國立清華大學資訊工程系109學年度上學期專題報告

專題名稱	智慧步態辨識				
參加競賽或計畫	☑ 參加對外競賽	☐ 參與其他計畫	☐無參加對外競賽或任何計畫		
學號	106062116	106062232			
姓名	黃晨	蔡登瑞			

摘要

Human activity recognition (HAR)是許多人研究的主題，能透過智慧型手機、sensor 的資料，完成動作的辨識。而我們專題的內容，是使用六軸加速器晶片，收集人在做各種動作時的加速度、角速度數據，建立自己的 dataset，並以 machine learning 的 CNN model 進行 training、testing。最終讓人只要穿上我們的穿戴式裝置，就能夠完成高準確率的動作辨識。

中華民國 109 年 10 月

1、專題研究動機與目的

當初在選擇題目時，由於實驗室的學長有在研究穿戴式裝置的藍芽通訊，我們在上網搜尋後， 發現竟然只需要收集很簡單的數據，如加速度、角速度，就可以通過機器學習，判斷一個人的動作型態，這引起了我們的興趣。因此，我們便以學長的裝置為基礎，加上 machine learning 的 classification model，希望讓人只需要配戴一個小小的裝置，就可以追蹤他的動作。這個結果或許可以應用在醫療照護，或是運動監控上。

2、現有相關研究概況及比較

前面提到，HAR是一個熱門的研究主題，已經有許多論文都研究過這個主題。其中包括影像辨識、sensor 辨識，而在我們研究的 sensor 辨識中，大部分都採用 machine learning 來做分類。然而，大多的研究都只專注於對網路上的 data set（e.g. UCI data set） 進行分類的優化，意即只提高準確率，沒有實際嘗試運用這個成果。

為了實際確定這個專題在生活上的可行性，我們使用自己的穿戴式裝置，收集自己的 data set，在經過 training 之後，我們設計的 CNN model 可以完成對受試者即時的動作辨識。

3、設計原理

受試者穿上的穿戴式裝置之後，六軸加速器會將加速度、角加速度的數據以固定頻率(50HZ)透過 arduino 以及藍芽模組傳給 Server，

圖 1: 穿戴式裝置外觀

Sequence Number	Ax	Ay	Az	Gx	Gy	Gz	CLK	Reserved	!
1 byte	2 bytes each						1 byte	5 bytes	1 byte

圖 2: 封包格式

到 server 端之後，資料會經過 preprocessing 之後，由事先訓練好的 CNN model 進行分類，並呈現結果。為什麼 CNN model 能夠完成分類問題呢？這是因為在每一層 CNN layer 中，有許多 filter，每個 filter 在經過各自的運算後，可以提取出不同的 features，也就是不同動作產生的數據會有獨特的波型。

4、研究方法與步驟

A、 Preprocessing

實驗蒐集到的原始資料是為一連串 X,Y,Z 三軸加速度與角加速度，為了將其轉換為 CNN model可以接收的 pattern，以及處理一些雜訊、掉封包的問題，我們需要進行 preprocessing。

首先，我們使用多項式插值法來解決封包掉落的狀況（圖12、13）。接著，我們分別使用 Median Filter 與 Low Pass Butterworth Filter 來解決雜訊干擾的狀況（圖13、14、15）。然後，使用長度為 2.56 秒、50% overlap 的 sliding window 將資料切割。最後再做 labling，原始資料就成為數個能夠透過 CNN model進行分類的 samples。

```
# interpolate
data_length = acce_length
if acce_length > gyro_length:
    gyro_data = Interpolate(gyro_data, gyro_length, acce_length, "cubic")
elif acce_length < gyro_length:
    acce_data = Interpolate(acce_data, acce_length, gyro_length, "cubic")
    data_length = gyro_length
```

圖12

```
def Interpolate(data, old_length, new_length, kind):
    old_samples=np.linspace(0, old_length, old_length)
    new_samples=np.linspace(0, old_length, new_length)

    fx = interpolate.interp1d(old_samples, data[0], kind=kind)
    fy = interpolate.interp1d(old_samples, data[1], kind=kind)
    fz = interpolate.interp1d(old_samples, data[2], kind=kind)

    return fx(new_samples), fy(new_samples), fz(new_samples)
```

圖13

```
def MF(data):
    return signal.medfilt(data, 3)

def LPBWF(cutoff, freq, data):
    b,a = signal.butter(3, 2*cutoff/freq, btype='lowpass', analog=False, output='ba')
    return signal.filtfilt(b, a, data)
```

