即時交通數據處理與流量預測模型

國立台灣師範大學資訊工程系

學生：40947012S 黃至瑜、40947064S 蔡少芸，指導教授：王科植

(二)摘要

交通數據視覺化主要解決日常生活需求，利用數據分析與演算法來瞭解塞車、道路新建、路線規劃等道路狀況，再由儀表板呈現給目標受眾操作。另外，在政府平台也有相關交通數據使用，例如：交通部數據匯流平台提供的交通大數據核心願景，包含提升交通運輸安全、重大交通路況疏導、公共運輸創新服務。

圖 1. 交通部數據匯流平台官網

資料數據主要可分為基本資訊、車速與車流量、其他資訊三種數據。基本資訊包含道路座標、名稱、分類等。其他資訊則是道路走向、更新時間等。時間分布從 2018 年 1 月至 2020 年 12 月，地點分布在台灣北部且以台北市為主(121.62~121.46°E, 25.17~24.9°N)。

(三)研究動機與研究問題

目標受眾包含交通部、公車業者、建設公司、旅行社四者。

1. 交通部

交通部可以預測車輛壅塞狀況。因為在連假可能會對部分高速公路路段進行高乘載管制，所以需要平面替代道路，如省道等。除此之外，交通部需要各種車的車速來進行道路的新建與改善，而道路規劃需要根據此路段車輛需求。

2. 公車業者

公車業者根據通勤人數與車流量多寡決定公車班次與規劃行車路徑路線。例如：羅斯福路車流量大，表示通勤的人多，所以公車班次密集。

3. 建設公司

建設公司新建案推出時，會先調查好旁邊道路噪音狀況，除了實地勘察之外，還有事前可利用車流量預估新住宅窗戶隔音。另外，靠馬路低樓層因為噪音較大，往往房價較低。

4. 旅行社

旅行社可以依據車流量選定旅行月份和預估車程往返的時間。若目的地路線的車流量較多，就可以預留更長的時間，也可以選擇流量較少的路線作為第一選擇，以便旅客規劃旅行路線及景點數目。

(四)文獻回顧與探討

首先，在繪製儀表板前，我們參考過各種套件的儀表板，由於我們所製作的為互動式儀表板以及資料數量相當可觀，因此想呈現的視覺化界面必須簡單、清楚。

Heavy.AI 上各式儀表板的版面設計提供我們諸多想法如何將數據實際應用在儀表板上，包含如何將地圖與其他圖表產生關聯，讓使用者依照個人需求取得特定資料。其中有 demo 使用熱力圖更提供我們可以將時間與日期的各項數據展示在儀表板，以及地圖旁邊的長條圖來表示地圖上數據大小的整理。

原先我們曾考慮過使用 D3.js graph gallery，但是最終選擇 Python Plotly 應用在 Dash 作為本次專題所使用的工具。Plotly 也提供不少圖表的選擇，而我們使用的方法都是參照官方文件的學習為主。遺憾的是 Dash 在官方文件描述並不多，在學習上遇到的問題較難排解，不過最終呈現的問題還是有找到因應的解決辦法。

(五)研究方法及步驟

本次專題中，我們選擇了使用 Python 來進行資料讀取、處理及實作互動式儀表板。

關於資料讀取及處理的部份，由於到手的資料份量頗多，無法直接用 Excel 打開，再來是 python 有提供很多視覺化的套件，所以我們先用 python 的 Matplotlib 套件，把資料以圖表的形式展現出來，觀察其特性及變化，然後是確定我們的目標使用者，並且繪製設計圖紙，考慮儀表板該有的功能，從資料中選定我們需要或有用的資料，之後把這些數據單獨提取出來，滙整成新的 CSV 文件，以免後續實作儀表板時讀取時間過長。

