淡江大學運輸管理學系運輸科學碩士班

碩士論文

指導教授：陶治中 博士

從社群媒體挖掘以感測日常交通滿意度之研究

Sensing Daily Travel Satisfaction Of Commuters by Mining Social Media

研究生：陳 翰 撰

中華民國 107 年 06 月

誌謝

時光匆匆新生訓練彷彿才在昨天

一轉眼在學校流連一留就是六年

六年時間對我未來造成劇烈改變

歷經朋友決裂、頭頂發綠、官司纏身、

戶頭歸零、文人相輕、二一聽牌等等

事情發生了，解決問題才是唯一的方法

我慶幸在蘭陽校區認識一群難兄難弟

我知足轉回淡水本部結識另一群兄弟

我珍惜北投地緣關係所剩無幾的朋友

我惜緣陶老師願意收我為陶家班門徒

我惜福出生在衣食無缺、無憂無慮的家庭

我感謝所有朋友在背後給予我精神上的支持

感謝所有在我人生不同階段幫助我的人

感謝老天爺在不同時刻安排不同貴人

相遇是緣起，相識是續緣，相知是緣定

就讓我們隨緣惜緣吧!

論文名稱： 從社群媒體挖掘以感測日常交通滿意度之研究　　頁數：93

校系(所)組別：淡江大學運輸管理學系 運輸科學研究所

畢業時間及提要別：106 學年度第 2 學期 碩士學位論文提要

研究生：陳　翰　　指導教授：陶治中　博士

論文提要內容：

本研究係針對日常交通下社群媒體進行文本挖掘，首先透過建置社群媒體文本資料庫，並將日常交通之社群媒體文本進行情感分析，以瞭解民眾對於日常交通下各公共運具評論之情緒，接著使用深度學習之 CNN 演算法建構多元情緒決策情感分析模型，再經由實證分析，運用文本挖掘及深度學習分類技術，分析民眾日常通勤通學搭乘公共運具之滿意度，並建構滿意度五個評級尺度，找出民眾搭乘不同公共運具之整體情緒，最後使用 K-means 集群演算法探討社群媒體輿情內部特徵，此為本研究建立之社群媒體挖掘與情感分析通用程序。

本研究之模式準確度在五個尺度下達 79%，並可找出民眾搭乘不同公共運具所關注的變數。研究結果顯示：不論通勤或商務旅次皆關注是否能準時到達目的地，可見民眾最為重視搭乘公共運具之效率；而搭乘火車類、公車類、客運類之通勤族皆關注駕駛員及排班，可見民眾重視資訊的掌握與服務及乘車的安全性。根據自社群媒體挖掘而產生的公共運輸滿意度結果，本研究經由相關文獻評析而研擬相對應的改善策略。若特定時間的班次常有誤點情況時，業者應適時發佈到站時間資訊，則可減少民眾的負面情感。而駕駛員的素質、服務態度、應對能力等，對於民眾的影響亦相當直接，因此業者應對駕駛員進行系統性的在職訓練，亦有望提升民眾的正面情感；當前所有通勤族群對於費率的關注程度皆不明顯，顯示目前我國公共運具在費率制定尚受民眾肯定。本研究之建立之社群媒體挖掘與情感分析通用程序具有擴展性，此可供決策者藉由模型的修正與改良而能快速掌握網路輿情正、負面情感趨勢之即時資訊。

關鍵字： 社群媒體挖掘、情感分析、日常交通滿意度、卷積神經網路演算法

Title of Thesis： *Total pages: 93*

Sensing Daily Travel Satisfaction Of Commuters by Mining Social Media

Key word: Social media mining, Sentiment analysis, Daily travel satisfaction, Convolutional Neural Networks algorithm

Name of Institute:
Graduate Institute of Transportation Science, Tamkang University

Graduate date: June 2018 *Degree conferred: Master Degree*

Name of student: Han Chen 陳 翰 *Advisor: Dr. Chi-Chung Tao* 陶 治 中 博士

Abstract:

This study aims at proposing a generalized process of social media mining and sentiment analysis to sense commuters' daily travel satisfaction. Firstly, available social media websites are chosen to perform text mining and filtered text database related to daily travel topics by using crawler systems. Secondly, a sentiment analysis is conducted to propose a multiple emotion recognition model which can be used to sense commuters' emotions about public transportation vehicles including high speed rail, commuter rail, mass rapid transit, urban bus, intercity bus, specific bus and taxi by using Convolutional Neural Networks (CNN) algorithm from deep learning. Thirdly, an empirical study is performed to validate commuters' daily travel satisfaction towards different public transportation vehicles with a five-grade-scale emotion recognition survey. Finally, influence factors depicting interrelationships among critical topics concerned public transportation services in social media mining are clustered with K-means algorithm and corresponding strategies to improve negative emotions against certain public transportation services are also provided. Empirical results show that the precision percentage of proposed model to verify critical variables of commuter's public transportation satisfaction approximates 79% under five-grade scales. Either commuter or commercial trip purpose arrivals at destinations on time are much concerned by public transportation commuters. Driver behavior and timetable are also valued by public transportation commuters. It is recommended that operators should pay more attention to adjusting timetable and conducting systematical driver training programs for better emotions. In addition, fare topic is not significant for public transportation commuters that means current fare structures of public transportation are relatively acceptable by commuters. The proposed generalized process of social media mining and sentiment analysis in this study can be expanded with adequate modifications or improvements to grasp real-time information about net citizens' emotion trends for decision makers.

目錄

第一章　緒論.....1

1.1　研究背景與動機.....1

1.2　研究目的.....4

1.3　研究範圍與流程.....5

第二章　文獻回顧.....8

2.1　社群媒體文本挖掘.....8

2.1.1　社群媒體.....8

2.1.2　文本挖掘.....10

2.2　情感識別.....11

2.2.1　輿情定義.....11

2.2.2　情感分析.....12

2.2.3　情感識別.....13

2.3　日常交通滿意度.....14

2.4　深度學習.....17

2.4.1　人工神經網路.....18

2.4.2　深度學習.....19

2.5　小結.....21

第三章　研究方法.....22

3.1 文本斷詞系統.....23
3.2 詞頻演算法.....24
3.3 卷積神經網路.....25
3.4 K-means 集群演算法.....32
第四章 實證分析.....33
4.1 實證分析流程架構.....33
4.2 爬蟲流程.....34
4.2.1 爬蟲流程說明.....34
4.2.2 資料內容說明.....39
4.3 結巴(jieba)斷詞.....40
4.4 情感值計算.....41
4.5 卷積神經網路.....48
4.5.1 輸入層.....49
4.5.2 卷積層.....50
4.5.3 激活函數.....51
4.5.4 池化層.....52
4.5.5 全連接層.....53
4.5.6 丟棄層.....54
4.5.7 模式建構.....55

4.5.8 模式驗證.....64

4.6 詞頻演算法.....68

4.7 K-means 集群結果.....69

4.8 管理意涵.....74

第五章 結論.....75

5.1 結論.....75

5.2 建議.....78

參考文獻.....79

附錄 A 模式驗證結果.....84

圖目錄

圖 1.1 國人擁有社群帳號比例.....2

圖 1.2 研究流程圖.....7

圖 2.1 民眾外出旅次目的占比.....15

圖 2.2 民眾對於搭乘公共運具服務的滿意度.....16

圖 2.3 文字辨識函數表示方式 (李弘毅).....17

圖 2.4 圖片識函數表示方式 (iker).....18

圖 2.5 神經元結構.....19

圖 2.6 人工神經架構.....19

圖 3.1 情感分析模式建構流程圖.....22

圖 3.2 結巴斷詞成果.....23

圖 3.3 CNN 基本架構.....26

圖 3.4 常用激活函數及其函數圖像.....29

圖 3.5 傳遞神經網路架構.....30

圖 4.1 實證分析流程圖.....34

圖 4.2 爬蟲程式(第一層關鍵詞).....35

圖 4.3 爬蟲程式(第二層關鍵詞).....36

圖 4.4 爬蟲程式(輸入情緒詞).....37

圖 4.5 爬蟲程式(結果).....38

圖 4.6 安裝結巴套件.....40

圖 4.7 結巴斷詞執行結果.....41

圖 4.8 訓練矩陣結果.....41

圖 4.9 情感區間尺度.....44
圖 4.10 情感值區間長條圖.....44
圖 4.11 捷運類情感趨勢圖.....45
圖 4.12 火車類情感趨勢圖.....46
圖 4.13 公車客運類情感趨勢圖.....47
圖 4.14 其他類情感值趨勢.....48
圖 4.15 Tensorflow 運算流程.....49
圖 4.16 卷積神經網路架構.....50
圖 4.17 卷積運算示意圖.....51
圖 4.18 線性整流函數.....52
圖 4.19 Max pooling 操作.....53
圖 4.20 全連接層示意圖.....54
圖 4.21 Dropout 機制示意圖.....55
圖 4.22 CNN 參數設計.....56
圖 4.23 CNN 執行過程(jieba 斷詞).....56
圖 4.24 CNN 執行過程(訓練 1).....57
圖 4.25 CNN 執行過程(訓練 2).....58
圖 4.26 CNN 執行過程(訓練 3).....59
圖 4.27 CNN 執行過程(訓練 4).....60
圖 4.28 CNN 執行過程(訓練 5).....61
圖 4.29 CNN 執行過程(評估 1).....62
圖 4.30 CNN 執行過程(評估 2).....62

圖 4.31 CNN 執行過程(評估 3) 62
圖 4.32 CNN 模式訓練測試震盪圖 63
圖 4.33 模式驗證結果(非常滿意).....64
圖 4.34 模式驗證結果(滿意).....65
圖 4.35 模式驗證結果(普通).....65
圖 4.36 模式驗證結果(不滿意).....66
圖 4.37 模式驗證結果(非常不滿意).....66
圖 4.38 擴增特徵詞.....69

表目錄

表 2.1 外出旅次目的之主運具類別占比.....16

表 2.2 民眾搭乘各項公共運具之滿意度.....17

表 4.1 資料型態.....39

表 4.2 部分情緒詞彙.....42

表 4.3 日常通勤情感值區間門檻.....43

表 4.4 設計情感值區間相關參數.....43

表 4.5 情感區間計算.....44

表 4.6 捷運類情感值.....45

表 4.7 火車類情感值.....46

表 4.8 公車客運類情感值.....47

表 4.9 其他類情感值.....48

表 4.10 新資料預測分類準確度.....67

表 4.11 混淆矩陣.....67

表 4.12 部分篩選完特徵詞.....68

表 4.13 合併後部分資料.....70

表 4.14 華德法凝聚過程表.....71

表 4.15 分群比較表(*表示平均值是最大或最小的一群).....72

表 4.16 K-means 分群特徵分布(粗體表示該變數最大或最小值) 73

表 4.17 各運具滿意度之改善方案.....74

第一章　緒論

1.1 研究背景與動機

我國交通部每年皆會進行「民眾日常使用運具狀況調查」，其中各項指標包括運具市占率、使用族群、旅次目的、主要運具、公共運輸服務滿意度以及未使用公共運輸的原因等，此係期能了解民眾對於交通服務之預期與實際感知差距，進而針對公共運輸服務進行調整與改善，以達到提升公共運輸使用率之目的。

整體交通施政滿意度之提昇，應是考量滿足民眾交通需求，使民眾感受交通服務之滿意度最大化為目標，然而當前對於交通服務「滿意度」之調查，僅就使用者對於該項服務提供的服務屬性(如旅行時間、旅行費用、是否舒適…等)是否感到滿意，並未對於服務是否能讓民眾滿足需求探討。此外，欲達到提升公共運輸使用率之目的，應瞭解使用者會評估該選擇對於自身的「效用」(utility)。因此僅調查使用者對於服務屬性之「滿意度」，而未考量使用者之實際體驗，將難以了解使用者對於交通服務之行為意向。

近年來隨著社群媒體(Social Media)的蓬勃發展，民眾經常使用 Facebook、LINE、Instagram、PTT、痞客邦(PIXNET)等社群媒體網路來表達自己的觀點，發表對日常生活熱門議題的評論。社群媒體儼然成為民眾發表意見、抒發情緒、彰顯自身觀點與立場的重要平台。又因大數據分析的流行，衍生出社群媒體挖掘技術，其體現在智慧商業分析、政府決策輔助、民調與市調等諸多應用上，以市場分析與顧客管理為例，經由民眾發布的意見內容、點擊的網頁與停留時間長短、針對產品所發表的評論等大數據挖掘，進而分析民眾觀點與情感傾向，可更好理解民眾需求，以修正或改善產品與服務，進而推估與預測市場趨勢。由此可見，社群媒體用戶評論的分析，對於無論產業界、學術界、政府相關單位皆具重要之意涵，然而至目前為止，雖前述已有諸多分析應用服務，惟交通服務相關社群媒體挖掘之研究卻仍少見於國內文獻。若能針對社群媒體用戶對於交通服務之意見

評論進行挖掘，並建立蒐集、建模、分析與應用回饋之流程化作業，從中尋找利於管理階層掌握問題之關鍵所在，則可有效擬定相關政策，提升服務水準。

根據資策會創新應用服務研究所 FIND 團隊(2017)調查結果顯示，國人平均擁有 4 個社群媒體帳戶，其中 Facebook(90.9%)與 LINE(87.1%)分別穩坐第一、二名的寶座，其他包括 YouTube(60.4%)、PTT(37.8%)、Instagram(32.7%)、微信、Twitter、Dcard 等，如圖 1.1 所示，可見國人對社群媒體依賴程度之高；其中，近年最能表現出社群媒體應用成效之案例，即為 2014 年台北市市長選舉，候選人柯文哲善用網路輿情，從 Facebook、PTT、網路新聞等各大社群媒體中，挖掘出年輕人有興趣之選舉議題，成功打贏選戰，而更早之前的太陽花學運亦同。過去社會輿情主要依靠報章雜誌、無線廣播、電視，但其傳遞速度慢、訊息感染力低及受地理範圍限制，使傳統社會輿情內容不易擴散傳播，然而網路輿情不論在傳遞速度及感染力皆非常迅速，其隱含的民意力量不容輕忽，倘若能有效蒐集、分析及管理輿情內容，則將有助於決策者了解問題所在，並可對其提出改善方案或建議。

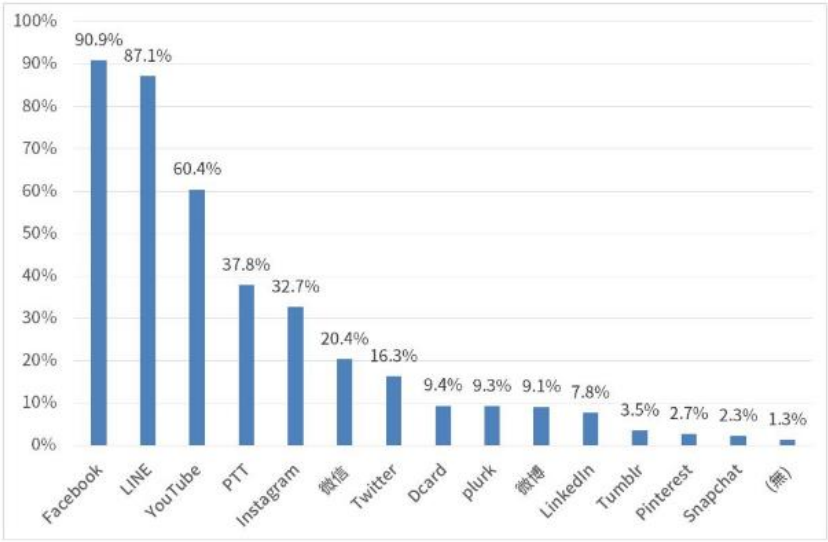

圖 1.1 國人擁有社群帳號比例

當前社群媒體輿情情感分析之主流技術為文本挖掘(Text Mining)技術，主要

是將文本依照文本的正負面情感進行分類，較常見的作法是先將文章進行斷詞挑選出情感特徵，再以機器學習的方式進行訓練分析。在過去情感分析研究中，大多致力於使用不同種特徵選取方式、不同機器學習方式或交叉搭配出不同組合結果，旨在透過系統化的分析流程，減少人力的介入，同時提高分析的精確性，尤其在針對領域性的輿情評論文本分析上，系統化的分析結果可最大化的減少人力的介入，使透過系統完成輿情評論分析之目的得以實現。

情感分析亦被稱為觀點挖掘、觀點分析與主觀、客觀分析等，情感分析目的係從文本中挖掘民眾所表達之意見與情感傾向，可提供策略訂定、決策輔助與營運管理等之參考。常見之情感分析方法以二元決策分類方法為主，係將情感分為正面與負面，相關的研究中亦將情感強度再分為高度、中度、低度之正面與負面，使分類結果具有量表性，方便體現重點之輿情內容；然而因二元決策分析在中立或低度之正、負面情感表現上容易模糊其界限，故後續研究有提出以三元決策分類，將較弱之正面或負面情感分類至中立情感，以增加分類之細緻度。

1956 年即有人工智慧(Artificial Intelligence, AI)定義，但隨著科技的進步，電腦硬體儲存及軟體效能大幅度提升，加上網際網路提供大量數據儲存並計算於雲端平台，使 AI 在近十年有爆炸式的發展及應用。根據多位學者(餘進程等, 2013, 尹邵龍等, 2015, Yanjie. D 等, 2016)研究結果得知，應用深度學習演算法比傳統淺層學習，不論在預測、分類皆有較好的準確度。

綜合上述，本研究沿用情感分析技術，將情感從廣義的二元正、負面分類做更細緻的分類，並參考三決策分類概念，針對細粒度更高之情感辨識進行準確分類。再者，本研究將採用深度學習演算法，改善傳統淺層學習方類準確度之不足，因此本研究將建立一個社群媒體文本挖掘與情感分析模式之流程，透過社群媒體之網路文本蒐集，運用文本挖掘及深度學習分類技術，探索民眾日常通勤通學使用公共運輸之滿意度，並進行民眾搭乘不同公共運具之整體情緒分析，以供公共運輸營運者了解問題所在，進而 對其提出改善、檢討策略。

1.2 研究目的

基於上述研究背景與動機，本研究之目的如下:

一、利用日常交通滿意度計畫案所建置之爬蟲系統，蒐集社群媒體文本，並建立日常交通滿意度之社群媒體文本資料庫。

本研究透過 2017 年 1 月至 10 月日常交通滿意度之社群媒體文本，建立日常交通滿意度之社群媒體文本資料庫，評論文本將納入文本相關資訊以提供後續情感分析及情感趨勢研究使用，相關資訊包含評論文本內容、精簡內容、抓取來源、輿情評論發生時間等。

二、利用文本挖掘及深度學習技術，分析民眾關注之相關話題與情感識別，透過實證分類結果，建構日常交通滿意度之輿情情感識別模型。

本研究結合結巴(jieba)斷詞套件、TensorFlow 軟體庫、Excel 軟體及計劃案爬蟲系統，建立日常交通滿意度之輿情情感識別模型，透過斷詞系統對社群媒體文本進行斷詞，利用 Python 程式及 Excel 軟體完成特徵詞庫建置，再利用卷積神經網路(CNN)進行情感識別模型建立，檢視其分類結果了解民眾關注議題及情感趨勢。

三、透過建置模型探討與分析網路輿情事件與現實事件之關係，並提出可用於日常交通滿意度之社群媒體輿情情感識別深度學習核心模型，以便繼續追蹤關注議題。

本研究建立日常交通滿意度之社群媒體輿情情感識別模型，可結合時間趨勢分析，直觀旅客於通勤通學所關注之議題，並將其分析結果提供相關主管單位，作為追蹤旅客通勤通學情緒變化之參考依據。

1.3 研究範圍與流程

本研究流程分別為確認研究動機與目的、現況分析與文獻回顧、確立研究架構與方法、實證分析及提出結論建議。各階段步驟說明如下：

一、 確認研究目的與範圍

本研究首先針對日常通勤學之社群媒體文本進行蒐集與分析，蒐集 2017 年 1 月至 2017 年 10 月之社群媒體文本，藉以瞭解民眾對於日常搭乘公用運具之通勤通學看法與使用者意見情形，以社群媒體文本資料為基礎進行分析，並提出情感識別分析模型。根據文獻回顧結果，研究範圍將運具限定在「計程車」、「交通車」、「捷運」、「高鐵」、「公路客運」、「市區公車」、「國道客運」、「臺鐵」；將旅次目的限定為「業務外出」、「商務」、「通勤」及「通學」之相關文本。本研究並未考慮各社群媒體之傾向性，僅認為各社群媒體皆多少可反映出部份民眾之真實意見。