圖14

```
# filt
_MF_acce = [MF(acce_data[0]), MF(acce_data[1]), MF(acce_data[2])]
_MF_gyro = [MF(gyro_data[0]), MF(gyro_data[1]), MF(gyro_data[2])]

_LPBWF_acce = [LPBWF(cutoff, freq, _MF_acce[0]),
               LPBWF(cutoff, freq, _MF_acce[1]),
               LPBWF(cutoff, freq, _MF_acce[2])]
_LPBWF_G    = [LPBWF(     g, freq, _MF_acce[0]),
               LPBWF(     g, freq, _MF_acce[1]),
               LPBWF(     g, freq, _MF_acce[2])]
_LPBWF_gyro = [LPBWF(cutoff, freq, _MF_gyro[0]),
               LPBWF(cutoff, freq, _MF_gyro[1]),
               LPBWF(cutoff, freq, _MF_gyro[2])]
```

圖15

B、 Training model

在專題初期，我們使用 UCI dataset 的資料訓練模型。確定模型大概的方向後，再用我們自己收集的資料訓練，並進行調整。其中包括 filter number、kernel size、validation set 等等的參數，也可以加入許多不同的 layer。在找到 testing 準確率最好的模型後，進行實際測試。

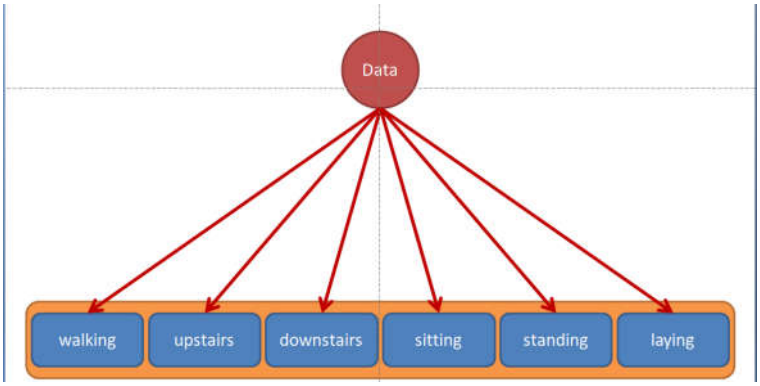
```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_13 (Conv2D)           (None, 98, 2, 128)        35840
_________________________________________________________________
activation_13 (Activation)   (None, 98, 2, 128)        0
_________________________________________________________________
batch_normalization_13 (Batc (None, 98, 2, 128)        392
_________________________________________________________________
conv2d_14 (Conv2D)           (None, 68, 1, 128)        1015936
_________________________________________________________________
activation_14 (Activation)   (None, 68, 1, 128)        0
_________________________________________________________________
batch_normalization_14 (Batc (None, 68, 1, 128)        272
_________________________________________________________________
max_pooling2d_9 (MaxPooling2 (None, 34, 1, 128)        0
_________________________________________________________________
conv2d_15 (Conv2D)           (None, 20, 1, 64)         122944
_________________________________________________________________
activation_15 (Activation)   (None, 20, 1, 64)         0
_________________________________________________________________
batch_normalization_15 (Batc (None, 20, 1, 64)         80
_________________________________________________________________
max_pooling2d_10 (MaxPooling (None, 10, 1, 64)         0
_________________________________________________________________
dropout_5 (Dropout)          (None, 10, 1, 64)         0
_________________________________________________________________
flatten_5 (Flatten)          (None, 640)               0
_________________________________________________________________
dense_5 (Dense)              (None, 2)                 1282
=================================================================
Total params: 1,176,746
Trainable params: 1,176,374
Non-trainable params: 372
```

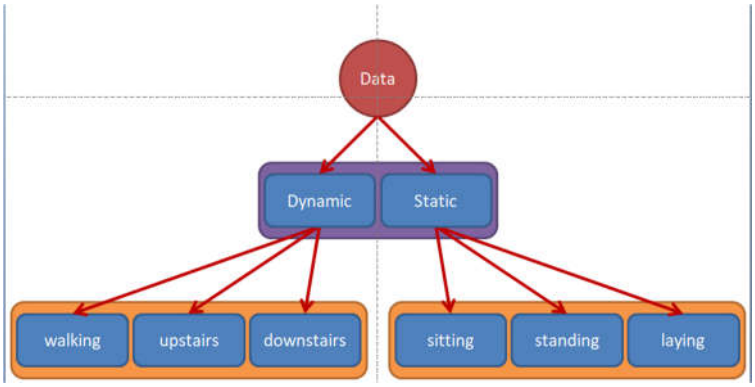
一開始我們利用UCI dataset訓練模型後進行實際測試，然而結果準確率卻是極低，我們認為這是因為UCI dataset的資料與我們收集的資料，兩者裝置位

