AIOT Final Project

組員名單

學號	姓名	分工內容
M11152025	陳彥合	實作並還原論文
M11115021	蘇峻緯	讀論文及分析實作結果，並為論文下定論
M11115015	廖唯任	讀論文及分析實作結果，並為論文下定論
M11115043	何亮牖	用台灣及世界的資料集應用在論文上

論文統整

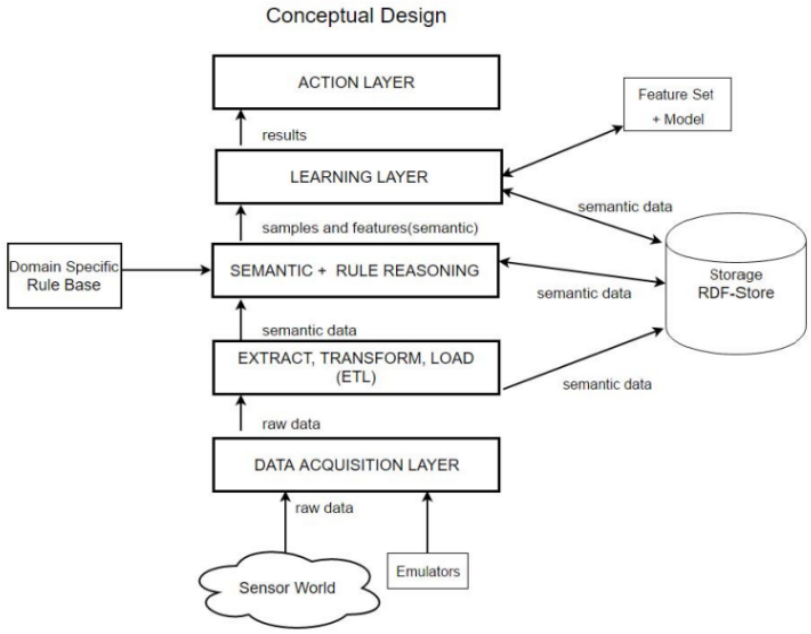
介紹 ：

提出了一個天氣分群模型實作在我們的IOT大數據分析框架，提供了每層的實作細節，其他的IOT Data分析問題也可以被這個框架以相似的方法來處理。這個論文用於天氣分群後得到的結果還可以用於天氣感測器的異常檢測，透過分群後的結果及感測器之間的距離來推測是否有感測器異常的問題。

論文框架 ：

這篇論文的框架主要分成五層，分別是數據採集層、ETL層、語意規則推理層、學習層、動作層，以下分別介紹各層的功能。

- 數據採集層 ： 負責從感測器採集數據，或者數據集可以批量加載到框架中。該層不負責解析或評估數據。數據採集層獲取數據後，將數據傳送到ETL(extract, transform, load)層。
- ETL層 ： 解析數據，生成RDF（Resource Description Framework）格式的語義註釋數據。然後將RDF格式的文件用於語義規則推理，然後將數據用於學習層。
- 語意規則推理層 ： 根據規則處理RDF數據並產生推斷數據。包含生成的物聯網數據的CSV（逗號分隔值）文件被傳輸到學習層。
- 學習層 ： 對輸入數據進行預處理，提取相關特徵並應用機器學習方法。然後機器學習算法在分佈式計算機服務器中並行運行，以更快的方式處理大數據。
- 動作層 ： 負責評估學習層產生的結果，並根據分析結果做出相應的行為。



RDF(Resource Description Framework)資料集介紹 ：

- LinkedSensorData : 該資料集中主要用來儲存感測器種類及感測器所屬位置資訊，如下圖所示，該感測器的種類為3CLO3，高度為20，經度及緯度分別為46.22和-124.00

```
sens-obs:point_3CLO3 a wgs84:Point ;
        wgs84:alt "20"^^xsd:float ;
        wgs84:lat "46.22"^^xsd:float ;
        wgs84:long "-124.00"^^xsd:float .
```