二、 現況分析與文獻回顧

針對研究課題進行現況分析及相關文獻之蒐集與整理，釐清問題所具備之基本特性，藉此奠定本研究的概念架構基礎。文獻回顧內容包含四大部分：(1)社群媒體挖掘；(2)日常交通滿意度；(3)情感識別；(4)深度學習；最後則彙整出本研究文獻小結與評析。

三、 研究方法

應用深度學習(Deep Learning)演算法之卷積神經網路(Convolutional Neural Networks, CNN)進行多元情感識別門檻分類。根據現況分析與文獻回顧界定研究問題，建立本研究之實證分析架構圖，並參考相關學者提出之原理、經驗法則及研究結果。

四、 實證分析

本研究係針對日常通勤通學之社群媒體文本進行文獻佐證，並透過文本資料蒐集與處理，建立已完成社群媒體挖掘之文本資料庫，再以卷積神經網路建立模型探討日常交通之社群媒體輿情情感傾向與使用者情感識別。

五、 結論與建議

依據實證分析結果，彙整本研究之結論與後續建議。本研究之研究流程如圖 1.2 所示：

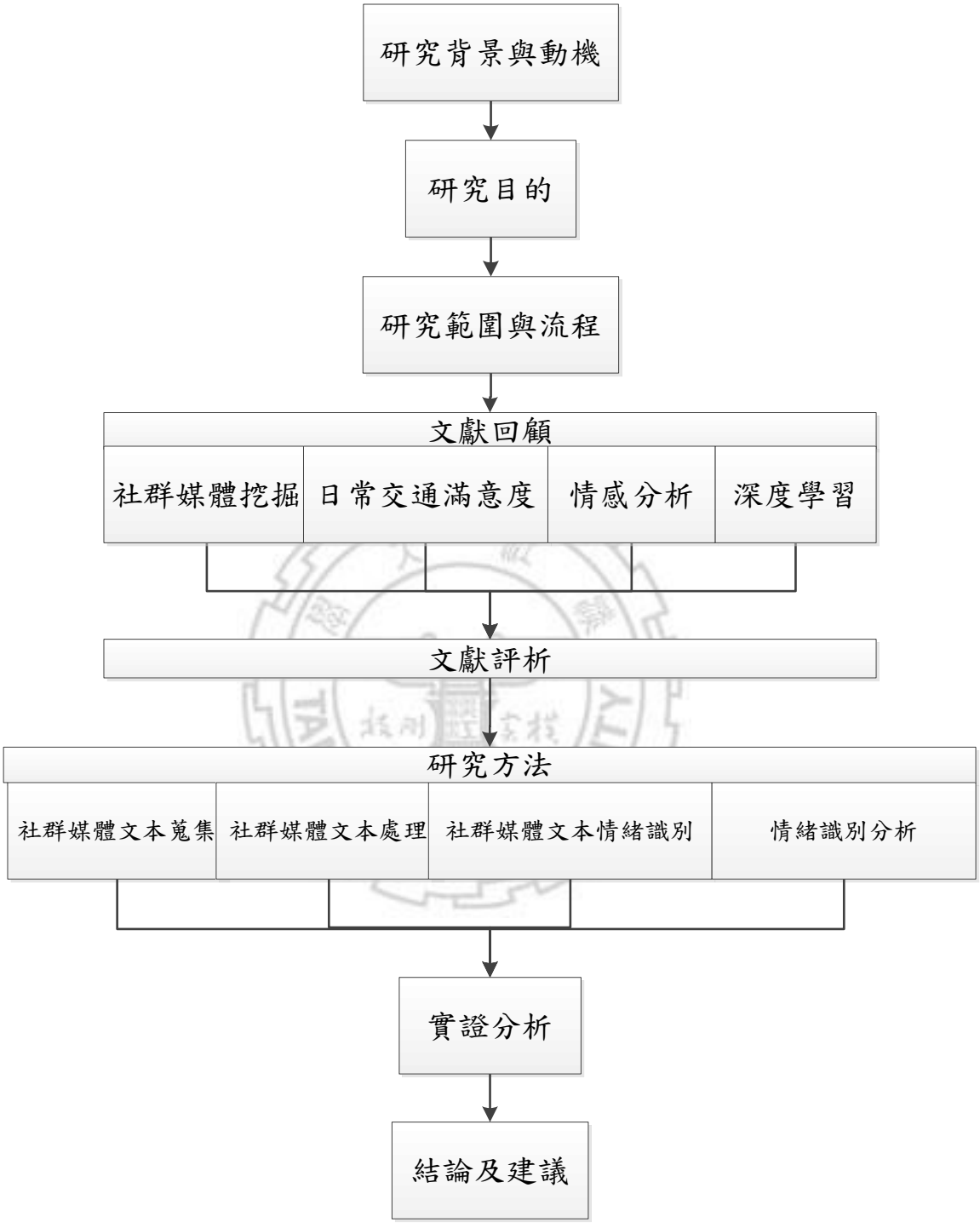
研究背景與動機
研究目的
研究範圍與流程
文獻回顧
社群媒體挖掘
日常交通滿意度
情感分析
深度學習
文獻評析
研究方法
社群媒體文本蒐集
社群媒體文本處理
社群媒體文本情緒識別
情緒識別分析
實證分析
結論及建議

圖 1.2 研究流程圖

第二章　文獻回顧

科技發展日新月異，互動性高且內容豐富的媒體網站相繼出現，使得社群媒體與情感識別(Emotional Recognition)之間的關係受到關注。社群媒體支援人類互動的需求，將傳統一對多(one to many)的傳播媒體，轉換成多對多(many to many)的社群媒體對話，把民眾從內容的消費者，轉變成內容產出者，因此社群媒體又稱為使用者內容產出(User-generated Content，UGC) (浩騰媒體，2009)。Kim and Geidner（2008）則認為線上的社群網路可被定義為以網路為基礎的社群，可讓個人展現自己、與其他社群網路連接並與他人維持聯繫 （Ellison et al., 2007）。

情感識別是隨著社群媒體發展而逐漸產生的現象，民眾提升對網路使用需求的習慣性，進而使報章雜誌、新聞媒體等訊息發布傳遞者，將過去民眾仰賴之紙本、無線電波等傳播媒介，移轉至網路上發佈，成為包含文字論述、表情符號及影音綜合多媒體訊息。因網際網路的發達及其快速、便捷的特性，民眾可藉由網路獲取即時資訊，並於社群媒體上表達各式各樣意見，其影響程度日漸重要。

藉由社群媒體文本經情感識別分析後，可提供決策者進行決策輔助。鑒於本研究以社群媒體文本挖掘與情感分析為研究範疇，故文獻回顧總共分為四個部分，首先回顧社群媒體文本挖掘及情感辨識，後續針對日常通勤滿意度及深度學習分類方式進行文獻彙整，以確立本研究之實證分析架構。

2.1 社群媒體文本挖掘

2.1.1　社群媒體

隨著 Web2.0 的概念的運用，虛擬社群的發展快速增加。Rheinggold (1993)認為虛擬社群是一種利用網路為媒介發展的人際關係，其間有足量的成員並有足量的討論、互動與情感上的交流。Ba (2001)也提出虛擬社群是以網路作為成員

之間相互連結的管道,社群成員間彼此會獲得情感支持、分享共同興趣與資訊等。社群網站(Social Network Sites, SNS)即是以這些特質為基礎發展，它提供了一個平台能讓使用者建立並維持彼此之間的關係，進一步形成社會網路，而在該網絡中使用者能主動與公開地使用該網站之服務 (Rao, Gao, and Ding, 2008)。Mayfield (2005)也提出：SNS 提供了比其他線上社群更直接的人際互動，過往線上社群多為資訊導向，提供成員與社群相關主題之資訊，而後成員才會彼此相互認識、交流，是一種由上而下的概念。但 SNS 卻相反，而是藉由較先公開的檔案資訊彼此互相交流，在締結社群，進而共享資源，是一種由下而上的概念。由此可知社群網站是 Web2.0 最具代表性的網站。

維基百科(Wikipedia)對社群媒體的定義為：人們用來創作、分享、交流意見、觀點及經驗的虛擬社區和網絡平台。社群媒體和一般的社會大眾媒體最顯著的不同是，讓用戶享有更多的選擇權利和編輯能力，自行集結成某種聽閱社群。社群媒體並能夠以多種不同的形式來呈現，包括文本、圖像、音樂與視頻。流行的社群媒體傳播介質包括了 blog、vlog、podcast、Wikipedia、Facebook、Instagram、plurk、Twitter、Google+、網絡論壇、Snapchat、LINE 等，某些網站也加入類似功能，例如百度、Yahoo! Answers、EHow、Ezine Articles 等。

近來社群媒體創造的評論吸引許多學者的關注，本研究針對社群媒體(Facebook、PTT、Blog 等)所提供的文本，進行語言處理、情感識別方面的探討。社群媒體系統提供簡單的介面，讓使用者發表具時間標記的文章，因此越來越多人使用社群在網路分享每天的生活經驗、發表對事物的看法與心情。根據社群搜尋引擎 Technorati 的報告指出，全球社群的數量已超過 7,000 萬個，並且平均每天有超過 12 萬個社群成立，因此整個社群網路每天所能貢獻出的新文本更在此數量之上。報告同時也指出，目前社群空間以日文及英文使用者居多，各占 37% 及 36%，而中文目前所占比例是 8%，但有增長的趨勢，本研究係以中文社群媒體之文本為研究對象。

在社群媒體的框架下，人們在使用社群搜尋引擎時，不但能找到較專業或具代表性的社群，同時也能找到一般使用者所提出的心得及想法。TREC 自 2006 年開始舉辦 Blog Track7 競賽，其競賽項目說明了使用者的資訊需求。Opinion Retrieval Task 針對特定議題找出使用者表達意見的文章，並判斷該意見文章的正負面傾向。舉例來說，使用者可能在一個著名的歌唱大賽結束後，瀏覽各社群媒體最新發表的相關文章，並挑有興趣的閱讀。使用者也可能剛接觸古典音樂，想找專門討論古典音樂的社群，並在確認某社的豐富的內容後，持續訂閱該社群。

2.1.2 文本挖掘

文本挖掘技術係應用知識挖掘技術，對存在於網路中的數據文本進行蒐集，並利用電腦系統辨識內容後加以歸類，藉由整理後的資料類別加以分析以獲得進一步的研究結果。一般而言，文本挖掘技術可概略分為文本蒐集、文本辨識、文本分類等三種。

1. 文本蒐集

文本蒐集是針對特定領域、議題、對象所蒐集的網路文本，一般多使用爬蟲系統，搜尋的基準是由使用者鍵入關鍵字進行搜尋，透過人工或是專家建置關鍵詞庫與關聯關鍵字集合，以此過濾爬蟲所蒐集之文本並刪除無相關文本，再依照分類加以儲存。

2. 文本辨識

文本辨識包含文本斷詞與特徵值轉換等過程，因中文文本具有構造複雜、一詞多義、一義多詞等性質，為使機器易於辨識，需由人工建立辨識規則，大多由人工檢視大量相關文本後，選出具有辨識性之詞彙，並依演算法賦予詞彙對應之數值，一般而言，詞彙集合稱為語料庫，詞彙對應之數值稱為特徵值。

3. 文本分類

文本分類是透過文本分類器對文本進行分類的過程，目前分類方法主要有以統計學方法為基礎的向量空間模型法、機器學習方法與以知識工程為基礎的方法。Pang et al. (2002) 與 Turney (2002) 分別提出了監督式學習與非監督學習兩種利用機器對特徵化後之文本分類與改進的方法，監督機器學習法的特色在於需要建立訓練集合，使機器透過對訓練集合的分類來獲得設置機器於分類時可用的參數，較常見方法有支持向量機 (Support Vector Machine, SVM) 與貝氏分類法；非監督機器學習法則無需訓練集合，係由機器透過演算法對文本集合直接進行分類，最常見的方法為 k 最近鄰居分類法 (k-Nearest Neighbors classification, k-NN)。

本研究係將陳亭愷(2015)及蔡易辰(2016)之文本挖掘流程及方法加以改良，應用知識挖掘技術對存在於網路中的意見文本進行蒐集，利用電腦辨識內容後進行詞率計算找出關鍵詞，在透過爬蟲獲取文本、比對語斷詞、給予權重後，接著利用文本分類演算法將文本加以分類。文本分類演算法是資料挖掘核心技術的一環，目的是將未歸類或已歸類的資料根據規則進行分類，此處所提及之分類不僅將文本分類至其所具有的特性下，計算過程中作為分類依據之相似度計算亦非常重要。在意見文本情感傾向性的研究中，為了獲得情感識別及情感傾向，即利用相似度計算將相似詞語、情感詞所構成之文本，透過歸類、分類至同一類別中。本研究加入人工智慧為建構模式之基礎，應用深度學習中演算法作為情緒分類方法，目的在於提高分類準確性。

2.2 情感識別

2.2.1 輿情定義

蔡易辰(2016)提到輿情為在一社會空間內，因某一社會事件的發生、發展以及變化，民眾在接觸到某社會事件後 民眾對社會管理者和事件所產生的社會政治態度與民眾之信，念、意見、態度與情緒等表現的集合。在傳統社會理論中，

輿情代表民眾對社會事件產生之綜合反應；社會輿情主要依靠報章雜誌、傳統通訊裝置等進行訊息傳遞，社會輿情具有傳遞速度、訊息擴散緩慢與受地理環境限制所影響等特性。網路輿情即為輿情反映在網路空間之現象，網路輿情相較社會輿情，網路輿情可不受地理環境、言論內容之多寡與訊息之接收能力等限制。

2.2.2 情感分析

情感分析(Sentiment Analysis)，也稱為意見挖掘 (Opinion Mining)，其目的是分析人們對某些有興趣的實體（例如，產品、服務、組織、事件等）的主觀看法，例如:意見、情感、評價等 (Liu, 2010, 2012; Pang and Lee, 2008; Feldman, 2013)，並將這些資訊依據評論的對象（稱為產品特徵）與表達的主觀意見（稱為意見傾向）適當地摘要彙整，轉換成結構化的知識，以協助決策者更加了解廣大使用者的意見，並以之發展重要的商業智慧應用，或提供一般使用者另一個訊息管道，避免被廠商官方資訊誤導，做出更佳的消費決策。Kim 與 Hovy(2004)對意見進行定義，意見由四個元素組成，分別為主題(Topic)、持有者(Holder)、陳述(Claim)與情感(Sentiment)。這四個元素之間存在之關聯，即意見的持有者針對某個主題發表之意見綜述帶有情感。

情感分析廣泛運用自然語言處理、計算語言學和文本挖掘，其判斷發言者或使用者對某些話題的態度，其態度係指發言者或使用者個人主觀判斷及評價、使用者發表言論當下的情感狀態，或使用者希望他人產生的情感效應。情感分析用來分類對立的文章或句子，進一步瞭解文章或內容段落所表達的意見為贊同(正面)、反對(負面)或中立情感。目前社群媒體輿情情感分析方法主要是針對民眾發表之評論文本進行分析，分析過程則多將社群媒體文本進行斷詞、特徵量化後，並加以分類之方式。在中文輿情情感分析研究中，情感分析方法與文本分類方法在流程上具有相似性，文本分類方法係將文本根據已建置之詞庫特徵量化、去噪後進行分類；而情感分析方法則導入情感詞後，將社群媒體文本根據情感詞特徵量化後進行分類，上述二者在現今處理社群媒體文本時多利用機器進行分析，故

需將文本轉化為數值使機器能進行判讀，因此使文本分類方法與情感分類方法產生在流程上相似度。情感分析方法在探討社群媒體文本之情感傾向性時，情感分析方法係以探討文本中情感詞的組成，故文本情感分析以傾向性分析居多。

情感分析依其使用的技術主要可區分為監督式(Supervised)與非監督式(Un-supervised)二種，監督式情感分析使用已標註之訓練資料(Training Dataset)訓練分類器，藉由所訓練出的分類器進行目標資料的情感分類。Pang 等人(2002)、Wiebe 等人(2004)、Melville 等人(2009)、Chiu 等人(2013)皆採取監督式學習方法進行文本之情感分析。非監督式情感分析則使用既有的情感辭典，或統計方法區別情感之正向、負向。Hu 和 Liu(2004)與 Cilibrasi 和 Vitanyi(2007)採用非監督式學習方法進行情感分析。Pang 等人(2002)透過詞性標註使文本具有情感傾向性，並藉由簡單貝氏分類器與支持向量機分類器，對網路上電影評論文本之情感傾向進行正向、負向的分類，其文本情感傾向性之正向與負向的識別的方法亦也沿用至今；Turney(2002)提出以非監督機器學習法計算文本中之正向、負向情感之關鍵詞與正向、負向情感中字詞間之相似度，透過其進行情感傾向分類，但經實驗證明成效不佳。因此目前網路輿情情感分析之相關研究，則以 Pang 等人(2002)利用監督機器學習方法進行文本之情感分類為主要方法，但由於監督式機器學習依賴大量人工標註的數據，使監督式機器學習需付出大量時間及金錢代價，因此本研究嘗試採用無監督式機器學習方法，以避免過度依賴人工調整參數，不僅能節省人工標註工作量，所建構模型亦具較佳之移轉性。

2.2.3 情感識別

情感識別為情感分析中一環，以往有關情緒分析相關研究，多將文本內容分成三元(正向、負向及中立)或二元(正向、負向)，三元情感分析有助於提高分類準確度，但能無法探討出其情緒反應(如：開心、悲傷、驚訝、恐懼等)，Carlo Strapparava 等人(2008)應用潛在語意分析(Latent Semantic Analysis, LSA)將文本情緒自動分成六類：憤怒、厭惡、恐懼、喜悅、悲傷和驚喜(Anger, Disgust,

Fear, Joy, Sadness, Surprise)，將 250 筆已標籤文本進行模式建構，以另外 1000 筆作為驗證。Mohamed Yassine 等人(2010)使用無監督式 K-means 集群演算法，將表情符號分成四類：開心、悲傷、玩笑、親密(Smiling, Sad, Joking, Kissing)，兩篇研究結果顯示皆有較高精確度。因此本研究將參考過去情感識別研究，改善蔡易辰(2016)三元情感分析模式，建構出更細粒之多元情感識別模型。

蕭乃沂等人(2015)研究指出，政府可導入社群媒體輿情分析並配合問卷調查，以利政策規劃。陳亭愷(2015)透過網路輿情分析對我國實施計程收費網路評論進行分析，了解民眾對於網路評論下之話題情感趨勢，可供決策者在制定決策與執行方面給予輔助。蔡易辰(2016)利用三元決策情感分析模型對網路評論文本進行分析，將社群媒體文本傾向分成正面、負面、中立，有利於決策者及時掌控重要議題。社群媒體文本資料與情感分析可提供管理者在管理及制定政策時給予輔助。

2.3 日常交通滿意度

消費者對於自己付費所想要得到的產品或服務，心中都有會一種隱藏的期待感或是預期心理，民眾搭乘大眾運具亦是如此，這種期待心理是否能獲得滿足，會影響消費者對其消費之商品或勞務的評價。Gronroos(1982)認為服務品質是消費者事前期望的服務與接受服務後的認知之比較產生。消費者在接受服務前有期望品質(expected quality)，接受服務後則會產生經驗品質(experienced quality)，兩者比較之後會得到一個差異值，稱之為總體認品質(total perceived quality)。如果經驗品質達到期望品質，則總體認知品質是好的，反之則是低劣的。

交通部「105 年民眾日常使用運具狀況調查分析」指標包括各運具市占率、使用族群、旅次目的、主要運具、公共運輸服務滿意度以及未使用公共運輸的原因等，調查「公共運輸服務滿意度」乃是希望能藉由了解民眾預期與實際感知服務的差距，進而針對公共運輸服務進行調整與改善，以達到提升公共運輸使用率