置的不同所造成的，前者裝置是穿戴在測試者腹部前方，而我們的裝置則是穿戴在右腳腳踝上，兩者除了XYZ三軸方向不同之外，其動作軌跡也不同。

因此我們改成自行收集資料來訓練模型，再進行實際測試。其結果很明顯比利用UCI dataset訓練的模型來的好。但總體上來看，我們認為我們可以做得再好一點。

接著我們改善了訓練模型。原本僅使用一個模型來判別六種動作，我們想到這六種動作可以先細分成動態與靜態，再去判別是何種動作。因為就六軸資料上來看，動態與靜態的差異非常大，如果僅僅使用一個模型來判別全部動作，靜態動作很可能因為些許的干擾而被誤判為動態動作。因此我們改而先用一個模型來判別動作是動態或靜態，再分別用一個模型來判別是何種動作。





改善過後的模型在實際測試時，準確率極高，僅在動作間的變換時會產生些許誤判，但我們認為這是無可避免的。由於資料收集人員為我們兩位專題組員，生怕模型會過度擬和我們的動作軌跡，因此也有請數名同學來做實際測試，其結果準確率依然極高。

5、系統實現與實驗

A、系統實現

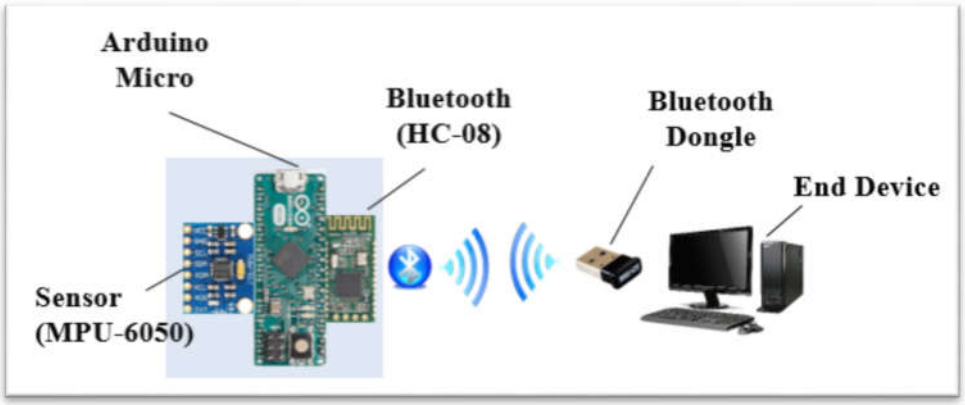


圖 16: System architechture

圖16為我們的系統結構圖。由 Arduino Micro 連接一個六軸加速器以及一個藍芽通訊模組，藉由藍芽通訊的方式，以50HZ的速度傳送封包給電腦，由電腦或著樹莓派的 VMWare 接收。

B、實驗

（1）收集資料

我們穿上穿戴式裝置後，分別進行六種動作: 走路、上樓、下樓、坐、站以及躺。經過 preprocessing之後，產生每種動作各約 500筆 samples。

（2）Training/Testing

不斷的修改 model，使其對 testing set 的預測準確率能夠至少達到 95% 以上。

```
Device: Arduino (First layer)
346/346 [==============================] - 1s 2ms/step
Accuracy:  0.9971098303794861

Predict:
```

Pred True	Dynamic	Static
Dynamic	234	1
Static	0	111

```
Device: Arduino (Second layer-Static)
130/130 [==============================] - 0s 2ms/step
Accuracy:  0.9846153855323792

Predict:
```

Pred True	LAYING	SITTING	STANDING
LAYING	42	0	0
SITTING	0	30	0
STANDING	0	2	56


```
Device: Arduino (Second layer-Static)
130/130 [==============================] - 0s 2ms/step
Accuracy:  0.9846153855323792

Predict:
```

Pred True	LAYING	SITTING	STANDING
LAYING	42	0	0
SITTING	0	30	0
STANDING	0	2	56

（3）及時預測

受試者穿上裝置後，將資料透過藍芽傳送到電腦的程式，並利用 CNN model 進行動作的即時預測。經測試，雖然在動作切換的時候會不準確，但是只要長時間持續單一動作，預測準確率約有9成。

6、團隊合作方式

A、Preprocessing: 蔡登瑞

B、ML model design: 蔡登瑞、黃晨

C、資料收集: 蔡登瑞、黃晨

7、效能評估與成果

我們設計的 CNN model，在對於自己蒐集的資料進行 training、testing時，不論是對於 training set、testing set的預測準確率都能夠達到 95%以上。而在即時預測的部分，只要持續維持一種動作，則準確率大約有 9成。

8、結論

雖然當初在選擇這個題目時，沒有考量到實用性，但是我們查詢資料後發現，對於動作辨識的技術可以應用於醫療照護以及生活管理上。比如設計成給老人的穿戴式裝置，搭配定位系統，如果老人在家中跌倒，可以透過地點以及動作辨識及早發現。雖然我們只完成了六種動作的辨識，但是相信如果這項技術發展成熟，能夠在生活上發揮一定的用處。