關於實作的部份，因為 Dash 的 Callback 可以實現圖表之間的互動，我們最終選定了使用 Dash 實作儀表板，由於這也是我們首次接觸 Dash 這個套件，所以花了不少時間去學習它的用法。我們在 source code 中建立了兩個函式，分別是 time_def 及 week_def，主要是處理熱力圖的資料時會用到，由於原資料中並沒有提供星期的數據，因此我們有定義 week_def 函式去做星期的計算，然後還有區分時段，我們有定義 time_def 把時間作分類。繪圖方面我們使用了 Python 中的 plotly.express 套件去繪製圖表，我們使用到的函式有 px.scatter_mapbox、px.bar、px.imshow、px.pie、及 px.line，其中 mapbox_style 設定為 open-street-map。然後是互動實作，地圖選取後的數據（SelectData）是由 Dash 生成，SelectData 的 Structure 可以參考圖片。全部的圖表在 SelectData 為 None 時，則以 CSV 的全部資料繪製圖表，否則按以下思路實作選取後的圖表：

1. 長條圖：

由於地圖中各個氣泡的大小（size）設定為流量數，而長條圖是繪製氣泡個數，所以這邊我們是以 SelectData 的 marker.size 作為長條圖的繪圖數據。利用迴圈比對每筆 SelectData 的 size 是否屬於某一個數值的範圍，是則 FLOW_COUNT 加 1，最後再重新繪製長條圖。

2. 其他圖：

我們要先找到地圖和其他圖表所使用的數據上有哪些相同的資料或有關聯的部份，由於地圖和其他圖相同的資料是路段名稱，因此我們選擇了以路段名稱（RoadName）作為資料篩選的條件，但 SelectData 的 Structure 僅提供了坐標點，所以在每一個 Callback 函式繪圖前，我們都先做了一個資料篩選的步驟，先是用 For 迴圈搜尋 SelectData 包括的路名（使用自定義函式 get_roadname），然後把回傳的字串加到 TargetRoadName（字串陣列）中，然後再進行

繪製圖表所需的資料處理，這邊的函式會跟 SelectData 為 None 時的 code 相同。

熱力圖是先做字串的切割，建立一個二維陣列 (7x3)，取得時間及日期後，把同一時段同一星期的大車流量，汽車流量及機車流量加總起來，儲存到陣列對應的位置中。

圓餅圖是建立一個大小為 3 的陣列，儲存大車、汽車及機車的資料，用 For 迴圈找到目標路名在原資料（df）對應的 index，然後把該 index 對應的各車輛流量存到陣列中。

折線圖也是先做字串的切割，用 For 迴圈找到目標路名在原資料（df）對應的 index，然後把該 index 對應的日期做切割，用原資料流量數乘原資料中的車輛速度計算各車輛總速度，如大車的流量乘大車的平均速度，這時會用把總車流量及總速度存到大小為日期個數的陣列中，總車流量存到 volume_big、volume_car 及 volume_mot 中，總速度存到 speed_big、speed_car 及 speed_mot 中。迴圈結束後，再把各個日期的總車流量除以總速度，數值會存回 speed_big、speed_car 及 speed_mot 中。

```
{
    "points": [
        {
            "curveNumber": 0,
            "pointNumber": 106,
            "pointIndex": 106,
            "lon": 121.53488,
            "lat": 25.06633,
            "hovertext": "\u5efa\u570b\u5317\u8def\u4e09\u6bb5",
            "marker.size": 4992
        },
        {
            "curveNumber": 1,
            "pointNumber": 0,
            "pointIndex": 0,
            "lon": 121.53322,
            "lat": 25.06844,
            "hovertext": "\u5efa\u570b\u5357\u5317\u5feb\u901f\u9053\u8def",
            "marker.size": 231936
        }
    ],
    "lassoPoints": {
        "mapbox": [
            [121.5379829885, 25.07096696289537],
            [121.53966045444412, 25.07096696289537],
            [121.542438757416, 25.071726671139118],
            [121.54443074822552, 25.072201486397645],
            [121.54448316903597, 25.072343930616753],
            [121.542438757416, 25.07148926281883],
            [121.54228149498323, 25.071346817606283],
            [121.54233391579368, 25.070539624937666],
            [121.5432250695768, 25.06982739169456],
            [121.54427348579281, 25.069589979693347],
            [121.54285812390236, 25.067263317726912],
            [121.5416524452537, 25.066123712067906],
            [121.54002740011998, 25.065553905263485],
            [121.53861203822964, 25.065363969072806],
            [121.5355192103936, 25.065553905263485],
            [121.53289816985449, 25.066123712067906],
            [121.53179733282815, 25.068450395680358],
            [121.53122070391044, 25.069115154309756],
            [121.53158764958499, 25.06973242694916]
        ]
    }
}
```