- LinkedObservationData : 該資料集中主要用來儲存感測器種類、感測器測得的值及當下紀錄的時間，如下圖所示，該感測器的種類為3CLO3，時間為2005/8/23 17:20，紀錄的值為300.0

```
sens-obs:
        MeasureData_WindSpeed_3CLO32005823_172000
        a om-owl:MeasureData;
        om-owl:floatValue "300.0"^^xsd:float ;
        om-owl:uom weather:degrees .
```

- 在語意規則推理層中會從RDF格式的資料集提取我們所需的資料並存成CSV檔，以便之後在學習層使用。

實際論文實作框架 ：

此篇論文的語意規則推理層及動作層都還在開發中，故這邊我們只使用資料採集層、ETL層及學習層，並且我們將語意規則推理層跟ETL層合併，將特徵轉成CSV格式的行為和ETL層一起做。

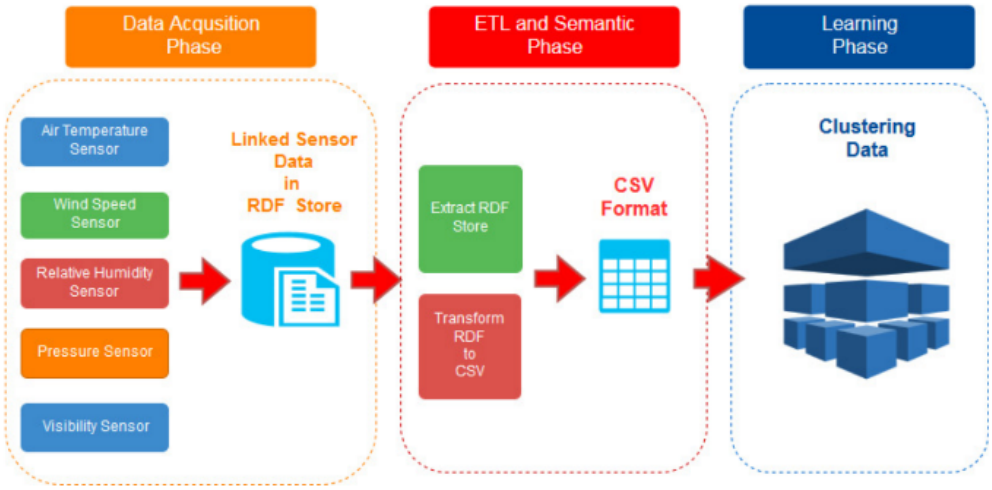


Fig. 2. Use-case Scenario of our proposed IoT Framework

學習層使用演算法 ：

這篇論文主要使用K-Means演算法來做分群，K-Means為一種非監督式演算法，故在這邊除了做天氣分群外，還可以用來做感測器異常檢測，這是非監督式演算法常被使用的行為。

實作論文

在實作論文方法的過程中，由於我們找不到RDF的資料集，且無法真正地去部署多個感測器於合適的場域，因此我們選用論文釋出程式碼中提供的CSV格式資料集，進行實作及還原。

在參考完作者於論文中提及的演算法後，我們將實作三個步驟：讀取資料、資料前處理、分群、作圖，最後將分群的結果與作者實作之結果做比較。此外，我們的實作也針對論文釋出的程式碼進行改良。

原論文釋出的程式碼問題

1. Hard Code
2. Readability
3. Scalability

舉例來說，像是K-Means演算法的K值，我們若要修改的話，得在程式碼中修改，若要讀取其他資料集的話，也得到程式碼中修改等等問題，而這樣子的情況大幅影響我們實驗、使用的效率，因此我們還原實作時，也會特別針對這些問題做修正。

使用工具及套件

- Numpy
- Pandas
- Scikit Learn

- Matplotlib
- Pylab
- Argparse

實作流程

1. Main Function

```python
if __name__=='__main__':
    args = parameter_parser()
    data = ReadData(args.datapath)

    feature=["AirTemp", "RelativeHumidity", "WindSpeed"]

    data = Preprocessing(data,feature)
    data = Clustering(int(args.k), data)
    Plot(data)
```