之目的。此問卷調查透過電話訪問方式，詢問受訪者調查日前一日(週一至週五)其所有的外出活動，其電訪中詳實記錄每個旅次及旅次所使用之運具。時間為 105 年 9 月 27 日至 12 月 31 日，對臺閩地區年滿 15 歲以上民眾進行電話訪談，有效樣本為 4 萬 246 人，在 95%信心水準下，抽樣誤差為± 0.49 個百分點。

105 年民眾外出旅次目的，以「通勤」43.8%最高，其次依序為「個人活動」16.3%、「購物」15.2%、「休閒」12.9%，及「通學」7.7%，如圖 2.1 所示。從圖中可得知超過一半的民眾在平日外出之目的為上班、上課，故本研究探討對象為平日通勤、通學、商務、業務民眾。

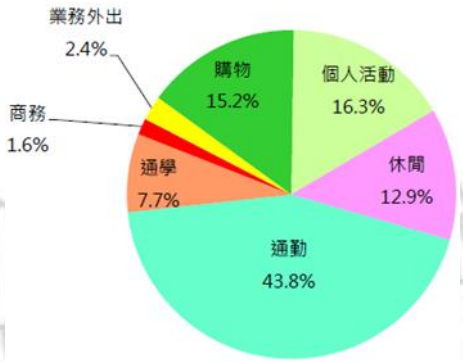

圖 2.1 民眾外出旅次目的占比

就主運具類別觀察，105 年民眾外出從事各項活動仍以使用「私人機動運具」為主，其中「通勤學」旅次使用私人機動運具占 76.6%，高於「休閒」使用私人機動運具占 52.7%，而公共運具在「商務」旅次占 23.8%最高，其餘依序為「休閒」18.1%，「通勤學」17.9%，如表 2.1 所示。由問卷結果可得知，公共運具使用率最高之旅次目的為「商務」及「通勤學」，故本研究將研究對象限定於搭乘公共運具且旅次目的為「業務外出」、「商務」及「通勤學」之相關社群媒體文本。

表 2.1 外出旅次目的之主運具類別占比

單位：%

運具類別	通勤學	商務	業務外出	購物	個人活動	休閒
總計	100.0	100.0	100.0	100.0	100.0	100.0
公共運具	17.9	23.8	9.6	7.7	15.2	18.1
非機動運具	5.5	2.0	0.6	16.2	7.7	29.2
私人機動運具	76.6	74.2	89.8	76.1	77.1	52.7

105 年日常有搭乘公共運具的民眾中，有高達 93.9%對公共運具之服務表示滿意，整體公共運具服務之滿意度維持在 9 成以上，如圖 2.2 所示，顯示政府及各運輸業者對於提升服務品質之努力受到民眾認同，但問卷內容僅能調查出民眾對公共運具之整體滿意度，無法了解民眾預期與實際感知服務的差距，故本研究以社群媒體文本挖掘方式，探討民眾對公共運具之預期與實際服務的差距。

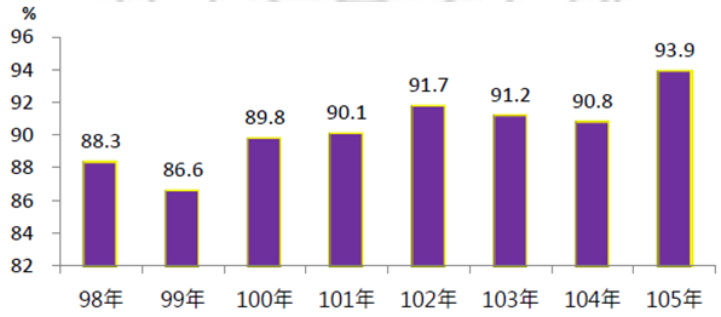

圖 2.2 民眾對於搭乘公共運具服務的滿意度

按運具別觀察，105 年「飛機」服務滿意度最高 99.4%，「計程車」、「渡輪」、「交通車」之服務滿意度均在 95%以上，其餘公共運具服務滿意度皆在 87%以上，如表 2.2 所示。本研究聚焦於市占率高且合理之通勤通學公用運具，故將「計程車」、「交通車」、「捷運」、「高鐵」、「公路客運」、「市區公車」、「國道客運」、「臺鐵」納入研究範圍，其他運具則暫不討論。

表 2.2 民眾搭乘各項公共運具之滿意度

單位:%

運具別	市占率	滿意	不滿意
總計	18.1	93.9	4.3
飛機	0.1	99.4	0.1
計程車	2.2	97.5	2.0
渡輪	0.1	95.8	4.2
交通車	1.1	95.7	3.2
捷運	5.1	94.7	2.7
高鐵	0.4	94.6	5.4
公路客運	0.7	94.2	5.5
市區公車	6.3	93.3	5.4
國道客運	0.6	89.8	9.0
免費接駁公車	0.2	89.5	0.3
臺鐵	1.4	87.5	7.1

2.4 深度學習

2016 年 3 月 AlphaGo 以四比一擊敗韓國職業棋士李世乭後，AlphaGo 所使用的深度學習技術引起了各界的關注。事實上早在 AlphaGo 問世之前，深度學習技術即已廣泛應用在各個領域。當我們對 iPhone 的語音助理軟體 Siri 說一句話，Siri 可以將聲音訊號辨識成文字，用的就是深度學習的技術；當我們上傳一張相片到 Facebook，Facebook 可以自動找出相片中的人臉，用的也是深度學習的技術。

深度學習為機器學習(Machine Learning)一支，「機器學習技術，就是讓機器可以自我學習的技術。」但實際上機器是如何學習？一言以蔽之，機器學習就是讓機器根據一些訓練資料，自動找出有用的函數(function)。例如將機器學習技術運用在文字辨識系統，就是要機器根據一堆文字訊號和其對應的文字，找出如圖 2.3 的「文字辨識函數」：

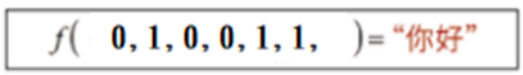

圖 2.3 文字辨識函數表示方式 (李弘毅)

輸入一段文字訊號，輸出就是該文字訊號所對應的文字；如果機器學習技術應用在影像辨識系統，那就是要機器根據一堆圖片和圖片中物件名稱的標註，找出「影像辨識函數」，如圖 2.4 所示：

圖 2.4 圖片識函數表示方式 (iker)

輸入一張圖片，輸出試圖片的物件名稱。以上要找的函數，共通點是它們都複雜到人類沒有能力寫出它們的數學式，只有靠機器才有辦法找出來。

傳統主流的自然語言處理(Natural Language Processing,簡稱 NLP)是基於統計學的機器學習方式。近年來隨著深度學習技術有突飛猛進的發展，應用於自然語言處理領域也獲得比傳統模型有更好的結果，本研究以深度學習演算法為主要的機器學習方法，因此將整理有關深度學習的相關文獻。

2.4.1 人工神經網路

根據 Balakrishnan et al.(1994)引述：類神經網路(Artificial Neural Network, ANN)或譯為人工神經網路，是指模仿生物神經網路的資訊處理系統。葉怡成(2003)認為類神經網路較精確的定義為：「一種計算系統，包括軟體與硬體，它使用大量簡單的人工類神經原來模仿生物神經網路的能力。人工神經元是生物神經元的簡單模擬，他從外界環境或者其他人工類神經元取得資訊，並加以運算，在輸出其結果到外界或者其他人工神經元。」

在人工神經元結構中類神經網路是由多個人工神經細胞(artificial neuron)所組成，人工神經細胞又稱為處理單元(processing element)，每一處理單元可接受一個或數個輸入訊號，然後經過中間層產生一個輸出訊號，輸出值與輸入值之間

關係可用下列函數表示:

$$\alpha = \sigma(\sum_{k} a_k w_k + b)$$

一個典型的類神經網路是由“層”所組成。最低階層是一輸入層，資料透過層內的神經元輸入至網路類神經，而最頂層是輸出層，其產生將由使用者解譯。輸入與輸出還會有一個或多個層，我們稱之為隱藏層。第一隱藏層的輸出傳送至下一隱藏層，直到訊號到達輸出層為止，如圖 2.5 與圖 2.6 所示。

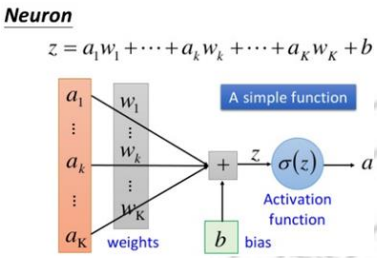

圖 2.5 神經元結構

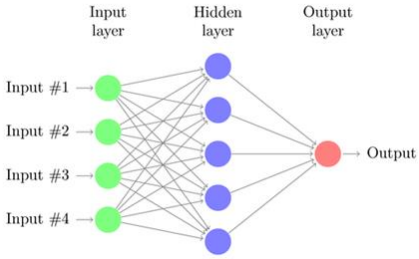

圖 2.6 人工神經架構

2.4.2 深度學習

深度學習的概念最早由多倫多大學的 G. E.Hinton 等於 2006 年提出，指基於樣本數據通過一定的訓練方法得到包含多個層級的深度網路結構的機械學習過程。傳統的神經網路隨機初始化網路中的權值，導致網路很容易收斂到局部最小值，為解決此問題 Hinton 提出使用無監督預測訓練方法優化網路權值的初值，再進行權值微調的方法，拉開了深度學習的序幕。

深度學習概念源於人工神經網路的研究，近年來在語音文字識別、圖片辨識等多應用中取得突破性的發展，其優勢在於建立模型模擬人類大腦的神經連接結構，在處理圖像、聲音和文本訊號時，透過多個變換階段分層對數據特徵進行描述，進而解釋數據。以圖像數據為例，人類的視覺系統中對這類訊號的處理依次為：首先檢測邊緣、初始形狀，然後再逐步形成更複雜的視覺形狀，深度學習透

過组合低層特徵形成更加抽象的高層表示、屬性類別或特徵，以表示數據的分類特徵。深度學習之所以被稱為“深度”，是相對支援向量機、提升方法(Boosting)、最大熵方法等“淺層”學習方法而言，深度學習所學得的模型中，非線性操作的層數(隱藏層超過四層)更多。淺層學習依靠人工經驗抽取樣本特徵，模型學習後獲得是沒有次結構的單層特徵；而深度學習透過對原始資料進行逐層特徵轉換，將樣本在原空間的特徵表示變換到新的特徵空間，自動的學習得到層次化的特徵表示，而更有利於分類或特徵的可視覺化。

深度學習所得到的深度網路結構包含大量神經元(Neural)，每個神經元與大量其他神經元連接，神經元間的連接强度(權值)在學習過程中修改並決定網路的功能。透過深度學習得到的深度網路結構符合神經網路的特徵，因此深度網路就是多層次的神經網路，即深度神經網路(Deep Neural Networks, DNN)。深度神經網路是由多個單層非線性網路疊加而成，常見的單層網路按照編碼解碼情況分為三類：只包含編碼器部分、只包含解碼器部分、既有編碼器部分也有解碼器部分。編碼器提供從輸入到隱含特徵空間的自底向上的映射，解碼器以重建结果盡可能接近原始輸入為目標將隱含特徵映射到輸入空間。

深度神經網路分為以下三類:

1. 前饋式深度網路(Feed Forward Deep Networks, FFDN)，由多個編碼器層疊加而成，如多層感知器(Multilayer Perceptions, MLP)、卷積神經網路。

2. 回饋式深度網路 (Feedback Deep Networks, FBDN),由多個解碼器層叠加而成，如反卷積網路(Deconvolution networks, DN)、層次稀疏編碼網路(Hierarchical Sparse Coding, HSC)、遞歸神經網路(Recurrent Neural Network, RNN)。

3. 雙向式深度網路(Bidirectional Deep Networks, BDDN),透過疊加多個編碼器層和解碼器層構成(每層可能是單獨的編碼過程或解碼過程，也可能既包含編碼過程也包含解碼過程)，如深度玻茲曼機(Deep Boltzmann Machines,

DBM)、深度信念網路(Deep Belief Networks, DBN)、堆疊式自動編碼器(Stacked Auto-encoders, SAE)。

本研究採用類神經網路中，前饋式深度路網(FFDN)之卷積神經網路(CNN)作為分類演算法。

2.5 小結

從「105 年民眾日常使用運具狀況調查分析」可得知，市占率高且合理之通勤學公共運具服務滿意度雖在 87%以上，但仍常耳聞民眾抱怨公共運具，在無法了解民眾預期與實際感知服務的差距下，因此具有蒐集並分析社群媒體意見之必要性。

本研究係針對日常交通下社群媒體進行文本挖掘與情感識別，透過建置社群媒體文本資料庫，並將日常交通之社群媒體文本進行情緒分類以瞭解民眾對於日常交通下各主題評論之情緒。以人工檢閱文本內容並結合網路情感辭典建置情感詞庫，所蒐集之日常交通文本的特徵詞，將透過詞頻演算法計算出情感值，並建構多元情緒決策情感分析模型，透過情感識別模型將情緒分類，達到日常交通下社群媒體輿情情感分析目的。

本研究係參酌陳亭愷(2015)、蔡易辰(2016)之情感分析流程並加以改良，與前兩者研究有三個相異處:

1. 文本分類執行流程較前兩者研究減少大量人工檢視時間。
2. 提供多元情感識別，較前兩者之二元與三元決策分析，細粒度更高。
3. 應用深度學習演算法，提高情感識別準確度。

第三章　研究方法

本章節將分別說明本研究方法，包含：斷詞系統、詞頻演算法、卷積神經網路演算法、K-means 集群演算法。

由文獻回顧得知，文本挖掘可概括為四個步驟：資料收集、資料處理、資料轉換、文本分類，且過去文本分類過程涉及多種演算法與軟體方可達成，如：資料抓取、斷詞系統、清理資料、詞頻計算、特徵質量化等，本研究改善過去文本分類方法及簡少大量人工檢視時間，情感分析模式建構流程如圖 3.1 所示。

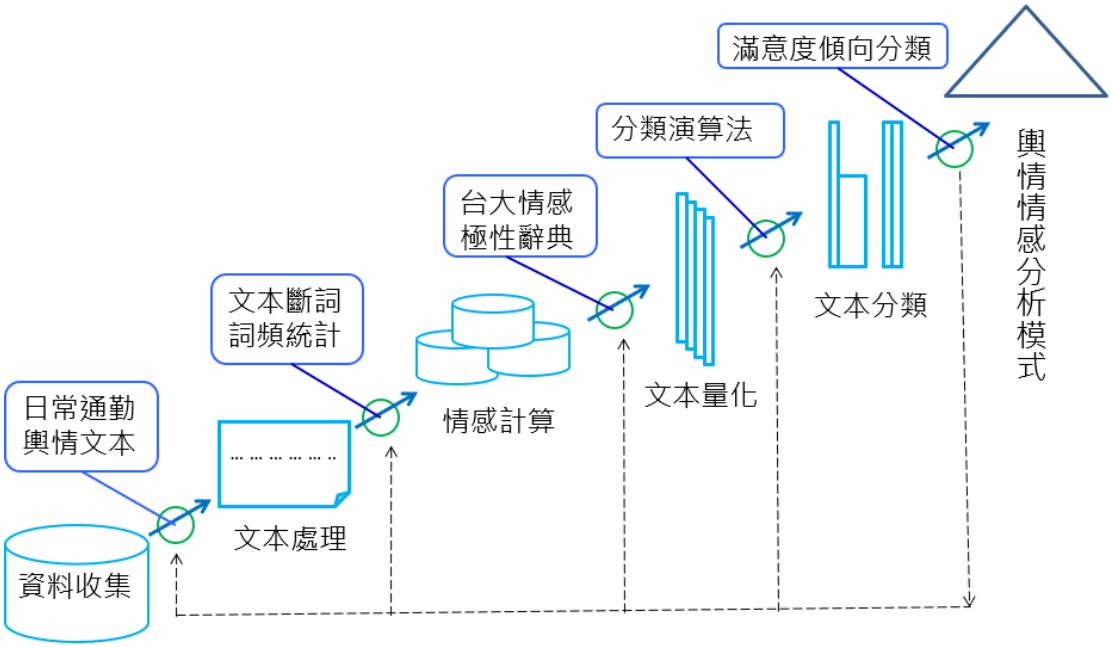

圖 3.1 情感分析模式建構流程圖

3.1 文本斷詞系統

詞是最小有意義且可自由使用的語言單位，在中文的自然語言處理問題中必須先能分辨出文中的詞才能進行後續處理，因此本研究使用 Python3.6 版中開放資源，結巴(jieba)斷詞套件，首先將自社群媒體蒐集之文本資料進行斷詞處理，斷詞完成會顯示總斷詞詞彙數與斷詞矩陣，矩陣及透過詞頻演算法得出。

jieba 中文斷詞所使用的演算法是基於 Trie Tree 結構去生成句子中，中文字所有可能成詞的情況，然後使用動態規劃(Dynamic programming)算法來找出最大機率的路徑，這個路徑就是基於詞頻的最大斷詞結果。對於辨識新詞(字典詞庫中不存在的詞)則使用隱馬可夫模型(Hidden Markov Model)及 Viterbi 算法辨識出來，如圖 3.2 所示。

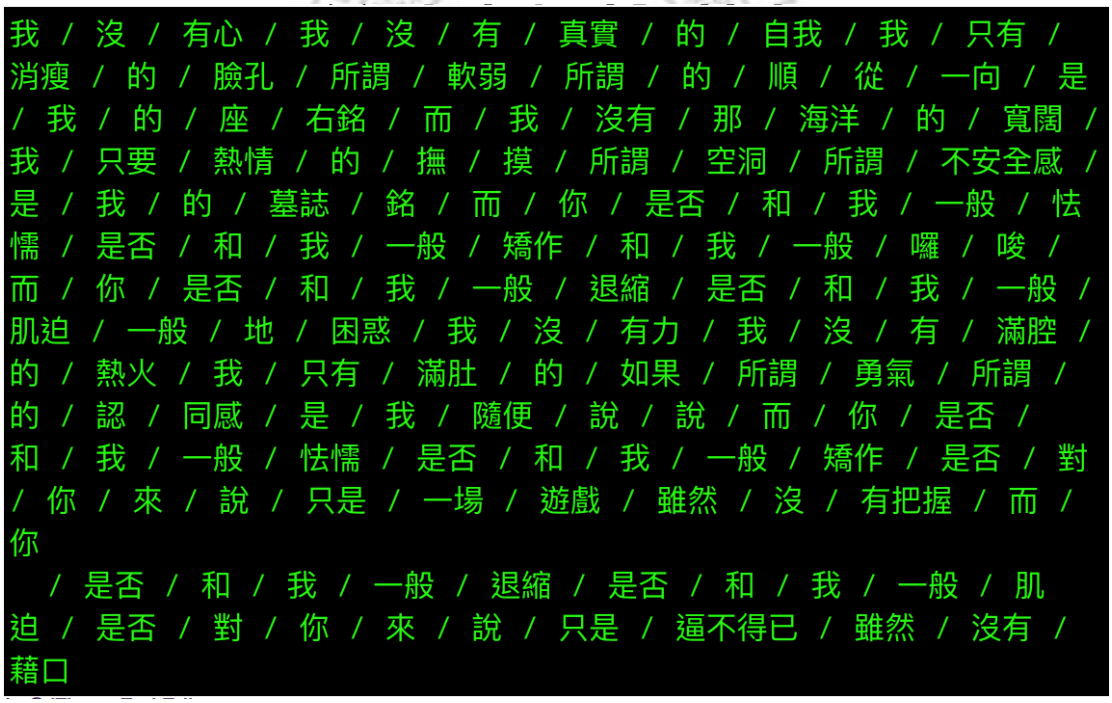

圖 3.2 結巴斷詞成果

3.2 詞頻演算法

為使電腦得以辨識文本構成，需將文本依其所具有之特性轉換為數值，透過斷詞系統對詞彙出現之數量進行統計，並決定適合表示文本特性之詞彙，以此表現文本所對應之分類性質。本研究使用卷積神經網路演算法，再提取特徵時不需要計算 TF-IDF，特徵值透過機器自動學習並產生矩陣，TF-IDF 特徵量化將用於 K-means 集群。