圖 2. sample.txt

(六)實驗結果

我們的交通資料會以互動式儀錶板來呈現數據的特點及變化，互動的方式依照圖表的可操作性而有所不同。在排版方面有分上下兩排，第一排從左到右依序為的圖表為地圖、長條圖及熱力圖；下排從左到右依序為折線圖和圓餅圖。

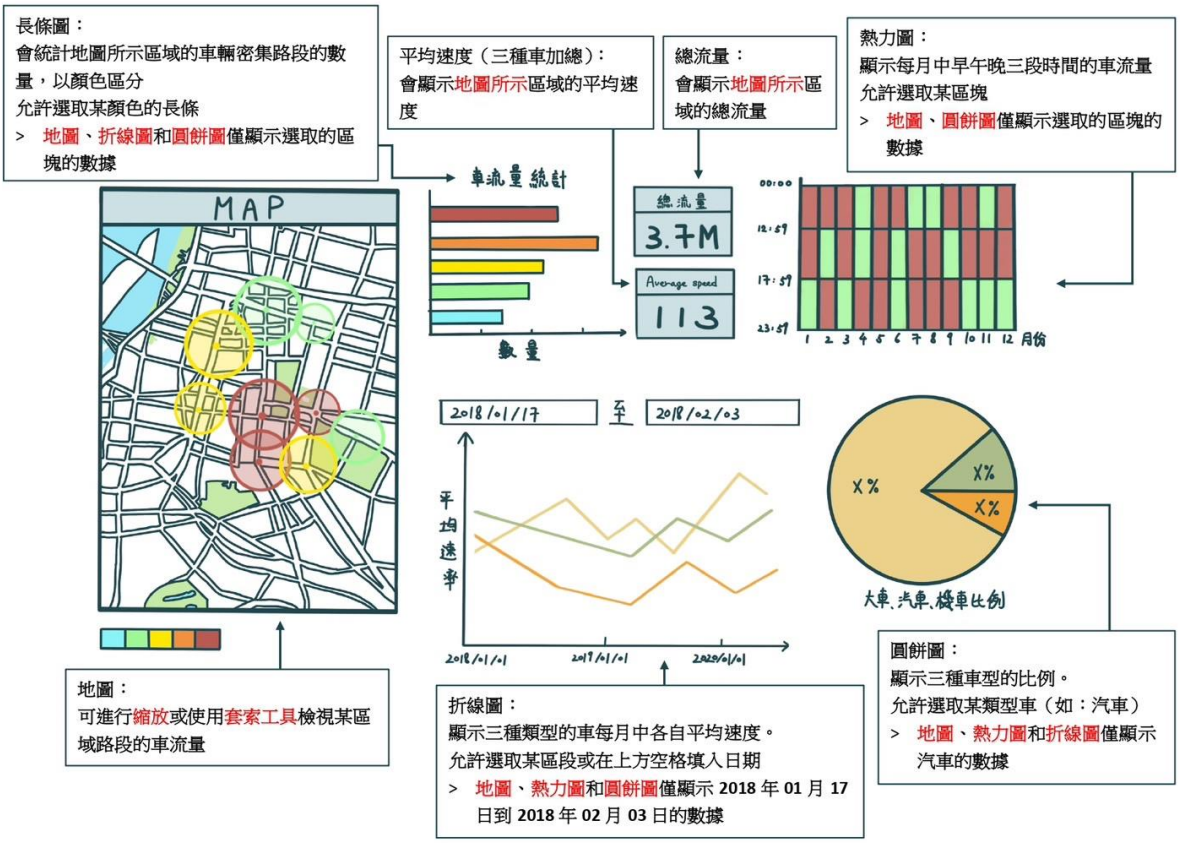

圖 3. 手繪原始 Dashboard

地圖所顯示的為不同路段中大車、汽車和機車三種車輛的總車流量，讓使用者可以看到車流量的分佈。預設狀態時會顯示所有路段的車流量，以不同顏色標示流量的大小—4 萬以上以紅色標示，3 萬至 4 萬以橙色表示、2 萬至 3 萬以黃色表示、1 萬至 2 萬以綠色表示及 1 萬以下則以藍色表示，可進行的操作為縮放地圖、選取分類標籤或使用方圖形或套索工具選取地圖上的氣泡。根據選取範圍，其他圖表會顯示對應數據(dashboard picture 1,2,3)，這部份會在介紹不同圖表時提及。

