與雲端實務

期末報告

- 自定義資料集之物件偵測與辨識 -

李峻瑋 **10627131**

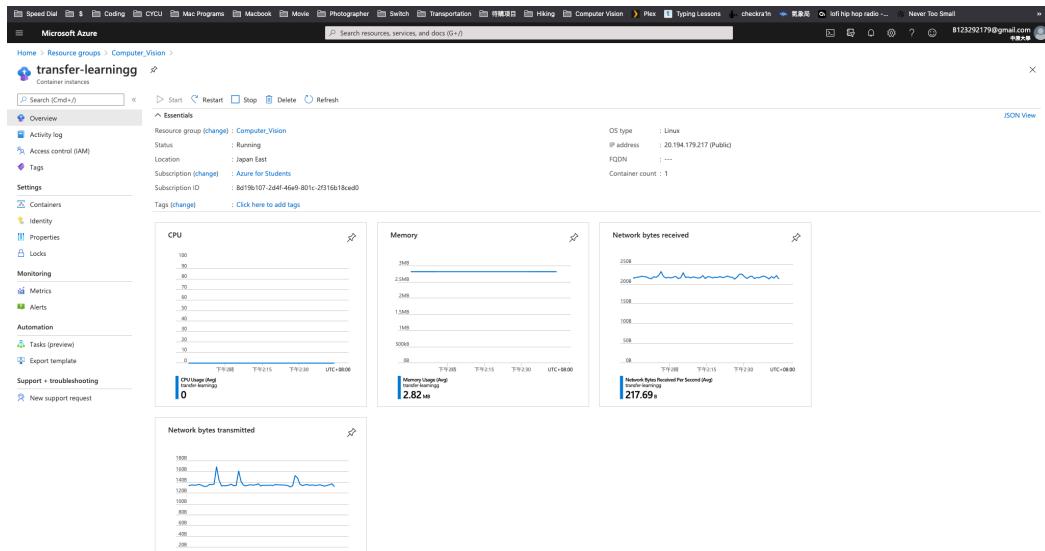
林冠良 **10627130**

陳凱 **10627133**

Scenario

WHAT ?

利用 Azure 架設網站，並且使用 AI 技術辨識且分類物體。



Webcam KNN Training

Model loaded.

[Enable Webcam](#) [Close camera](#) [Predict](#)

[Add Class](#) [Reset](#)

Model loaded.

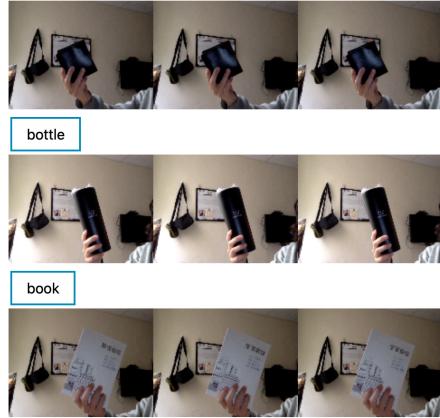
[Enable Webcam](#) [Close camera](#) [Predict](#)



prediction: wallet, probability: 100%

[Add Class](#) [Reset](#)

wallet



HOW ?

利用 OpenCV.js 與 KNN-Classifier，得以讓使用者開啟視訊鏡頭並且記錄要分類之物品。輸入完畢後即可取特徵點與其分類，並且只要將分類完的物品移至畫面中，便可以輕鬆地辨識出其物體之分類，且可自行增加物品分類以及定義物品標籤。再將網站利用 Azure 以及 Docker 之技術，將網站上架至網路上，讓使用者可以輕鬆地在網路上使用上述功能。

Techniques

Programming Language:

HTML, CSS, Javascript.

The screenshot shows a code editor with three tabs: index.html, transferlearning.js, and Dockerfile. The index.html file contains HTML code for a Webcam KNN Training application. The transferlearning.js file contains JavaScript code for predicting classes from a webcam video. The Dockerfile defines the build process for a Docker container.

```
index.html
transferlearning.js
Dockerfile
```

```
You, 3 days ago | 1 author (You)
1 <!DOCTYPE html>
2 <html>
3
4   <head>
5     <meta charset="utf-8">
6     <title>Webcam KNN Training</title>
7     <!-- Load the latest version of TensorFlow.js -->
8     <script src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs/dist/tf.min.js"></script>
9     <script src="https://cdn.jsdelivr.net/npm/@tensorflow-models/mobilenet"></script>
10    <link rel="stylesheet" href="/css/style.css">
11  </head>
12
13 <body>
14   You, 3 days ago + js and css unsolved
15   <h1>Webcam KNN Training</h1>
16   <h2>Model loading...</h2>
17   <div>
18     <button id="webcamButton" class="button button3">Enable Webcam</button>
19     <button id="disableWebcamButton" class="button button3" disabled>Close camera</button>
20     <button id="predict" class="button button4">Predict</button>
21   </div>
22   <div id="liveView">
23     <video id="webcam" autoplay width="320" height="240"></video>
24     <p id="result" style="font-size:16px; text-decoration:underline"></p>
25   </div>
26   <button id="addBtn" class="button button1">Add Class</button>
27   <button id="reset" class="button button5">Reset</button>
28   <div id="addDiv"></div>
29   <div id="buttonDiv"></div>
30   <!-- Load js after the content of the page --->
31   <script src="https://cdn.jsdelivr.net/npm/@tensorflow-models/knn-classifier"></script>
32   <script src="/js/transferlearning.js"></script>
33
34 </body>
```

```
function predictCam() {
  if (classifier.getNumClasses() > 0) {
    // Get the activation from mobilenet from the webcam.
    const activation = model.infer(video, true);
    // Get the most likely class and confidence from the classifier module.
    classifier.predictClass(activation).then(result) => {
      document.querySelector("#result").innerHTML =
        `prediction: ${buttonNameArray[result.label]}, probability: ${result.confidences[0]}`;
      console.log(`predict${result}`);
    }
    window.requestAnimationFrame(predictCam);
  }
}

function addButton(event) {
  var buttonName = prompt("Please enter new class name:");
  if (buttonName != null) {
    var newBtn = document.createElement('button');
    document.getElementById('buttonDiv').appendChild(newBtn);
    newBtn.innerHTML = buttonName;
    newBtn.className = 'button button2';
    newBtn.id = 'buttonName' + buttonNameIndex.toString();
    newBtn.addEventListener('click', (e) => addClass(e, tempIndex));
    buttonNameArray[buttonNameIndex] = buttonName;
  }
}

var newDiv = document.createElement('div');
newDiv.id = 'div' + buttonNameIndex.toString();
var tempIndex = buttonNameIndex;
document.getElementById('buttonDiv').appendChild(newDiv);
buttonNameIndex += 1;
// div
var newBtn = document.createElement('p');
// useless shit
} // if

document.body.appendChild(newBtn);
}

function addClass(event, className) {
  var canvas = document.createElement('canvas');
```