詞頻演算法是一種將文本進行特徵量化的方法，常用於資訊檢索與文字挖掘的加權技術，本研究將使用 Luhn 及 Jones 所提出的詞頻法與逆向檔案頻率法 (Term Frequency–Inverse Document Frequency, TF-IDF)，處理文本特徵量化，Luhn 所提出之詞語頻率演算法是透過計算詞彙在文本中出現的頻率來體現詞彙所具備之重要性，公式如下：

$$TF_{ij} = \frac{n_{ij}}{\sum_k n_{kj}}$$

TF_{ij}表示特徵詞 i 在文本 j 中出現的次數，$\sum_k n_{kj}$則表示文本 j 中所有出現詞彙的次數總和，詞語頻率(term frequency)演算法在決定特徵量上擁有計算快速且簡單的優點，但缺點則是難以排除低相關高頻率之無用詞彙以及抓取低出現率但高重要性之詞彙，容易因納入過多不相干詞降低整體特徵項與特徵量的辨識能力，因此需要透過人工檢視並刪除不相關詞彙，以提高準確度。

詞語頻率演算法與逆向文本頻率演算法在面對文本特徵項與特徵量的決定上皆尚有不足，透過詞頻-逆向文本頻率(TF-IDF)演算法，若詞彙在某文本出現頻率越高，但在整體文章集合出現頻率較低時，即可表現該詞具有特殊性，故給予較高權重，其公式如下：

$$IDF_i = \log \frac{|D|}{|\{j: t \in d_j\}|}$$

$|D|$語料庫中的檔案總數；$|\{j: t \in d_j\}|$包含詞語t_i的檔案數目(即$n_j \neq 0$的檔

案數目)如果詞語不在資料中，就導致分母為零，因此一般情況下使用 1+|$\{j: t \in d_j\}$|，最終 TD-IDF 法公式如下：

$$\text{TF} - \text{IDF} = TF_{ij} \times IDF_i$$

某一特定檔案內的高詞語頻率，以及該詞語在整個檔案集合中的低檔案頻率，可以產生出高權重的 TF-IDF。因此 TF-IDF 傾向於過濾掉常見的詞語，保留重要的詞語。

舉例來說：一篇文本檔案的總詞語數為 100，而詞語「捷運」出現了 40 次，那「捷運」一詞在該文本檔案中的詞頻(TF)就是 40/100=0.4。文本頻率是計算有多少份文本出現過「捷運」一詞，除以文本裡包含的文本總數。如果「捷運」一詞在 1,000 份文本出現過，而文本總數是 10,000,000 份的，其逆向檔案頻率(IDF)就是 log（10,000,000 / 1,000）=4。最後的 TF-IDF 的分數為$0.4 \times 4 = 1.6$。

3.3 卷積神經網路

卷積神經網路(CNN)又被稱為 CNNs 或 ConvNets，它是目前深度神經網路（Deep Neural Network, DNN）領域的發展主力，在圖片辨別上甚至可以做到比人類還精準的程度，由文獻回顧可知 CNN 能處理多種型態資料，只要將輸入(input)格式轉換成類似圖片的形式，CNN 能有卓越的研究成果，因此本研究採用此演算法來進行文本情感識別分類。

典型的 CNN 架構如圖 3.3 所示：包含輸入(input)、卷積層(convolution)、池化層(pooling)、激活函數(activation function)、全連接層(full connected layers)、輸出(output)。

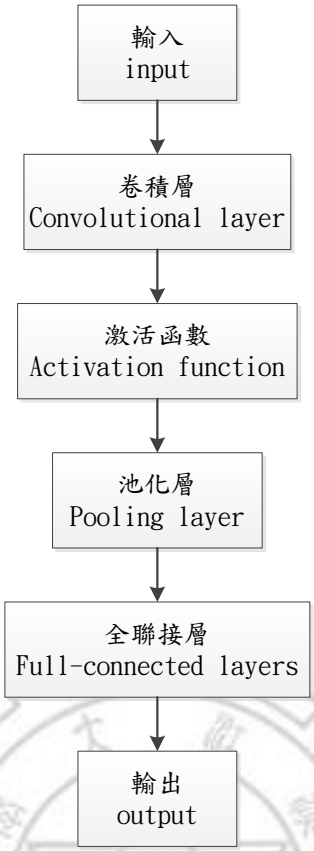

圖 3.3 CNN 基本架構

1. 輸入(input)

首先從文本訓練集中提取出每條文本的特徵矩陣，先將文本每條句子 T 進行斷詞，在對整個文本集進行分詞的同時，CNN 建立了文本集的辭典 D，並將 D 中所有的詞表達初始化，補充到 D 中，後將 T 作為分詞工具的輸入，輸出的 T 被分為 d 個詞語w_1， w_2，w_3，…，w_d將 T 表示為:

$$\mathrm{T}:\{w_1,\ w_2,\ w_3,\ \ldots,\ w_d\}$$

從 D 中獲取 T 的每個詞語w_i的詞表達，將w_i表示為$R^{V\times 1}$空間中的向量w_i:

$$w_i:\{m_{i1},\ m_{i2},\ m_{i3},\ \ldots, m_{iv}\}\ i\in\{1,d\}$$

T 矩陣表示即是將 T 中所有詞語的詞表達按照詞序列從上至下排列起來:

$$w_{i:d}=w_1+w_2+w_3+\cdots w_d$$

將文本 T 轉換成$R^{V\times d}$特徵空間的矩陣:

$$T = \begin{bmatrix} m_{11} & m_{12} & \cdots & m_{1v} \\ m_{21} & m_{22} & \cdots & m_{2v} \\ \vdots & \vdots & & \vdots \\ m_{d1} & m_{d2} & \cdots & m_{dv} \end{bmatrix}$$

將 T 的特徵矩振作為 CNN 的輸入，然後通過 CNN 的卷積層、池化層進行特徵提取，獲取文本的句子表達。

2. 卷積層(convolution)

CNN 應用在圖像識別時，在卷積層的操作中，卷積核在像素矩陣的行列兩個方向都發生移動，將 CNN 應用到分類時，以前述 T 的特徵矩陣作為輸入，卷積核在行方向的移動不具解釋性，在這裡，輸入到 CNN 中的句子表達矩陣，在列方向上保留了文本的語序訊息，卷積核在列方向上的移動可以獲取到原文本的固有特徵，該特徵的大小也是由原文本語法特徵決定的，無需人工干預。

CNN 在卷積層中設置了 m 個卷積核 $C_1, C_2, C_3, \ldots, C_m$。設置多個卷積核是為了能夠更加全面地獲取到句子表達的特徵，降低特徵提取過程的偶然性。其中任意卷積核 $C \in R^{k\times v}$：

$$C = \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1v} \\ c_{21} & c_{22} & \cdots & c_{2v} \\ \vdots & \vdots & & \vdots \\ c_{k1} & c_{k2} & \cdots & c_{kv} \end{bmatrix}$$

卷積層的操作是將文本的特徵矩陣 T 的從上至下 d-k+1 個子矩陣 $T_P \in R^{k\times v}$,

$P \in \{1, d-k+1\}$，如下所示，分別與 C 進行運算。

$$T_p = \begin{bmatrix} m_{p1} & m_{p2} & \cdots & m_{pv} \\ m_{p+1,1} & m_{p+1,2} & \cdots & m_{p+1,v} \\ \vdots & \vdots & & \vdots \\ m_{p+k-1,1} & m_{p+k-1,2} & \cdots & m_{p+k-1,v} \end{bmatrix}$$

CNN 的卷積層實際上是一個特徵抽取的過程，一個卷積核抽取一種特徵，得到一個特徵矩陣。CNN 在抽取某一種特徵時，通過相同的卷積層核對原始輸入的不同進行相同的訊息轉換，將局部的特徵簡化，保留了整體的特徵。

3. 池化層(pooling)

文本 T 的句子表達經由 m 個卷積核進行卷積操作後，從卷積層傳遞了 m 個 $R^{(d-k+1)\times 1}$ 空間矩陣表達的特徵。池化層將這些特徵進一步聚合，簡化特徵的表達，在池化層定義了池計算：

$$\text{pooling} = \left(S_{(d-k+1)\times 1}\right) = \alpha(S_1, S_2, S_3, \dots, S_{d-k+1})$$

其中 α 是可選擇的，常見的為最大化(Max pooling)、最小化(Min pooling))、平均值等。一般而言，α 需要通過對比實踐為當前的應用選擇最適合的。

池化操作將每個卷積矩陣轉換為一個一维特徵值，得到一個 $R^{m\times 1}$ 空間的特徵向量 P：

$$\text{p} = (P_1, P_2, P_3, \dots, P_M)$$

CNN 的卷積層和池化層分別通過卷積操作和池化操作對文本 T 的句子表達進行了特徵提取，得到簡化後的特徵向量 P，後 CNN 將該特徵向量傳遞到分類氣中，計算文本 T 所屬的情緒分類。

4. 激活函數(activation function)

在上述各層操作完成後，皆會經過激活函數作用，若不使用激活函數，神經網路只能構建線性分類器且有梯度爆炸問題，無論使用多少層神經網路，輸出都是輸入的線性組合，與只有一個隱含層效果相當。常見的激活函數有 Sigmoid 函數、tanh 函數、線性整流函數(Rectified Linear Unit, ReLU)函數等，其函數形式和圖像如圖 3.4 所示。根據文獻回顧得知 ReLU 為人工神經網路常用的激活函數，且較其他函數有更好的效果。

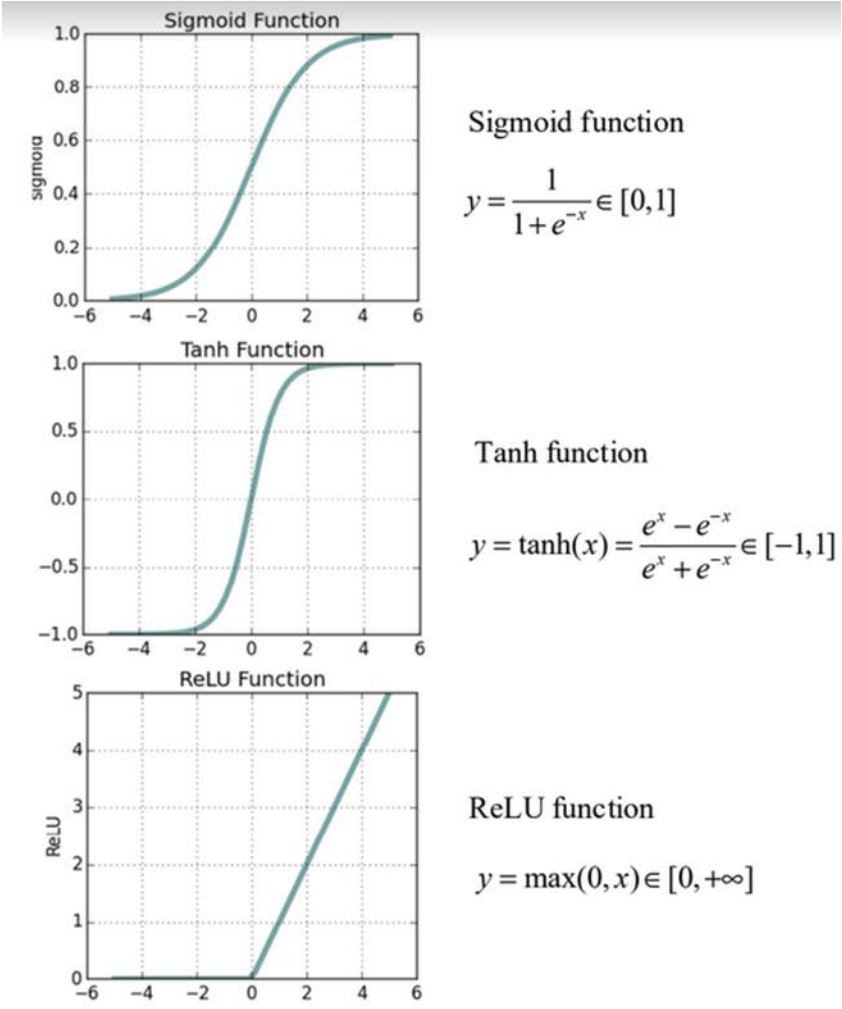

圖 3.4 常用激活函數及其函數圖像

5. 連接層(full connected layers)

經由上述卷積層、池化層、激活函數不斷重複運算，使輸入資料不斷降維簡化，可得到 N 個降維後之輸入，利用倒傳遞神經網路(Back-Propagation Neural Network, BPNN)演算法進行全連接層，計算誤差後再更新權重，則可得到最後解，圖 3.5 為傳遞神經網路架構。

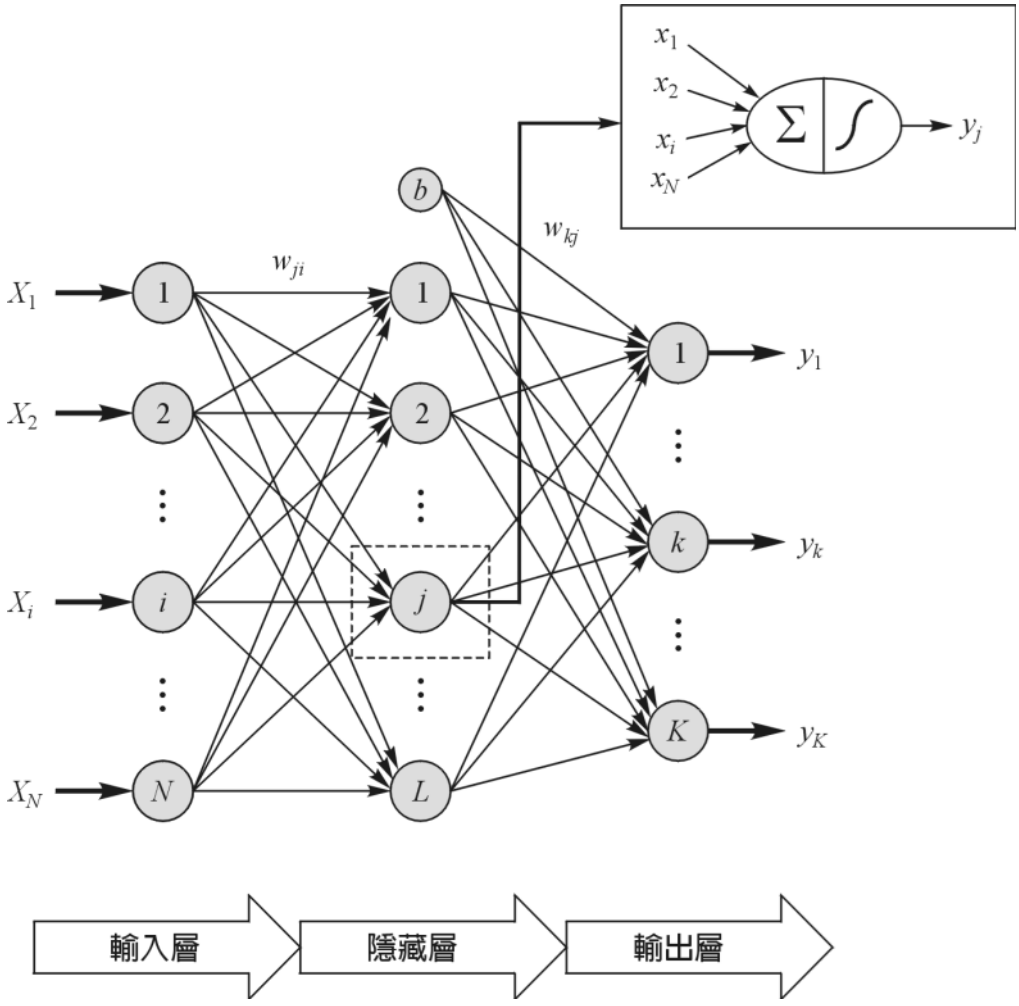

圖 3.5 傳遞神經網路架構

學習的原理是利用誤差(目標輸出值{Target Output}和實際輸出值{Actual Output}的差)來調整加權值。假設訓練資料集(Training Data Set)只有一筆資料，我們首先定義誤差衡量值(Error Measure)為：

$$E = \frac{1}{2}\sum_{i=1}^{m}(t_i - o_i)^2$$

t_i 是目標輸出值，o_i 是實際輸出量，為了要 E 使最小，而定義加權數值改變量如下：

$$\Delta w_{ij} = -\eta \frac{\partial E}{\partial w_{ij}}$$

w_{ij} 是神經元 i 到神經元 j 的加權值，η 是學習率{Leaming Rate}，其值通常介於[0,1]間。上式表示在負的梯度方向上有最小值。第二層的輸出值可表示為：

$$o_i = f(net_i)$$

其中

$$net_i = \sum_{j=1}^{n} w_{ji} x_j$$

E 所以對 w_{ji} 的梯度值為：

$$\frac{\partial E}{\partial w_{ji}} = \frac{\partial E}{\partial net_i}\frac{\partial net_i}{\partial w_{ji}}$$
$$= \frac{\partial E}{\partial net_i} x_j$$

其中

$$\frac{\partial E}{\partial net_i} = \frac{\partial E}{\partial o_i}\frac{\partial o_i}{\partial net_i}$$
$$= -(t_i - o_i) f\ (net_i)$$

因此可求得

$$\frac{\partial E}{\partial w_{ji}} = -(t_i - o_i) f\ (net_i) x_j$$

即

$$\frac{\partial E}{\partial w_{ij}} = -(t_j - o_j) f\ (net_j) x_i$$

加權值的學習過程，將原來的加權值加上其改變量 $w_{ij} = w_{ij} + \Delta w_{ij}$

3.4 K-means 集群演算法

K-means 法(MacQueen, 1967)是分析集群問題時最常見的方法之一，以歐氏距離(Euclidean distance)，$d(X,Y) = \sqrt{\sum_p (X_p - Y_p)^2}$計算觀察值間之距離，其核心概念為假設將資料集分為 k 個集群，給定每個集群一個中心點，將各點與其最接近的中心點歸為同一個集群，以迭代的方式反覆運算移動各集群之中心點，最終目標為最小化各集群之離均差平方和，運算步驟如下：

Step1:將 n 筆資料$\{X_1, X_2 \cdots X_n\}$分為 K 個初始集群。

Step2:將資料分配到距離最近之集群(以歐幾里得法計算)，重新計算各集群之平均值(新的中心點)。

Step3:以 step2 計算出之新中心點重複執行步驟 2，直至中心點不再移動。

$$\min Z = \sum_{j=1}^{k} \sum_{i=1}^{n} \left\| X_i^j - C_j \right\|^2$$

X_i^j表示第 i 筆資料屬於第 j 集群

C_j表示第 j 集群之中心點

$X_i^j - C_j$表示第 i 筆資料與其所屬集群之中心點的距離差

K-means 之優點在於操作簡單，在多個統計軟體上皆可執行，如 WEKA、SPSS)、用於較大的資料集合也能夠有很好的效率，以及較高的可擴充性等，缺點是不能處理非球形數據或數據大小和密度不同的分群、不能處理包含離群值的資料集合，需要 Outlier Detection 配合、K-means 要求資料集合需要有中心的概念。

CNN 僅能訓練及驗證模式好壞，無法探討影響日常交通滿意度之因素，因此本研究最後使用 K-means 方法。

第四章 實證分析

本研究之實證分析流程，首先確認社群媒體文本資料型態，接著說明 CNN 文本分類及模型建構流程，最後根據 K-means 集群結果解釋影響因素，期能建立一套快速分析社群媒體日常交通滿意度之情感分析流程，供決策者提出改善方案及決策輔助之用。

4.1 實證分析流程架構

本研究係以日常交通滿意度社群媒體文本進行處理與分析，首先透過爬蟲系統抓取社群媒體文本，將抓取回來之資料進行情感分析前處理，如：中文斷詞、詞頻計算、卷積核矩陣設計、文本情感值計算、情感值區間設計，處理完畢後將文本資料帶入 CNN 進行模型建構。為了觀測模型是否與其測試結果相符，將帶入少量新資料進行比對驗證，確定模式可以使用後，再將資料以 K-means 集群分群，探討影響日常交通滿意度之因素，如圖 4.1 所示。