長條圖所顯示的數據是統計地圖上的氣泡個數，以便使用者更直觀的看到流量的程度。預設狀態時會顯示所有顏色的氣泡個數，標籤顏色和地圖相同，可進行的操作為選取分類標籤及選取放大某區段的數據顯示。已實作的互動為根據地圖上選取的範圍，長條圖會顯示該範圍中氣泡的個數統計。

熱力圖所顯示的數據是一週中在早上、中午及晚上三個時段的車輛密集程度，讓使用者可以觀察時間對流量變化的影響，若要進行路段分流，有哪些時段比較急需做分流。時段的劃分為早上（00：00：00 至 11：59：59）、中午（12：00：00 至 17：59：59）及晚上（18：00：00 至 23：59：59）。數據在熱力圖上所示之顏色為連續顏色，數人值從少到多依序為藍色、綠色、黃色、橙色及紅色。預設資料範圍為讀入資料的所有路段，可進行的操作為選取分類標籤及選取放大某區段的數據顯示。已實作的互動為根據地圖上選取的範圍，熱力圖會計算該範圍中所包含的路段的總車流量大小的密集程度並顯示結果。

折線圖所顯示的數據是資料中各日期中大車、汽車及機車的平均速度，讓使用者可以觀察三種車型的車輛每日的平均速度，結合其他的視覺化的圖表，分析道路擠塞的情況。大車平均速度以綠色線段表示，汽車平均速度以黃色表示，機車平均速度則以橙色表示。預設資料範圍為各車輛中在全部資料中所有日期的平均速度，可進行的操作為選取分類標籤或選取放大某區段的數據顯示。已實作的互動為根據地圖上選取的範圍，折線圖會計算選取資料中所包含的路段中各車輛每一日的平均速度並顯示結果。

圓餅圖所顯示的數據是大車、汽車和機車三種類型的車在流量中所佔的比例，讓使用者可以依照車輛的佔比去做道路的設計或擴建。預設狀態下的數據為各類型的車輛在總流量中各自的佔比，標籤顏色與折線圖相同，大車以綠色表示，汽車以黃色表示，機車則以橙色表示。可進行的操作為選取分類標籤及選取放大某區段的數據顯示。已實作的互動為根據地圖上選取的範圍，圓餅圖會計算該範圍中所包含的路段中各車輛佔比並顯示對應的結果。