將每個步驟包裝起來，使得整份程式更容易讀懂

2. Args Parsing

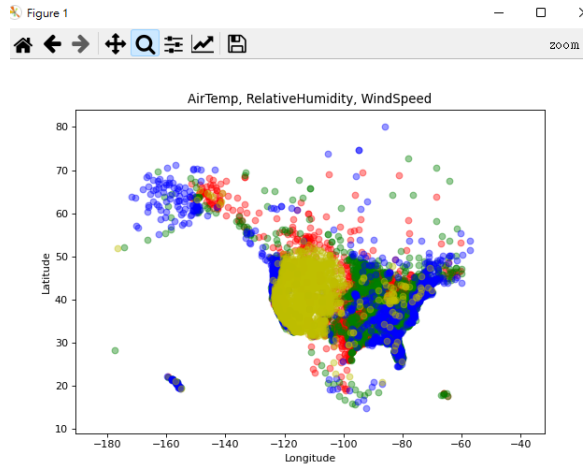
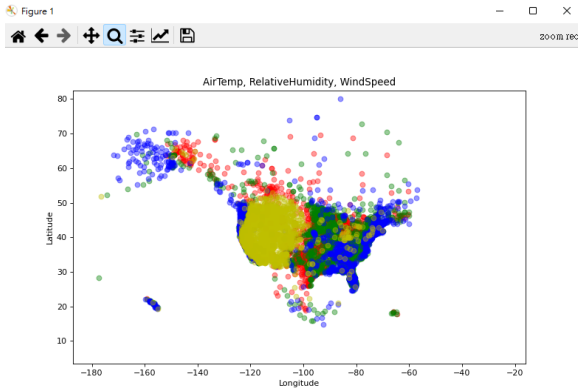
```python
def parameter_parser():
    '''
    for parsing args (cmd)
    '''
    parser = argparse.ArgumentParser(description="Run IoTSensorClustering.")

    parser.add_argument('-datapath',
                        nargs='?',
                        default='SensorsDataSet/28_1800.csv',
                        help='input dataset (.csv) .')

    parser.add_argument('-k',
                        nargs='?',
                        default=4,
                        help='input k (num of cluster) .')

    return parser.parse_args()
```

透過Args Parsing來調整參數，使得我們的實驗更加容易

3. Data Reading
   使用Pandas Dataframe讀取CSV

4. Preprocessing
   將每個sensor的資料統整，並選取溫度、風速、濕度作為特徵，將所有數值標準化到0, 1之間，以利後續的分群
5. Clustering
   使用sklearn套件提供的KMEANS，進行分群作業
6. Plot
   使用Pylab進行，好處是可以做很多縮放、選取等工作，更適合觀察資料點

結果比較

論文釋出程式碼實驗結果	我們實作結果
	

可以從圖看出來，分群出來的結果幾乎與論文的一樣，為了驗證我們的結果完全相同，我們將全部的資料點及其被分配到的Cluster儲存起來，並且做比對：

```
1  paperResult = pd.read_csv('ClusteredData.csv')
2  ourResult = pd.read_csv('ClusteredData_modified.csv')
```

```
1  paperResult[['Name','Cluster']].equals(ourResult[['Name','Cluster']])
```

```
True
```

結果為完全一樣，完成還原論文結果的工作

實例應用

我們主要去收集台灣以及世界的氣象資料來做實驗，首先蒐集資料:

- 台灣:

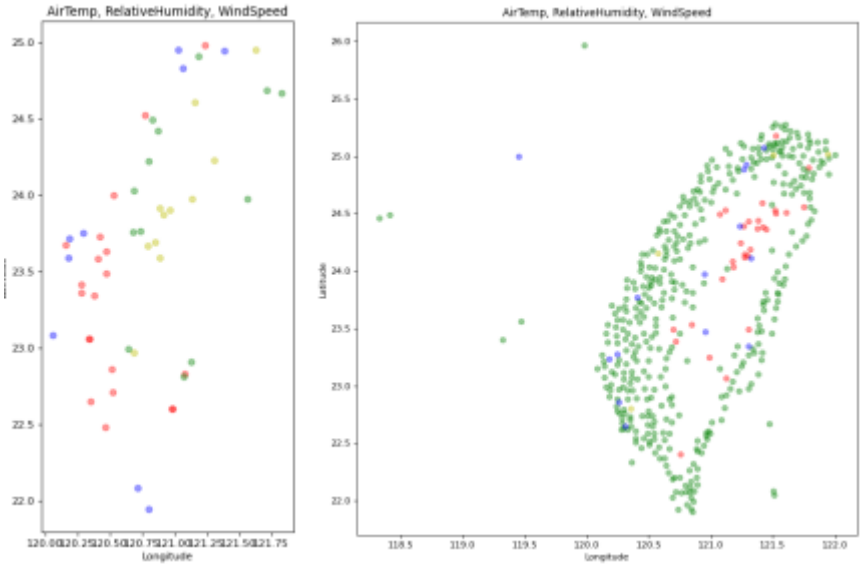
台灣的部分我們主要是從台灣中央氣象局的網站上面抓資料，該網站會有公開台灣各地檢測站的觀測紀錄，而我們主要抓兩種資料，分別是: 月資料、日資料