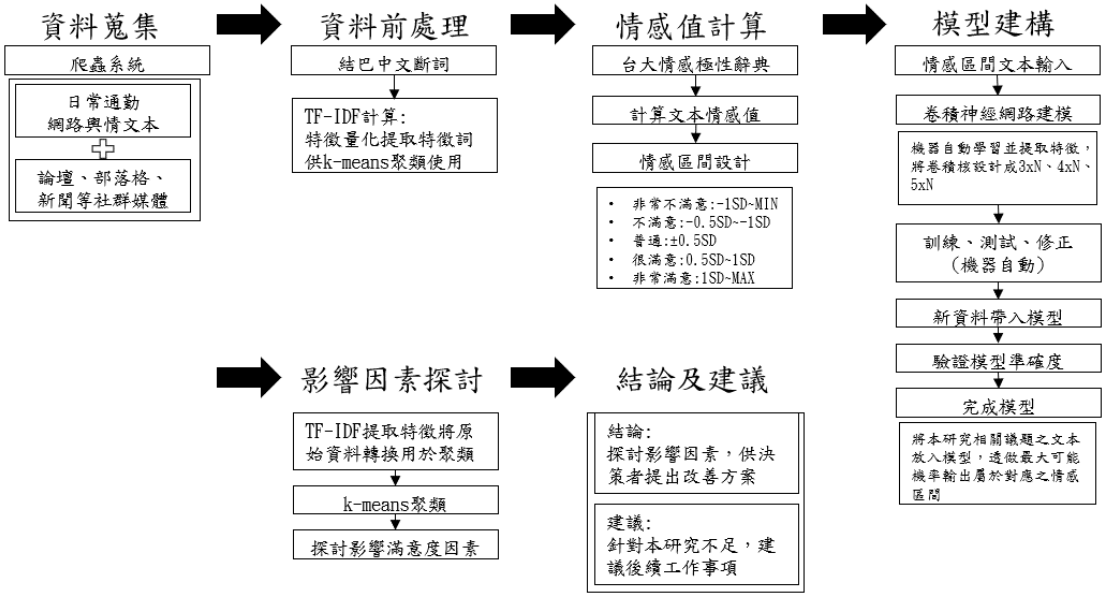

圖 4.1 實證分析流程圖

4.2 爬蟲流程

4.2.1 爬蟲流程說明

此爬蟲程式執行係以一層一層收斂資料，首先開啟新專案自訂名稱後，第一步驟為輸入第一層關鍵詞彙，本研究主題為探討日常交通滿意度，透過文獻回顧結果，第一層輸入目的相關詞彙如：上學、上課、通學、下課、放學、上班、通勤、下班、商務、業務外出等詞，如圖 4.2 所示。

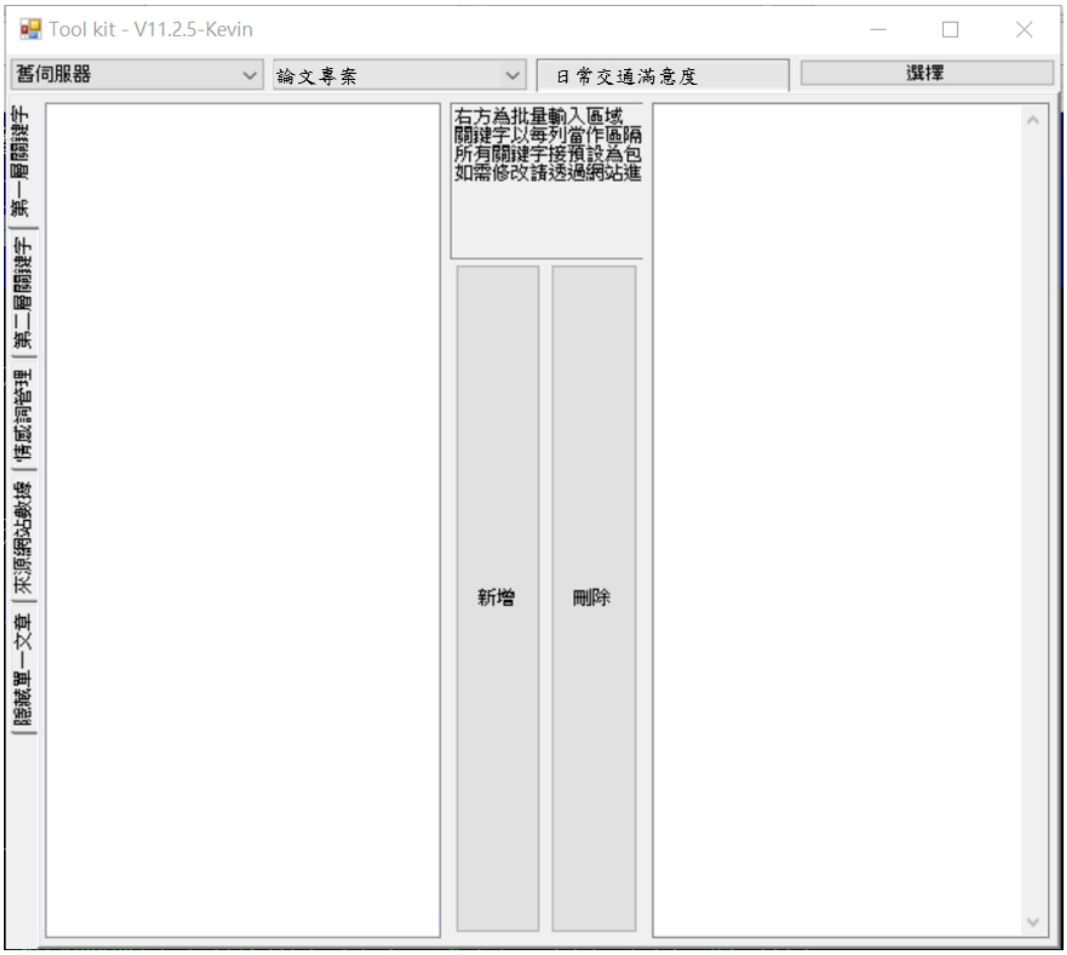

圖 4.2 爬蟲程式(第一層關鍵詞)

第二步驟為輸入第二層關鍵詞彙，透過文獻回顧結果，第二層輸入運具相關詞彙：捷運包含(北捷、台北捷運、高捷、高雄捷運、機捷、機場捷運、桃捷)、火車包含(火車、高鐵、區間車、自強號、台鐵、電聯、復興號、普悠瑪、莒光號、太魯閣號、普快車)、公車客運包含(公車、客運、國道客運、公路客運)、其他包含(計程車、交通車、校車)等詞，如圖 4.3 所示。

圖 4.3 爬蟲程式(第二層關鍵詞)

第三步驟為輸入情感詞，本研究使用台灣大學情感極性辭典(NTUSD)之情緒詞作為依據，將正面 2810 個詞彙；負面 8274 個詞彙，總共 11084 個情緒詞彙輸入，如圖 4.4 所示。

圖 4.4 爬蟲程式(輸入情緒詞)

最後爬取結果如圖 4.5 所示，有爬取來源、分類版、日期、網址等資訊，本研究僅取時間日期、標題、內文作為分析資料。

SiteName	Description	Article	Reply	ArticleStartD...	ArticleEndDate	Url
自由評論網	評論	13	0	2017/04/15 ...	2018/01/20 ...	http
關鍵評論網	政治	8	0	2017/08/04 ...	2018/01/30 ...	http
PTT	DPP版	4	0	2017/02/09 ...	2017/04/01 ...	http
PTT	北投板	18	1	2017/01/04 ...	2017/11/06 ...	http
PTT	男生板	9	85	2017/06/10 ...	2018/02/15 ...	http
PTT	公車板	6	54	2017/01/10 ...	2018/03/19 ...	http
PTT	嘉義板	1	2	2017/08/07 ...	2017/08/07 ...	http
PTT	烹飪板	1	4	2017/06/18 ...	2017/06/18 ...	http
PTT	軟體工作板	5	42	2017/06/15 ...	2017/09/22 ...	http
PTT	巴士時刻板	95	23	2017/01/04 ...	2018/03/18 ...	http
PTT	三籃板	6	16	2017/02/06 ...	2017/09/22 ...	http
PTT	貸款板	4	0	2017/06/26 ...	2017/12/27 ...	http
PTT	耳機板	1	87	2017/06/19 ...	2017/06/19 ...	http
PTT	精打細算板	11	264	2017/01/18 ...	2017/10/06 ...	http
PTT	女孩板	41	1,457	2017/01/20 ...	2018/03/09 ...	http
PTT	實況主板	2	0	2017/02/09 ...	2017/11/23 ...	http
PTT	林口板	1	0	2018/01/26 ...	2018/01/26 ...	http
PTT	恨板	8	158	2017/05/27 ...	2017/12/06 ...	http
PTT	媽媽寶寶板	46	648	2017/05/04 ...	2018/03/13 ...	http
PTT	重機板	2	17	2017/04/04 ...	2017/04/29 ...	http
PTT	捷運板	9	503	2017/06/13 ...	2018/02/22 ...	http
PTT	美妝板	2	0	2017/07/13 ...	2017/10/20 ...	http
PTT	公共議題板	13	365	2017/05/07 ...	2017/12/01 ...	http
PTT	陸劇版	1	0	2017/03/13 ...	2017/03/13 ...	http
PTT	水族板	1	0	2017/02/23 ...	2017/02/23 ...	http
PTT	新竹板	23	75	2017/01/10 ...	2018/03/08 ...	http
PTT	台劇版	2	5	2017/03/29 ...	2017/09/22 ...	http
PTT	花蓮板	10	107	2017/01/22 ...	2018/01/07 ...	http
PTT	臺南板	33	148	2017/01/04 ...	2018/03/10 ...	http
PTT	新店板	6	8	2017/01/07 ...	2017/12/02 ...	http
PTT	台灣綜藝板	3	233	2017/05/17 ...	2017/10/12 ...	http

圖 4.5 爬蟲程式(結果)

4.2.2 資料內容說明

本研究自 2017 年 1 月 1 日至 10 月 31 日，針對各大新聞網站、論壇、討論版等社群媒體抓取文本資料，共 1104 篇文本，包含：公車類 377 篇、火車類 376 篇、捷運類 178 篇、其他類 173 篇，相較於陳亭愷(2015)、蔡易辰(2016)兩者研究，本研究之文本內容經過人工快速檢視，可節省大量時間，但資料內容相對較雜亂，此現象較符合實務情況，資料欄位包含：日期時間、標題、內文、情感值，前三個欄位為爬蟲抓取，情感值為計算完後自行填入，如表 4.1 所示。

表 4.1 資料型態

文章日期	標題	內容	情感值
2017/1/1 11:33	北捷跨年運量破264萬人次！比去年	▲北捷運站內湧入大量人潮。	0.015981
2017/1/3 11:32	首都客運開漲價第一槍　李博文：	網搜小組／綜合報導現代人傾向	-0.02803
2017/1/4 21:33	[閒聊] 在大眾運輸上分享影片與音	（手機發文，排版請見諒）　剛	-0.42717
2017/1/5 11:15	[創作] 占卜筆記：計程車	<img src='	-0.24134
2017/1/5 19:00	[經驗] 坐捷運發生的事	「2017南高屏澎交通論壇」今5	0.156776
2017/1/6 17:41	大叔捷運車廂狂挖鼻孔　抓桿變成	想問一下 ， 最近有打算要辦 台	0.013269
2017/1/6 22:02	[請益] 高鐵站發車的南投客運39號	●國民黨黨魁選戰開打 吳敦義	0.073075
2017/1/8 8:16	女高中生等公車 掏錢準備上車就被	幫你分析一下 如果你有開車騎	-0.13439
2017/1/10 0:00	Google 地圖開放台灣即時等公車系	（中央社記者沈如峰宜蘭縣10日	0.09543
2017/1/10 5:50	桃園心語－桃園，迎接捷運新時代	▲一例一休上路，勞工的權益受	-0.12148
2017/1/10 5:50	大溪免費公車時刻變 引民怨	為提升台灣使用者的行程規劃與	0.058081
2017/1/10 8:48	[疑問] 早上通勤火車擁擠度	哈囉大家好 最近在整理記帳 順	0.060035
2017/1/10 13:53	Google MAP即時大眾運輸資訊在台	自從Google在2005年開放線上地	0.042272
2017/1/10 21:58	[時刻] 170126 統聯 1626	大家好 新	-0.05383
2017/1/10 22:33	[請益] OFFER選擇 (漢微科/奇景/高	中國時報【陳芃／台北報導】大	-0.24479
2017/1/10 23:07	Re: [疑問] 早上通勤火車擁擠度	這是一棟老式建築的公寓，五層	-0.66392
2017/1/11 0:00	BUS+：最好看的等公車App，不被	勞基法修正案日前正式上路，原	0.033109
2017/1/11 19:50	[問題] 在交通不便地區工作	自從火車站搬去新站後 公車數量	-0.02042
2017/1/11 23:47	[閒聊] 公車到底要等多久	大家好 小弟剛到桃園工作一陣	0.035387
2017/1/12 0:00	北北桃等到了 機捷年後通車	https://i.imgur.com/OHxYp7b.jpg	-0.06806
2017/1/12 5:23	Re: [問題] 在交通不便地區工作	<a href="http://news.ltn.com.tw/ne	0.062652
2017/1/12 11:06	上班車程一小時...你們會想搬出去	▲桃園機場捷運建置DVR系統，	0.045296
2017/1/12 11:44	台中市公車是怎麼了？	我是要去家樂福，三個人在雙	-0.31781
2017/1/12 20:45	[閒聊] 搭捷運遇到很吵的人會制止	如題，近期想申請去錐麓古道，	0.021997
2017/1/13 5:50	本報直擊 台西客運違規跨縣載客	各位大大 大家好 本魯每周都要	-0.5007
2017/1/13 8:41	台北市規畫幹線路廊　公車轉乘公	搭乘路線：【665】榮總－信義	0.024648
2017/1/13 12:00	客運罔顧安全 用報廢車載學生!	※ 引述《jjw55	-0.05042
2017/1/13 14:49	搭計程車就嬌貴嗎?	草根影響力新視野圖文/陝紅宇	-0.25904
2017/1/14 13:08	[問題] 高鐵新竹站北上月台到六家	工商時報【方明／台北報導】赴	0.075947
2017/1/16 22:30	[紀實] 161227 大有 大園-1961直達	房屋地點：　新竹市關區市區 房	0.035287
2017/1/17 2:24	[討論] 在家附近跑業務，但要下班	不知道大家有沒有遇到這樣的狀	-0.23431
2017/1/17 13:10	北捷電扶梯左側「沒人站」？　網	姓 名 ：Nelson　　性 別：m	0.008018
2017/1/17 14:08	看演唱會會搭高鐵嗎?(跨縣市)	網路圖文部落格版： <a href="ht	-1.09696
2017/1/17 17:16	Re: [心得] uber搭乘	中國時報【張家豪／台北報導】	-0.06403
2017/1/18 15:33	高鐵周一上午北上列車自由座仍多	基隆市暖暖區暖江橋頭站的公車	0.034748

4.3 結巴(jieba)斷詞

本研究使用 Python3.6 版，安裝結巴斷詞套件，如圖 4.6 所示。結巴斷詞擁有不錯的斷詞正確率，且它方便擴充自訂辭典的設計以及簡單的操作方式讓使用者可以快速上手，為目前最廣泛使用之中文斷詞套件。

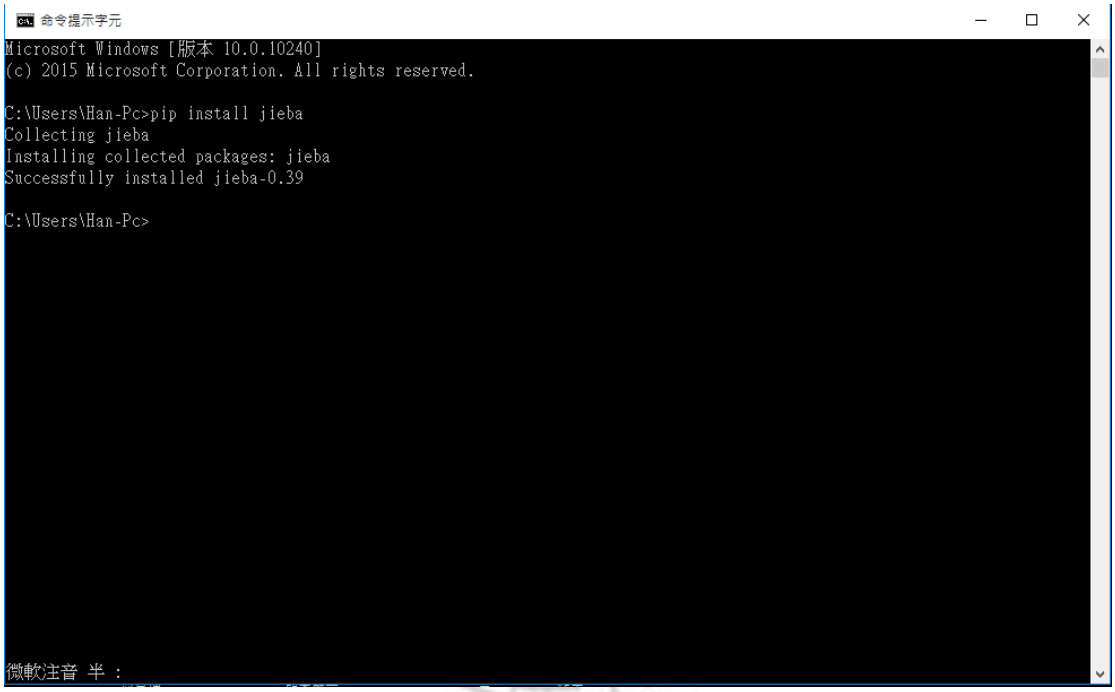

圖 4.6 安裝結巴套件

安裝完斷詞套件後，將文本輸入結巴斷詞，執行結果如圖 4.7 所示，將所有輸入文本斷詞完後，圖 4.8 為機器自動訓練提出特徵設計矩陣，總共斷出 72363 個詞彙，本模型訓練將資料輸入隨機抽取 90%為訓練資料，另 10%為測試文本，訓練及測試為 971/108，25 筆新資料做驗證。

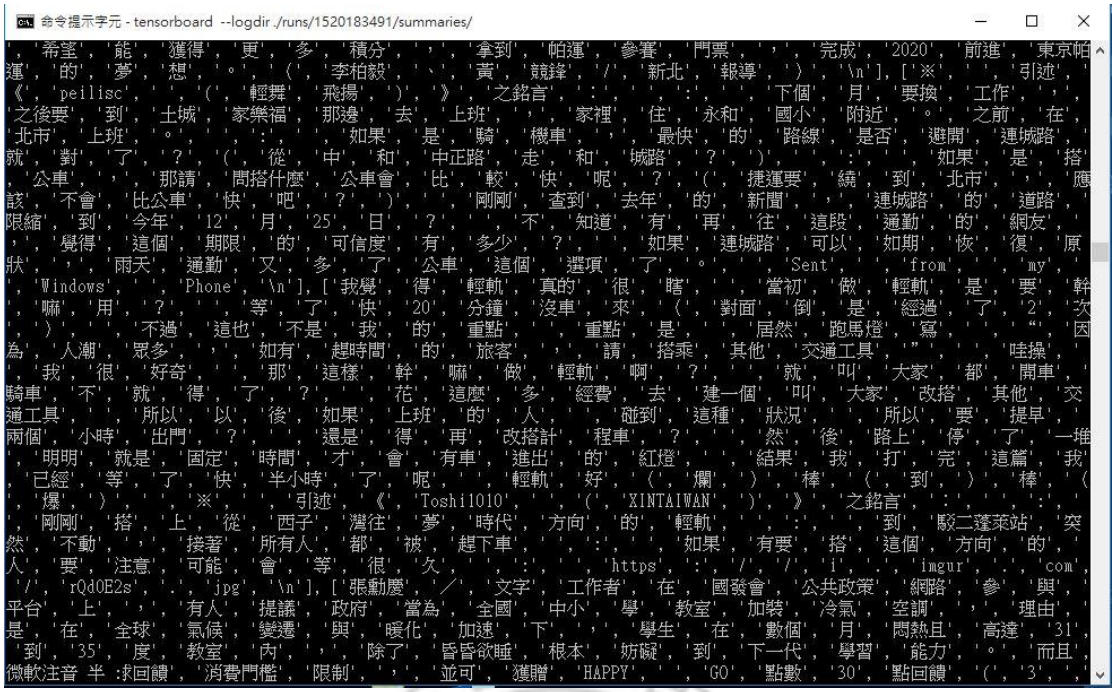

圖 4.7 結巴斷詞執行結果

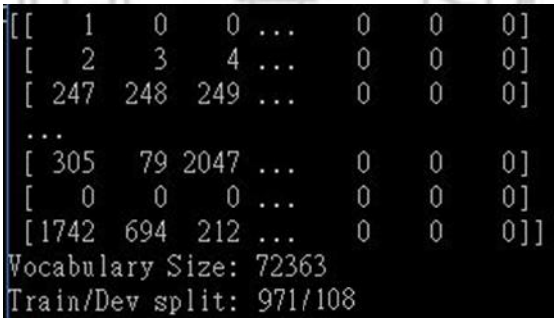

圖 4.8 訓練矩陣結果

4.4 情感值計算

台灣大學整理出的台大情感極性辭典(NTUSD)，為目前中文情感值計算較多人使用的辭典，向自然語言處理實驗室(Natural Language Processing, NLPLab)提出申請(供學術用途)，收到兩個辭典包含：正面 2810 及負面 8274 個詞彙，透過程式比對每篇文本辭典內詞彙，如表 4.2 所示。經 TF-IDF 提取情感特徵值，