The screenshot shows a code editor with five tabs: index.html, transferlearning.js, nginx.conf, Dockerfile, and another index.html. The index.html files show configuration for a web server. The transferlearning.js file is identical to the one in the previous screenshot. The nginx.conf file defines a server block for port 80 and 443, including SSL certificates. The Dockerfile defines the build process for a Docker container.

```
index.html
transferlearning.js
nginx.conf
Dockerfile
index.html
```

```
You, 3 days ago | 1 author (You)
1 server {
2   listen 80;
3   server_name localhost;
4
5   location / {
6     root /usr/share/nginx/html;
7     index index.html index.htm;
8   }
9 }
10 server {
11   listen 443 ssl;
12
13   server_name localhost;
14
15   # 憑證與金鑰的路徑
16   ssl_certificate /etc/nginx/ssl.csr;
17   ssl_certificate_key /etc/nginx/ssl.key;
18
19   location / {
20     root /usr/share/nginx/html;
21     index index.html index.htm;
22   }
23 }
```

```
FROM nginx
COPY index.html /usr/share/nginx/html
COPY css/* /usr/share/nginx/html/css/
COPY js/* /usr/share/nginx/html/js/
# Remove the default nginx.conf
RUN rm /etc/nginx/conf.d/default.conf
# Replace with our own nginx.conf
COPY nginx.conf /etc/nginx/conf.d/
COPY ssl.csr /etc/nginx/ssl.csr
COPY ssl.key /etc/nginx/ssl.key
EXPOSE 443
```

Tools:

利用 OpenCV.js 與 KNN-Classifier 去進行分類與辨識，在自行利用 SSL 生成自己的 Certificate，再將 Application 利用 Docker 配合 Nginx 之技術架至網路上 Azure 的 Server，讓使用者可以直接在網路上我們所實作的功能。

Cloud Features

硬體不受限制：

使用者在使用網站時，背後運行是使用當初建立container instance時所選取的CPU大小，能夠令使用者較不受限於本地端的硬體條件

Change container ... ×

Configure the resource requirements for your container. The available values are based on region, OS type, and networking options.
[Learn more about resource requirements in ACI](#)

Number of CPU cores * ⓘ
1
1-4

Memory (GiB) * ⓘ
1.5
0.5-16

GPU type (Preview) * ⓘ
None

任何人都能取用：

藉由azure上的container建立webserver擁有固定的ip位置另任何人都能連到此IP並使用網站

OS type	: Linux
IP address	20.194.179.217 (Public)
FQDN	: ---
Container count	: 1

Lessons learned

概要：

- 在前半學期主要學到了如何使用 Javascript 配合 HTML 寫網頁，也學到了很多有關灰階, Filter 等對於圖像進行處理的用法，之後再加入各種model去對於圖像做出辨識以及分類。在課程後半段則是有關雲端的部分，一開始在學習在虛擬機中建立 images 並且放入 docker 運行，後來才分別在 Azure 利用 VM 與 Container 建立 Webserver。
- 最後是期末 Project，一開始上雲端時，Webcam 因為 SSL 簽章無法通過，無法開啟 Webcam，後來上課時有提到可以建立自己的簽章，所以使用 OpenSSL 建立自己的簽章令 Webcam 可以使用，並且成功的架設網站秀出我們自己實作的東西，十分的有成就感。
- 而這樣的課程內容在以往資工系的必修課程中並不常見，所以很多東西都是從頭開始學起，雖然一開始矇矇懂懂，但實作到最後就能真正了解網頁以及雲端背後運行的原理，在當中也學到很多與以往不一樣東西。

References

Nginx & Docker & SSL:

- <https://www.maxlist.xyz/2020/01/19/docker-nginx/>
- <https://medium.com/starbugs/web-server-nginx-2-bc41c6268646>
- <https://blog.gtwang.org/linux/nginx-create-and-install-ssl-certificate-on-ubuntu-linux/>