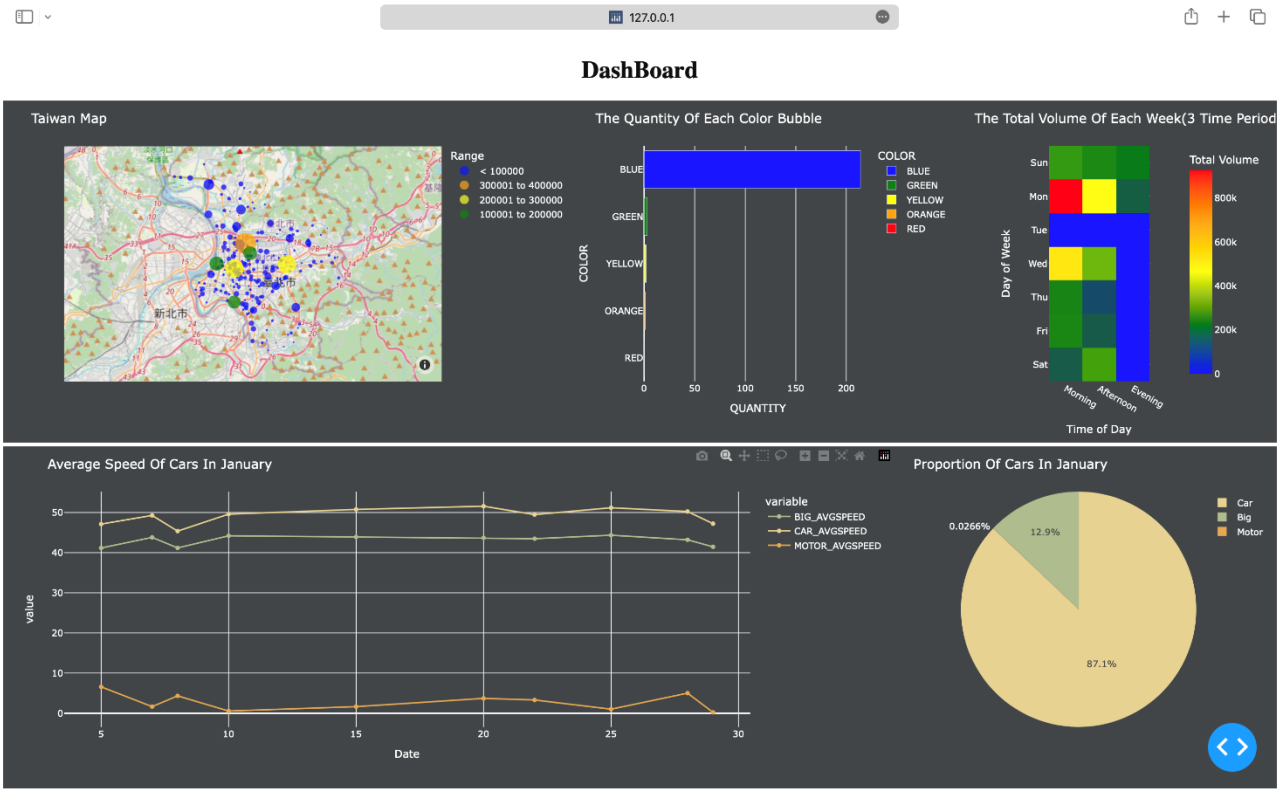

圖 4. 實作預設 Dashboard

圖 5. 實作方塊選取 Dashboard

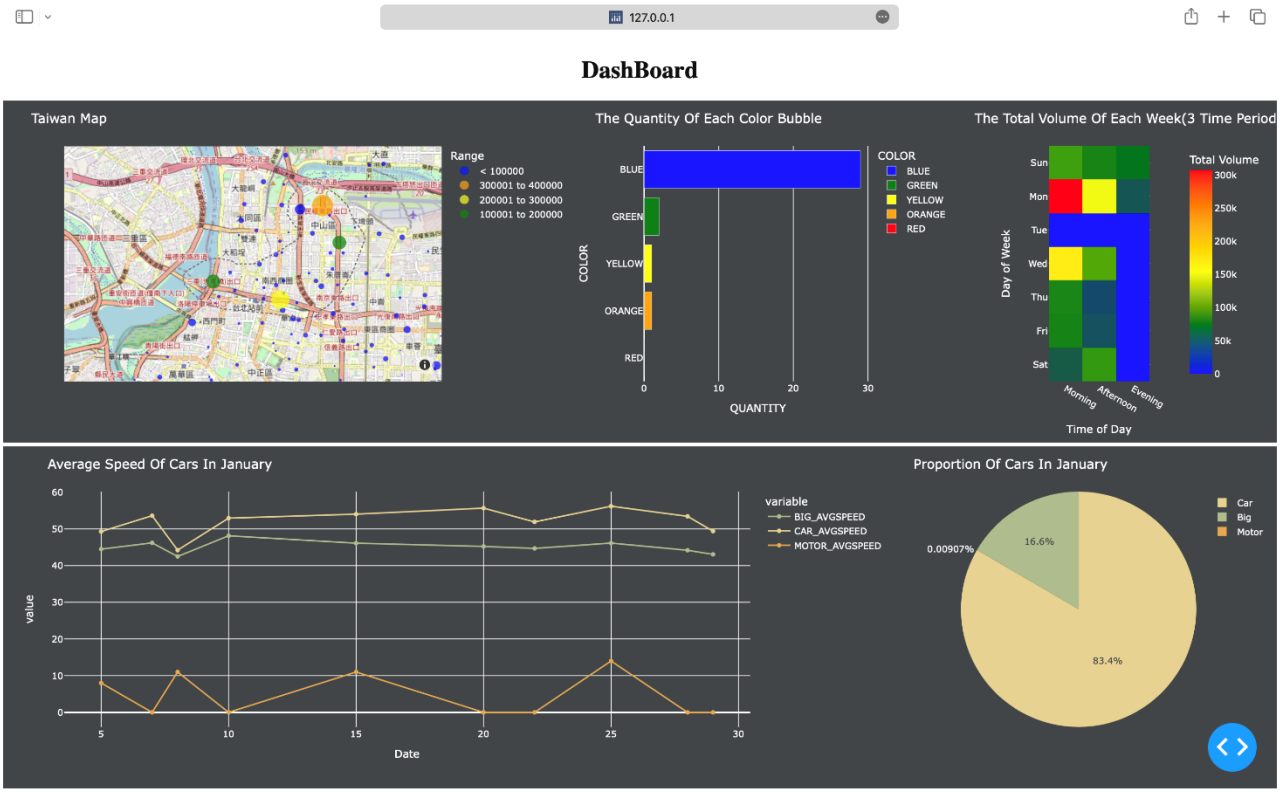

圖 6. 實作任意選取 Dashboard

(七)分析與討論

開始接觸資料時，因為資料量很龐大，所以我們組內討論把一些不用的變數給去年掉，這樣也可以縮減資料載入的時間，最後決定留下只坐標、道路 ID、路名、三種車的流量及速度、平均速度、速限、資料更新日期及時間的數據。目標使用者方面，我們原本也考慮了不少，比如公車業者、交通部公路總局、建設公司、建商及旅行社或自由行的旅客，以及他們可以使用這個儀表板的情況，目前的我們的設計主要以交通部需求為主，我們組在和老師的指引下，最終決定了儀表板的功能為顯示流量分佈的地圖、統計分佈數量的長條圖、流量密集程度的熱力圖、統計車輛比例的圓餅圖及三種車輛每日的平均速度。討論的部份還有實作的工具，老師推薦了 D3.js 和 Dash 給我們，由於我們前面處理數據時是使用 Python 的，而且學習時間有限，所以決定用較為簡易的 Dash 來實作我們的儀表板。

然後是現在的情況，雖然儀表板的基本框架做好了，可是實作互動後程式明顯會卡頓，這部份還需要再進行優化，還有就是下學期的方向，在專題開始前，我們也有跟老師討論相關的方向，這部份會在下一點補充。

(八)結論及未來研究方向

目前的互動部分實作了以地圖為主操作，其他圖表會依照地圖的選取資料而更改其內容，其他，如點選長條圖中紅色的部份時，地圖僅顯示紅色區域的資料等，則會在期末報告後進行再處理，這是因為僅目前的互動明顯有載入時間過長的問題，這個問題會在下學期時一併處理，目前的想法是把算法優化，如使用其他排序演算法讓搜尋目標路段名稱的時間可以縮減。