加總成情感值，並透過標準差設計情感值區間，建構多元情感識別供後續卷積神經網路建模使用。

表 4.2 部分情緒詞彙

	正面情緒詞彙	次數	TF-IDF
2	一帆風順	0	0
3	一帆風順的	0	0
4	一流	3	9.04E-05
5	一致	16	0.000482276
6	一致的	1	3.01E-05
7	了不起	5	0.000150711
8	了不起的	0	0
9	瞭解	16	0.000482276
10	人性	9	0.00027128
11	人性的	1	3.01E-05
12	人格高尚	0	0
13	人格高尚的	0	0
14	人情	6	0.000180854
15	人情味	6	0.000180854
16	入神	0	0
17	入神的	0	0
18	入迷	0	0
19	入迷的	0	0
20	上好	8	0.000241138
21	上好的	0	0
22	上等	9	0.00027128
23	上等的	0	0
24	口頭通過	0	0
25	大方	26	0.000783699
26	大方的	2	6.03E-05
27	大無畏	0	0
28	大無畏的	0	0
29	大量的	16	0.000482276
30	大膽	4	0.000120569
31	大膽的	2	6.03E-05
32	小天使	3	9.04E-05
33	小心	84	0.002531951
34	小心的	1	3.01E-05
35	小心謹慎	1	3.01E-05
36	小心謹慎的	0	0

	負面情緒詞彙	次數	TF-IDF
2	一下子爆發	0	0
3	一下子爆發的一連串	0	0
4	一巴掌	2	-6.14E-05
5	一再	17	-0.000521921
6	一再叮囑	0	0
7	一拳	2	-6.14E-05
8	一般殺人罪	0	0
9	一陣狂風	0	0
10	一陣緊張	0	0
11	一掌	0	0
12	一團糟	0	0
13	一摑	0	0
14	一點點	32	-0.000982439
15	一蹶不振	0	0
16	人事不省	0	0
17	人為	17	-0.000521921
18	人為的	0	0
19	入迷	0	0
20	入迷的	0	0
21	入迷的人	0	0
22	刀刃	0	0
23	刁難	1	-3.07E-05
24	力盡	1	-3.07E-05
25	匕首	0	0
26	下地獄	1	-3.07E-05
27	下垂	2	-6.14E-05
28	下垂度	0	0
29	下流	4	-0.000122805
30	下流的	0	0
31	下降	24	-0.000736829
32	下陷	3	-9.21E-05
33	下等	0	0
34	下等的	0	0
35	下跌	8	-0.00024561
36	下獄	1	-3.07E-05

計算出文本情感值後，將正面區間門檻值之條件分別為：正面情感之最大值(P_{max})、平均值(P_{avg})、最小值(P_{min})；負面區間門檻值之條件分別為：負面情感之最大值(N_{max})、平均值(N_{avg})、最小值(N_{min})，結果如表 4.3 所示，負面文本數多於正面文本數，且負面情感最小值相較於正面情感最大值低出許多，符合運輸負效用及網路輿情貶多於褒之特性。

表 4.3 日常通勤情感值區間門檻

正面情感門檻值	負面情感門檻值
P_{max}：1.226980018	N_{max}：-0.0000307
P_{avg}：0.072500841	N_{avg}：-0.2395899
P_{min}：0.0000301	N_{min}：-3.221261708
正面文本總數:427	負面文本總數:677

以往研究多係利用二元決策分類將情感值界定為正面與負面或採用三元決策分析，將情感值區分成正面、中立與負面。本研究為了增加情感分類細緻度，透過標準差將情感值設計為五個區間，用於設計情感區間相關數值：整體情感平均值(E_{avg})、標準差(S.D)、整體情感平均值加 1 倍標準差(E_{avg}+1S.D)、整體情感平均值減1倍標準差(E_{avg}-1S.D)、整體情感平均值加0.5倍標準差(E_{avg}+0.5S.D)、整體情感平均值加 0.5 倍標準差(E_{avg}-0.5S.D)，如表 4.4 所示。

表 4.4 設計情感值區間相關參數

整體情感平均值(E_{avg})：-0.118729542
標準差(S.D)：0.312970524
(E_{avg})+(S.D)：0.194240982
(E_{avg})−(S.D)：-0.431700066
(E_{avg})+0.5(S.D)：0.03775572
(E_{avg})−0.5(S.D)：-0.275214804

情感辨識設計為：非常不滿意、不滿意、普通滿意、很滿意、非常滿意，針對各區間算法如表 4.5 所示，各情感區間門檻值如圖 4.9 所示，各區間次數統計則如圖 4.10 所示。

表 4.5 情感區間計算

非常滿意	$(E_{avg}) + (S.D) \sim P_{max}$
很滿意	$(E_{avg}) + 0.5(S.D) \sim (E_{avg}) + (S.D)$
普通滿意	$(E_{avg}) - 0.5(S.D) \sim (E_{avg}) + 0.5(S.D)$
不滿意	$(E_{avg}) - 0.5(S.D) \sim (E_{avg}) - (S.D)$
非常不滿意	$(E_{avg}) - (S.D) \sim N_{min}$

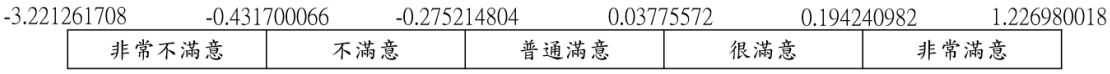

圖 4.9 情感區間尺度

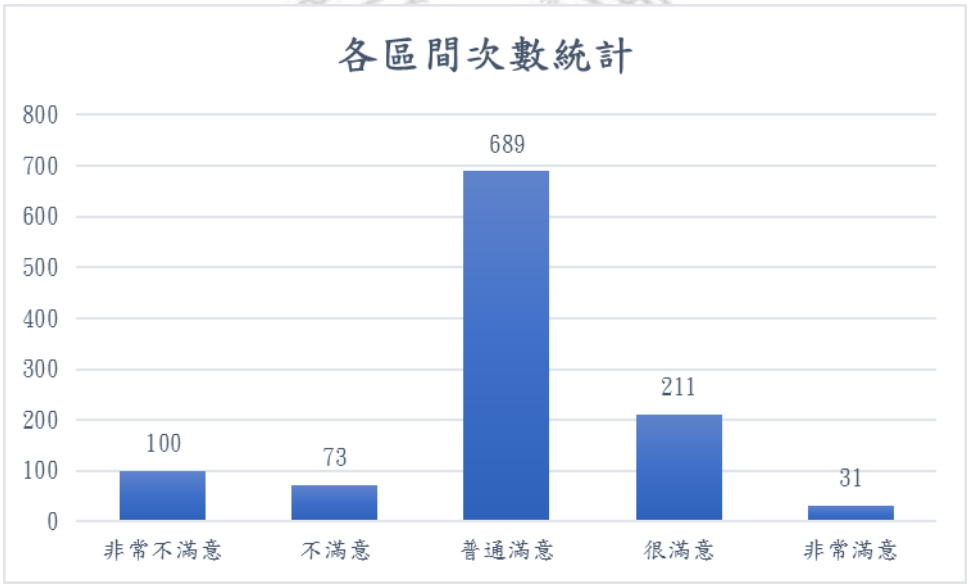

圖 4.10 情感值區間長條圖

以下圖表分別為各類運具每月情感趨勢圖及情感值區間統計。橫軸為月份，縱軸為正負文本當月累積數量。

捷運類情趨勢圖如圖 4.11 所示，滿意度區間大部分落於普通滿意，有一個月出現很滿意；情感最大值為 0.68；情感最小值為-0.79，如表 4.6 所示。

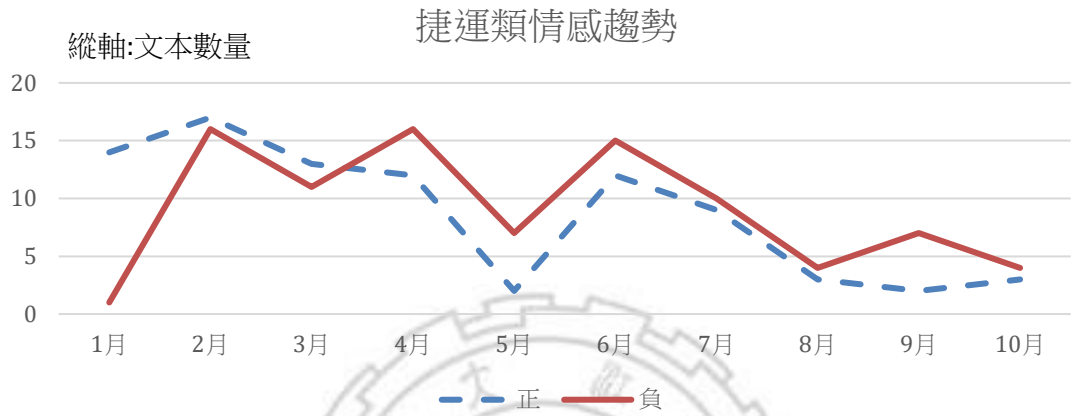

圖 4.11 捷運類情感趨勢圖

表 4.6 捷運類情感值

文本數量	1月	2月	3月	4月	5月	6月	7月	8月	9月	10月
正	14	17	13	12	2	12	9	3	2	3
負	1	16	11	16	7	15	10	4	7	4
情感最大值	0.213264	0.428356	0.677724	0.665335	0.068173	0.533944	0.397632	0.080085	0.278851	0.17648
情感最小值	-0.06886	-0.78724	-0.24633	-0.29325	-0.4378	-0.35438	-0.34666	-0.33546	-0.5258	-0.20608
情感值平均	0.0728	-0.05705	0.024637	-0.0404	-0.16237	-0.02294	0.001237	-0.05232	-0.11863	-0.01614
滿意度區間	很滿意	普通	普通	普通	普通	普通	普通	普通	普通	普通

火車類情趨勢圖如圖 4.12 所示，滿意度區間均落於普通滿意；情感最大值為 1.23；情感最小值為-2.51，如表 4.7 所示。

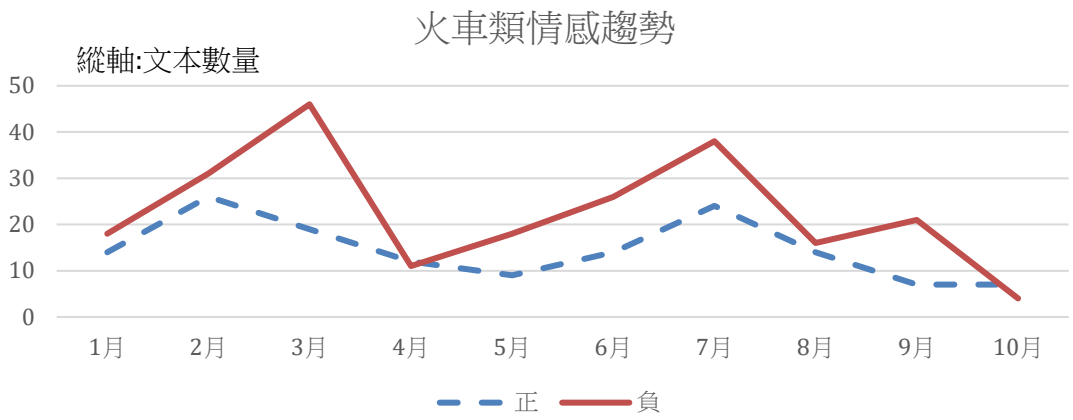

圖 4.12 火車類情感趨勢圖

表 4.7 火車類情感值

文本數量	1月	2月	3月	4月	5月	6月	7月	8月	9月	10月
正	14	26	19	12	9	14	24	14	7	7
負	18	31	46	11	18	26	38	16	21	4
情感最大值	0.134645	0.273849	0.327888	0.251228	1.22698	0.189905	0.213346	0.250154	0.194363	0.132418
情感最小值	-0.82879	-1.80735	-2.50738	-0.47525	-1.58626	-2.23332	-1.74889	-1.18352	-1.01273	-0.66165
情感值平均	-0.09758	-0.12654	-0.15518	-0.03584	-0.09988	-0.17066	-0.12818	-0.07205	-0.17875	-0.04537
滿意度區間	普通	普通	普通	普通	普通	普通	普通	普通	普通	普通

公車客運類情趨勢圖如圖 4.13 所示，滿意度區間均落於普通滿意；情感最大值為 0.27；情感最小值為-1.70，如表 4.8 所示。

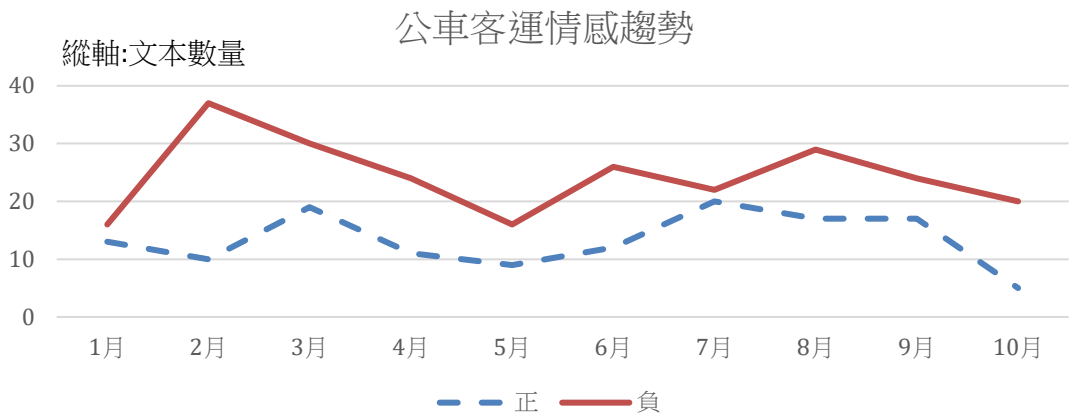

圖 4.13 公車客運類情感趨勢圖

表 4.8 公車客運類情感值

文本數量	1月	2月	3月	4月	5月	6月	7月	8月	9月	10月
正	13	10	19	11	9	12	20	17	17	5
負	16	37	30	24	16	26	22	29	24	20
情感最大值	0.116391	0.097808	0.232869	0.266208	0.096226	0.109199	0.144467	0.207447	0.219322	0.170679
情感最小值	-1.29872	-1.69171	-1.54204	-0.46079	-0.5741	-1.09974	-0.56517	-0.98691	-0.46762	-1.35331
情感值平均	-0.14264	-0.18293	-0.14895	-0.08084	-0.11485	-0.21163	-0.08136	-0.13006	-0.03646	-0.16476
滿意度區間	普通	普通	普通	普通	普通	普通	普通	普通	普通	普通

其他類情趨勢圖如圖 4.14 所示，滿意度區間大部分落於普通滿意，有一個月出現不滿意；情感最大值為 0.26；情感最小值為-3.22，如表 4.9 所示。

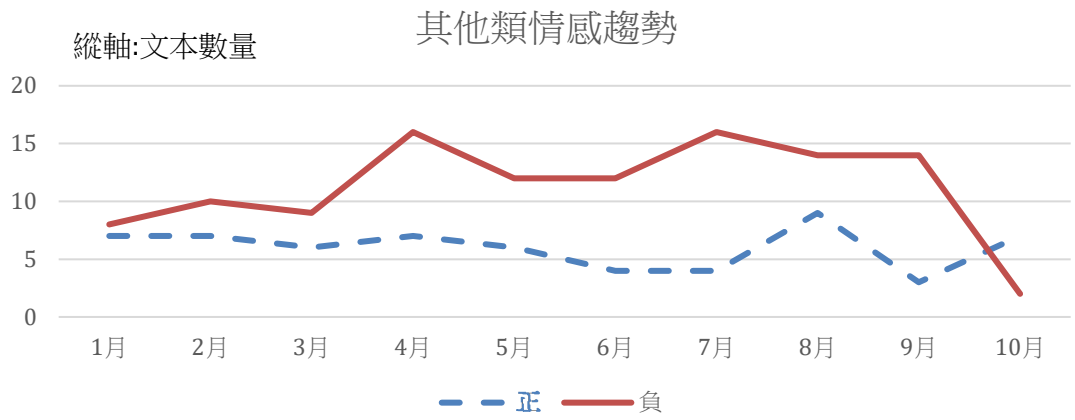

圖 4.14 其他類情感值趨勢

表 4.9 其他類情感值

文本數量	1月	2月	3月	4月	5月	6月	7月	8月	9月	10月
正	7	7	6	7	6	4	4	9	3	7
負	8	10	9	16	12	12	16	14	14	2
情感最大值	0.213626	0.121555	0.121649	0.066433	0.259124	0.064311	0.225218	0.113559	0.026043	0.088451
情感最小值	-0.69986	-0.81241	-1.37072	-2.187	-0.68328	-3.22126	-0.9825	-0.83669	-1.74432	-0.50894
情感值平均	-0.10515	-0.13432	-0.24314	-0.23627	-0.10335	-0.41361	-0.14159	-0.07084	-0.27745	-0.03351
滿意度區間	普通	普通	普通	普通	普通	普通	普通	普通	不滿意	普通

綜合上述圖表可顯示，各運具滿意度區間多落於普通滿意；大部分每月負文本累積數數多於正；情感值擺盪幅度從大到小之順序為：其他類(計程車、交通車、校車)、火車類(台鐵、高鐵)、公車客運類、捷運類。

4.5 卷積神經網路

本研究使用 Python3.6，為執行卷積神經網路，首先需要安裝三個擴充程式庫，jieba(斷詞)、Numpy(矩陣運算)、Tensorflow(卷積神經網路)，本研究流程及程式碼參考 Denny Britz (Google Brain team)分享於 Github 公開資源，經過修改調整為符合本研究之程式碼。

本研究計算 CNN 模型流程圖如圖 4.15 所示，透過 Tensorboard 視覺化呈現 Tensorflow 運算流程，包含輸入層(Input layer)、卷積層(Convolution layer)、

池化層(Pooling layer)、激活函數(Activation function)、全連接層(Full connected layer)、丟棄層(Dropout layer)，說明如下:

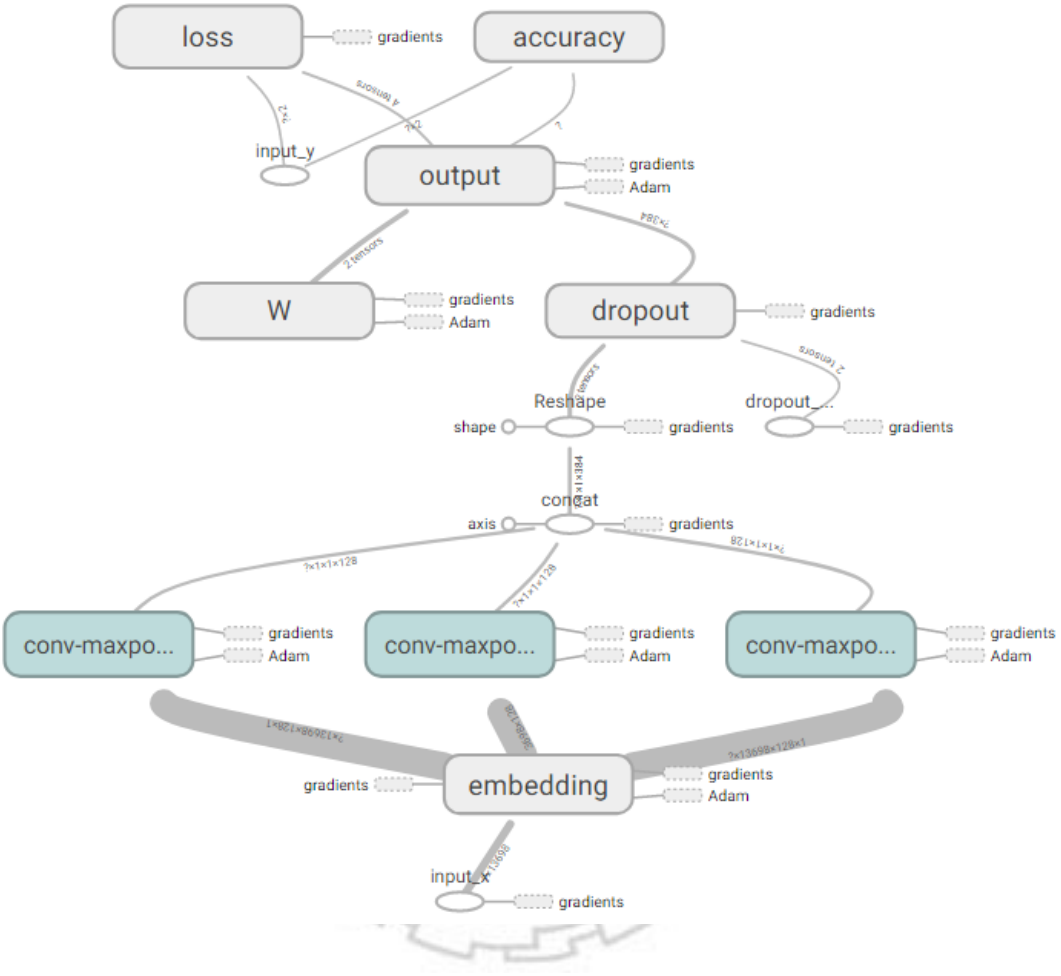

圖 4.15 Tensorflow 運算流程

4.5.1 輸入層

資料前處理將爬蟲抓回來的 1104 篇文本，90%當作訓練資料，10%作為交叉測試資料，以人工標籤方式標記成五個不同區間，同時輸入五個 txt 檔案，使資料轉換成 CNN 讀取格式，透過 embedding layer 將資料轉變成n × k矩陣(n 為斷詞後句子最大長度，k 為詞向量維度)，embedding layer 執行 jieba 斷詞後機器自動學習尋找規律並設計特徵矩陣，如圖 4.16 所示。

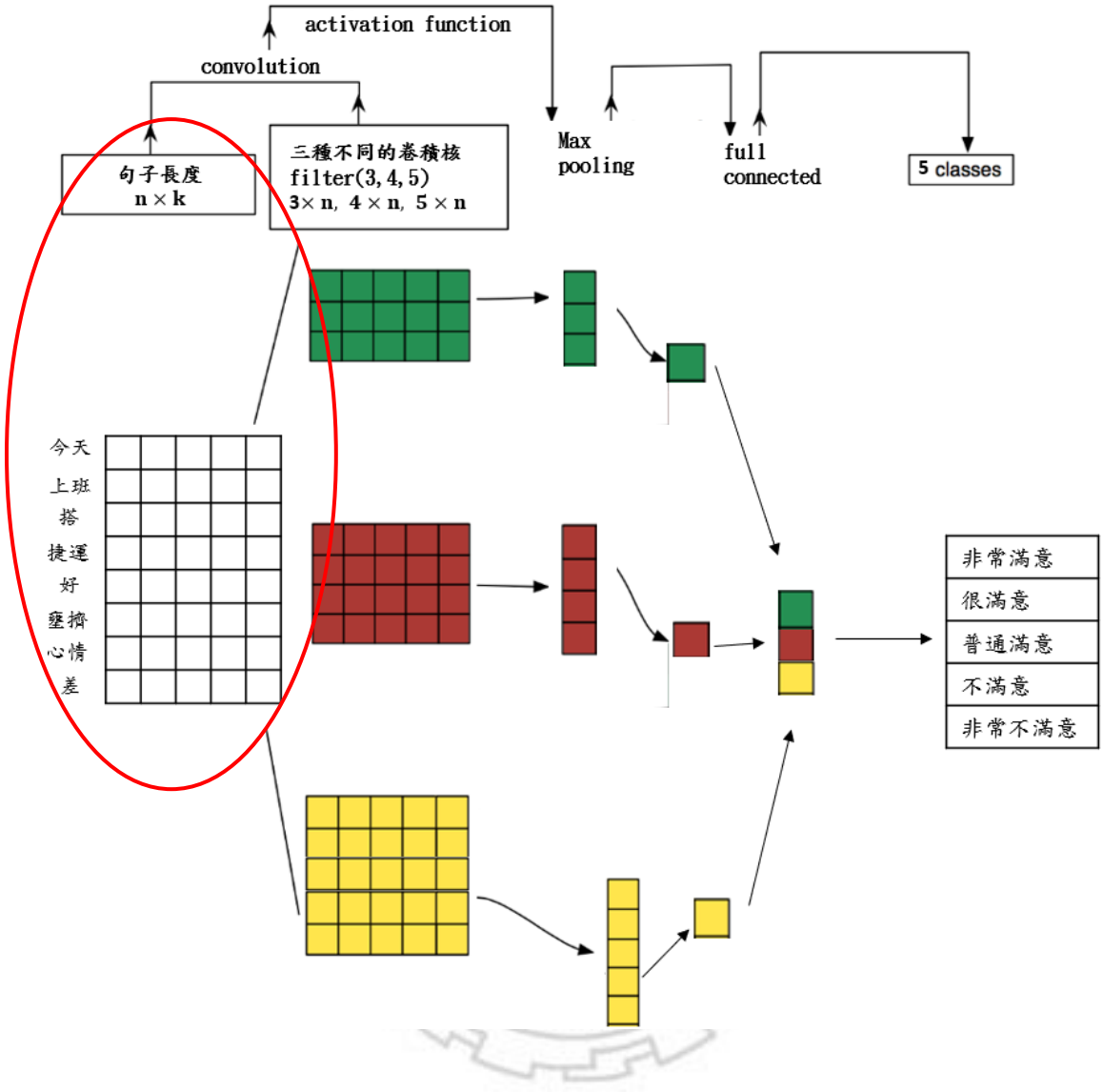

圖 4.16 卷積神經網路架構

4.5.2 卷積層

卷積層的運算是透過卷積核(filter)操作，本研究用3 × n、4 × n、5 × n三種不同卷積核做卷積，由左至右每次滑動 1 單位，運算示意如圖 4.17 所示。

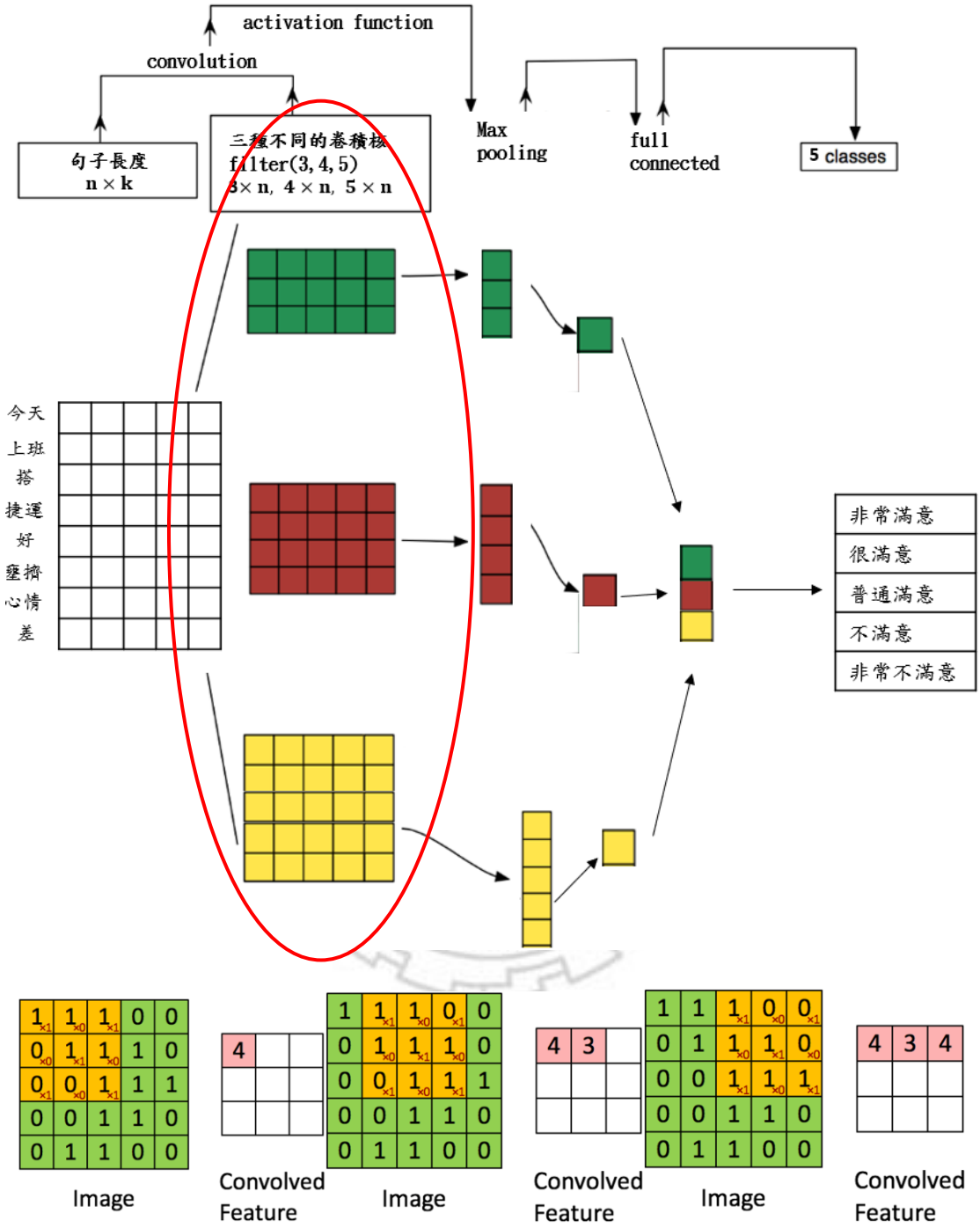

圖 4.17 卷積運算示意圖

4.5.3 激活函數

如果不使用激活函數，神經網路只能構建線性分類器且有梯度爆炸問題，無論使用多少層神經網路，輸出都是輸入的線性組合，與只有一個隱含層效果相當。

本研究使用線性整流函數 ReLU，圖 4.18 為人工神經網路的激活函數，且較其他函數有更好的效果。

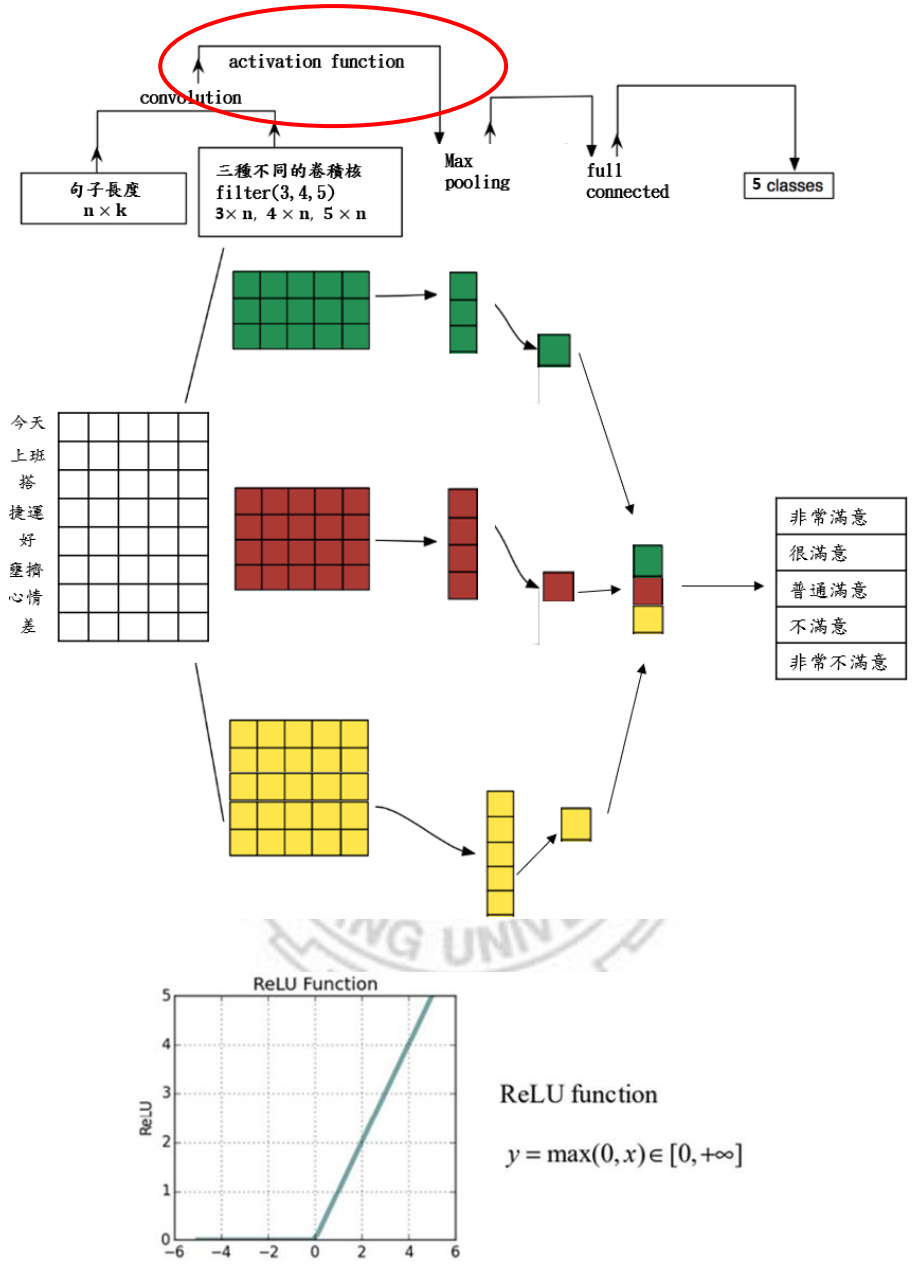

圖 4.18 線性整流函數

4.5.4 池化層

根據近年來學者研究以及實務上的驗證，以池化層作為特徵提取有三種不同之方法，分別為最小、平均、最大池化層。因最大池化層(Max pooling)特徵提取

之準確度最高，提取效率最好，本研究以最大池化層作為特徵提取之方法，藉以降低輸入矩陣之維度，並從中得到最能代表該矩陣的特徵(即為矩陣中的最大值)。於本研究中，輸入向量之維度從 4x4 降低為 2x2，如圖 4.19 所示。

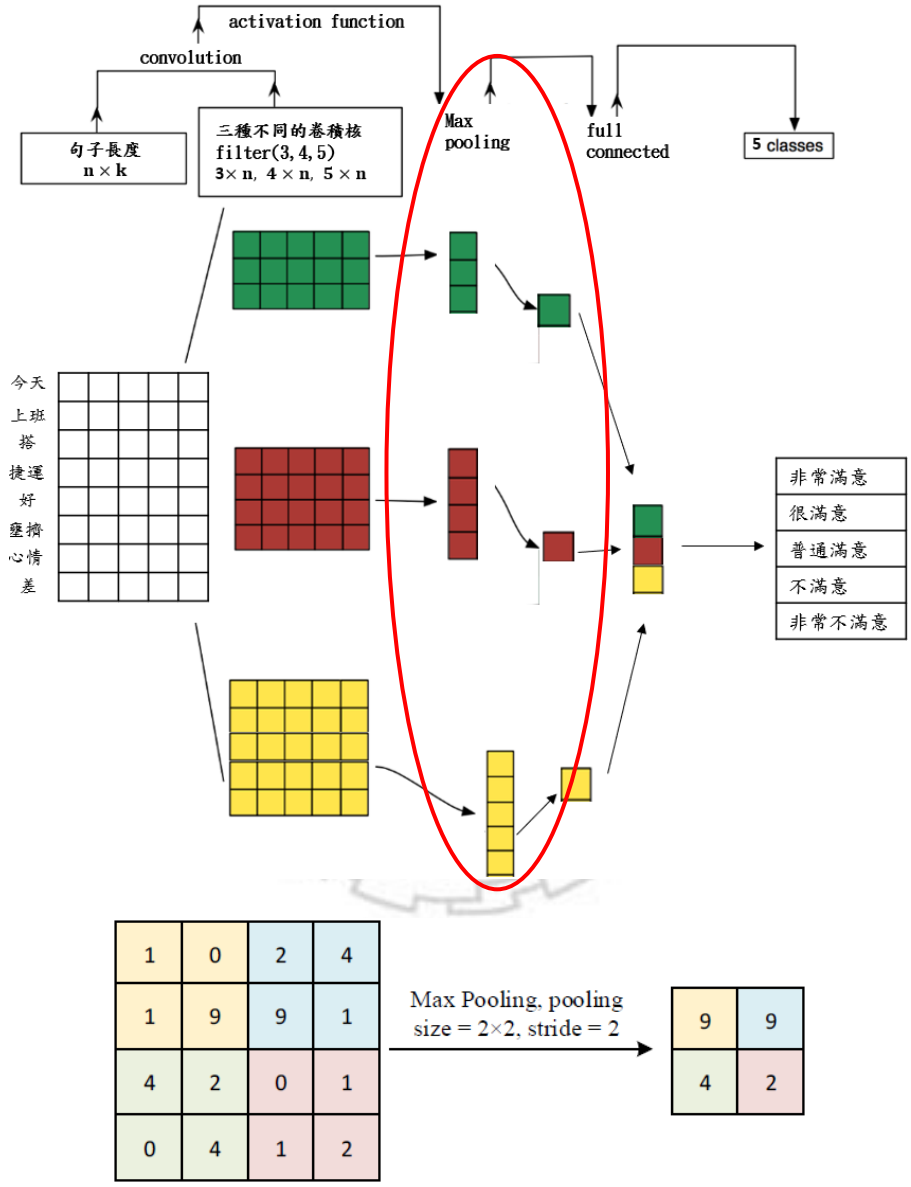

圖 4.19 Max pooling 操作

4.5.5 全連接層

經由上述卷積層、池化層、激活函數不斷重複運算，使輸入資料不斷降維簡化，最後得到 N 個降維後之輸入，利用倒傳遞神經網路(BPNN)演算法進行全連

接層，計算誤差後更新權重，則可得到最後解，圖 4.20 為全連接層示意圖。

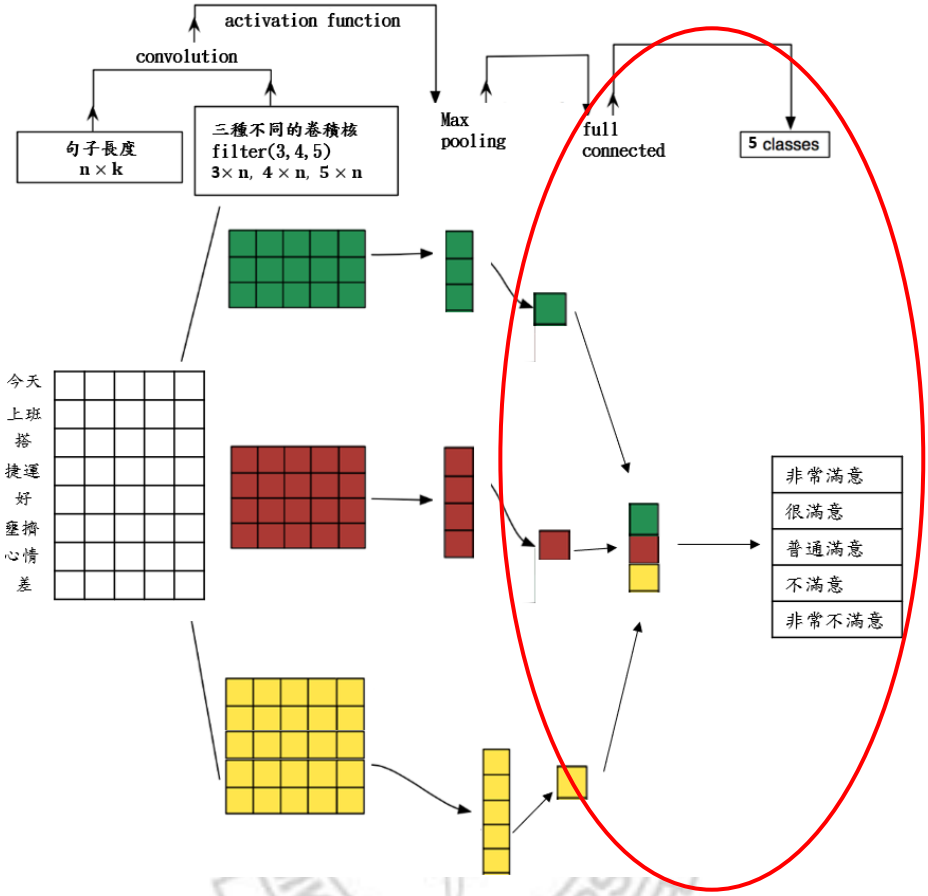

圖 4.20 全連接層示意圖

4.5.6 丟棄層

在小規模資料集上訓練卷積神經網路容易發生過度擬合(overfitting)的問題，為了防止出現過度擬合，在全連接層使用丟棄(dropout)機制，dropout 透過隨機關閉神經元方式，防止特徵間互相適應，減少模型過度擬合，如圖 4.21 所示。