網址分別是:

https://opendata.cwb.gov.tw/dataset/observation/C-A0009-001

https://opendata.cwb.gov.tw/dataset/observation/O-A0001-001

而我們跑出來的結果如下圖



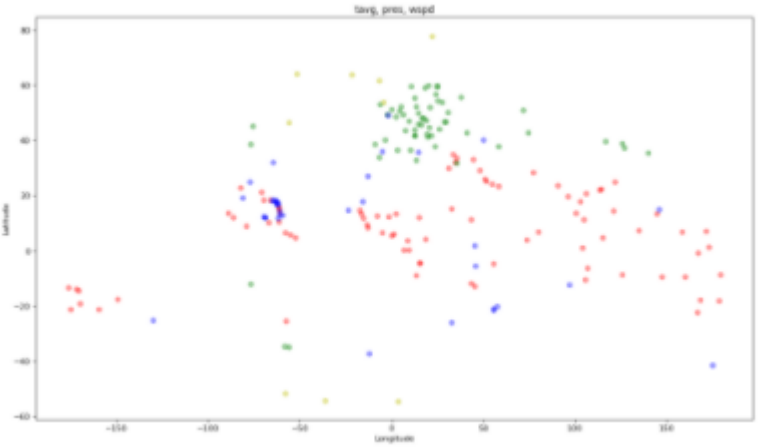
在圖中可以發現他基本上是可以看的出來有成功的將台灣的山區與沿海分群出來。左圖的月資料由於資料比較時間拉得比較長，因此比較能看出差異。至於右圖由於資料比較密集，因此我們認為可以拿來做異常檢測，例如 台南、高雄那一帶沿海，理論上應該要被分為同群，但是若出現像 藍點 這樣的不同群，這樣我們就可以猜測該檢測器可能有出現問題

- 世界:

  世界的資料我們找蠻多網站的，但後來發現 kaggle 有一個 dataset 最全面，因此最後決定用它來跑實驗，網址:

  https://www.kaggle.com/datasets/muthuj7/weather-dataset

  而我們做出來的結果如下圖:



雖然資料有點少，但還是可以看出大陸及沿海的差異，而且由於此實驗有以氣壓來做區分，因此其實可以發現像是非洲亞洲高海拔地區，會跟其他海拔較低的地方有明顯的差異。

結論

這篇論文主要的貢獻是在天氣分群的應用上驗證作者他們的IOT大數據分析框架是有用的，這個框架也可以應用在其他的場景上，例如:交通系統、金融應用、智能電網等，但對於這個天氣分析的應用上我們覺得這篇論文應該花多一點篇幅在異常檢測上，因為我們認為他花太多篇幅在敘述分群的結果上，但我們認為這篇最有用的是在異常檢測的應用上，因為天氣分群可能在地理方面已經被研究的很透徹了，所以可能這個天氣分群沒辦法對日常生活有很大的幫助。

對於論文中天氣偵測應用上的Feature work我們覺得他可以把sensor異常偵測自動化，因為我們認為論文中它們對於異常偵測的部分是用人工去看得。

參考資料

ONAL, Aras Can, et al. Weather data analysis and sensor fault detection using an extended IoT framework with semantics, big data, and machine learning. In: *2017 IEEE International Conference on Big Data (Big Data)*. IEEE, 2017. p. 2037-2046.