至於下學期的研究方法，我們現在考慮的地方有建立數據庫，由於我們現在拿到的資料筆數很多，而且資料都被拆分成多個檔案，我們在開首接觸資料時也發現了每個 CSV 檔案的數據排序依據也不一樣，像是 2018 年的會以日期作資料排序，而 2019 的資料則是以路段名稱作排序的依據，為了方便數據讀入的問題，我們考慮未來會建立資料庫，把我們現在拿到手的資料可以整合在一起並處理。其次就是整理出進階模型，用一些演算法協助使用者做數據的分析，而不是單純一些視覺化圖表，比如找出不合理數據的原因或者路段流量過多的可能因素。

之後還有考慮做成動態的數據視覺化，現在我們的資料都是老師提供給我們的，然後我們再做處理，但是這些數據都是過往的資料，跟當下的車輛數據也會有差距，因此我們希望可以做得到較為即時的數據顯示，即可以從後端載入資料，做到一個即時更新數據的儀表板。以上為我們組未來的研究方向。

(九)參考文獻

1. Interactive Visual Analytics Demos： https://www.heavy.ai/demos

2. Plotly Open Source Graphing Library for Python： https://plotly.com/python/

3. Dash Python User Guide： https://dash.plotly.com

4. D3.js graph gallery： https://d3-graph-gallery.com