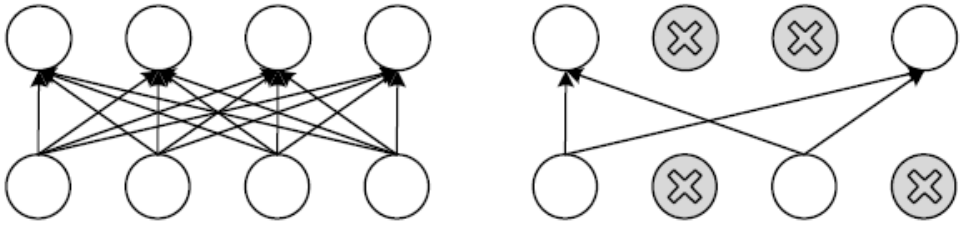

圖 4.21 Dropout 機制示意圖

圖 4.21 為標準兩層神經網路，右圖是使用 dropout 機制隨機關閉神經元，灰色打叉的神經元處於關閉狀態，在訓練網路時，節點以一定機率隨機關閉，因此不能保證每兩個節點每次同時出現，可以讓網路參數的更新不再依賴於固定關聯式結構的節點，防止節點間的共同作用，阻止某些特徵僅在其他特定特徵出現下才發揮顯現的情況，在自然語言處理中(NLP)，dropout 機制只在全連接層使用，不在卷積層及池化層中使用。

4.5.7 模式建構

根據前述分別說明卷積神經網路各層之功能，本節展示以 Python 的擴充程式 tensorflow 完整實作出卷積神經網路，以藉由神經網路及情感值達到將文本有效地以滿意度分類之目的。

首先，有關程式中的參數設定如圖 4.22，說明如下:

- sequence_length-句子長度，填充使所有句子擁有相同長度
- num_classes-輸出分類數目，本研究為五個分類。
- vocab_size-詞彙量的大小。此參數為確定詞向量嵌入層的大小，最終的總詞向量維度是[vocabulary_size, embedding_size]
- embedding_size-每個單詞的詞向量的維度 128
- filter_sizes-此參數為每次卷積覆蓋幾個單詞。filter_sizes=[3, 4, 5]，

表示卷積核一共有三種類型，分別是每次卷積覆蓋 3 個單詞，每次卷積覆蓋 4 個單詞和每次卷積覆蓋 5 個單詞。

- num_filters-每個卷積核數量，每個卷積核有 128 個總共 384 個。

```
import tensorflow as tf
import numpy as np

class TextCNN(object):
    """
    A CNN for text classification.
    Uses an embedding layer, followed by a convolutional, max-pooling and softmax layer.
    """
    def __init__(
      self, sequence_length, num_classes, vocab_size,
      embedding_size, filter_sizes, num_filters):
        # Implementation...
```

圖 4.22 CNN 參數設計

再者，藉由 Python 環境下所實作出的卷積神經網路過程如圖 4.23 所示，首先是執行 jieba 斷詞。

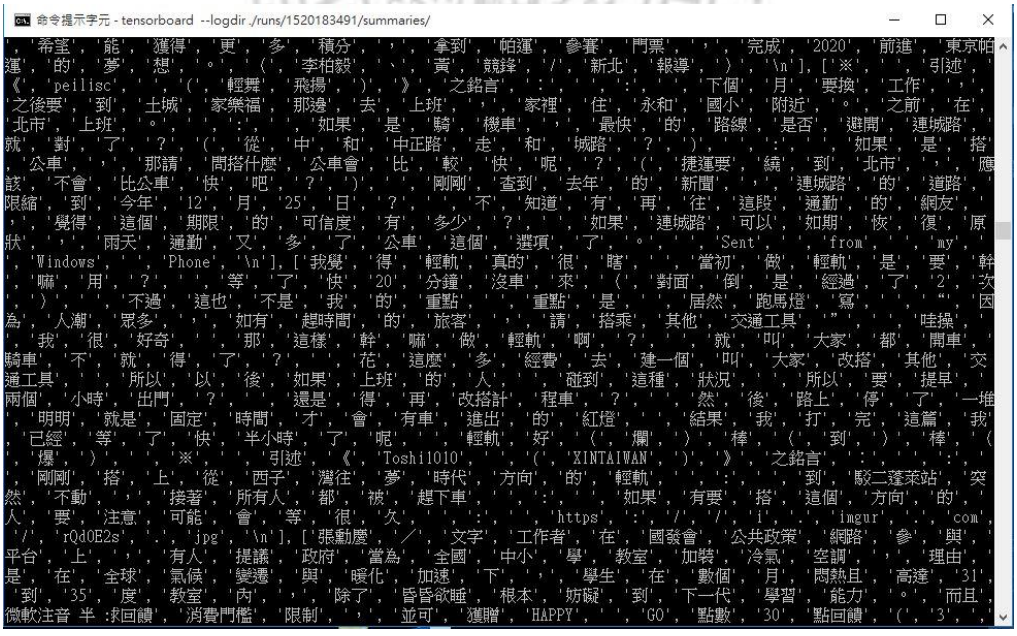

圖 4.23 CNN 執行過程(jieba 斷詞)

接著開始運算卷積神經網路，圖 4.24 至圖 4.28 為每一個 Step 執行訓練的準

確率。

```
2018-03-05T01:12:03.005096: step 1, loss 11.359, acc 0.390625
2018-03-05T01:12:31.403766: step 2, loss 7.99403, acc 0.578125
2018-03-05T01:12:59.464080: step 3, loss 4.05494, acc 0.65625
2018-03-05T01:13:27.047271: step 4, loss 3.18229, acc 0.734375
2018-03-05T01:13:54.992444: step 5, loss 2.41713, acc 0.78125
2018-03-05T01:14:22.480103: step 6, loss 3.31946, acc 0.78125
2018-03-05T01:14:50.377894: step 7, loss 4.82583, acc 0.71875
2018-03-05T01:15:18.971112: step 8, loss 3.63209, acc 0.765625
2018-03-05T01:15:46.437533: step 9, loss 3.43536, acc 0.8125
2018-03-05T01:16:14.793568: step 10, loss 3.31607, acc 0.84375
2018-03-05T01:16:42.508971: step 11, loss 2.73807, acc 0.84375
2018-03-05T01:17:10.303501: step 12, loss 4.97515, acc 0.796875
2018-03-05T01:17:37.981073: step 13, loss 2.61654, acc 0.859375
2018-03-05T01:17:43.792759: step 14, loss 5.43291, acc 0.769231
2018-03-05T01:18:11.739324: step 15, loss 6.75917, acc 0.703125
2018-03-05T01:18:42.484625: step 16, loss 5.96919, acc 0.78125
2018-03-05T01:19:12.148601: step 17, loss 4.23653, acc 0.84375
2018-03-05T01:19:41.784121: step 18, loss 5.82369, acc 0.78125
2018-03-05T01:20:11.581399: step 19, loss 4.14327, acc 0.859375
2018-03-05T01:20:40.738770: step 20, loss 6.46819, acc 0.8125
2018-03-05T01:21:10.175698: step 21, loss 4.97206, acc 0.84375
2018-03-05T01:21:39.843262: step 22, loss 4.06475, acc 0.875
2018-03-05T01:22:09.147085: step 23, loss 8.89956, acc 0.71875
2018-03-05T01:22:38.944230: step 24, loss 8.93913, acc 0.734375
2018-03-05T01:23:08.411217: step 25, loss 5.64239, acc 0.84375
2018-03-05T01:23:38.386628: step 26, loss 7.88349, acc 0.78125
2018-03-05T01:24:08.336514: step 27, loss 3.25698, acc 0.921875
2018-03-05T01:24:14.325965: step 28, loss 9.67235, acc 0.769231
2018-03-05T01:24:44.307689: step 29, loss 2.52021, acc 0.9375
2018-03-05T01:25:14.093710: step 30, loss 7.0064, acc 0.796875
2018-03-05T01:25:42.925873: step 31, loss 8.89254, acc 0.78125
2018-03-05T01:26:12.645831: step 32, loss 6.94405, acc 0.828125
2018-03-05T01:26:42.402118: step 33, loss 14.0244, acc 0.640625
2018-03-05T01:27:11.844537: step 34, loss 7.81973, acc 0.8125
2018-03-05T01:27:41.791974: step 35, loss 8.29395, acc 0.8125
2018-03-05T01:28:11.770758: step 36, loss 8.28419, acc 0.828125
2018-03-05T01:28:41.085381: step 37, loss 8.50579, acc 0.828125
2018-03-05T01:29:10.220720: step 38, loss 8.92446, acc 0.828125
2018-03-05T01:29:39.095626: step 39, loss 13.0265, acc 0.75
2018-03-05T01:30:10.395532: step 40, loss 8.96461, acc 0.8125
2018-03-05T01:30:38.188076: step 41, loss 9.1619, acc 0.828125
2018-03-05T01:30:43.977928: step 42, loss 7.18487, acc 0.846154
2018-03-05T01:31:12.572464: step 43, loss 11.2057, acc 0.796875
2018-03-05T01:31:41.096391: step 44, loss 6.90141, acc 0.859375
2018-03-05T01:32:09.428293: step 45, loss 14.8066, acc 0.71875
2018-03-05T01:32:37.865136: step 46, loss 9.85147, acc 0.828125
2018-03-05T01:33:05.858943: step 47, loss 10.6212, acc 0.8125
2018-03-05T01:33:35.009834: step 48, loss 5.79316, acc 0.890625
2018-03-05T01:34:03.110812: step 49, loss 12.9528, acc 0.8125
2018-03-05T01:34:32.508933: step 50, loss 8.83646, acc 0.859375
```

圖 4.24 CNN 執行過程(訓練 1)

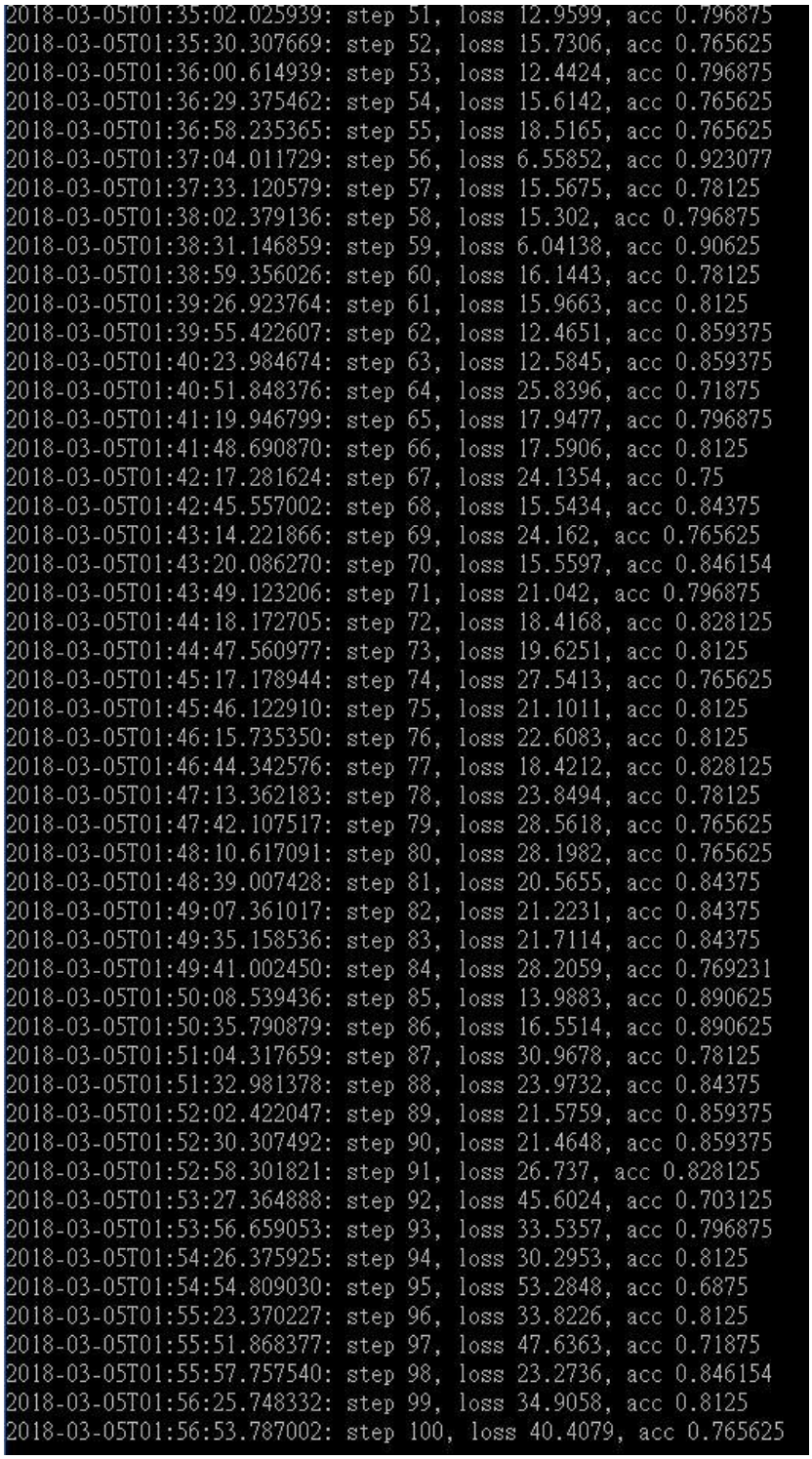

圖 4.25 CNN 執行過程(訓練 2)

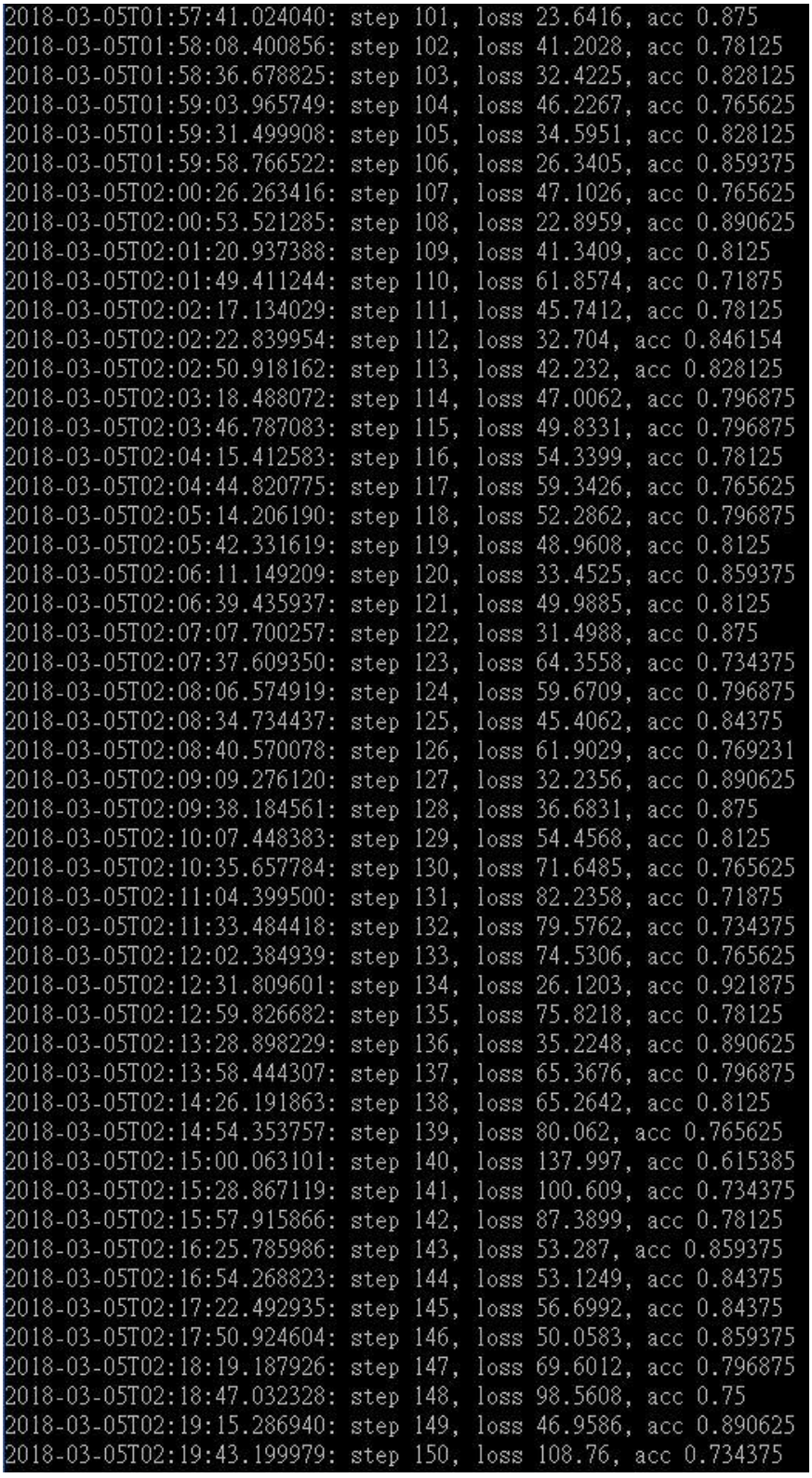

圖 4.26 CNN 執行過程(訓練 3)

```
2018-03-05T02:20:11.603975: step 151, loss 67.4521, acc 0.84375
2018-03-05T02:20:40.366153: step 152, loss 74.661, acc 0.8125
2018-03-05T02:21:09.165634: step 153, loss 111.332, acc 0.734375
2018-03-05T02:21:15.045681: step 154, loss 64.2777, acc 0.846154
2018-03-05T02:21:43.598502: step 155, loss 84.4397, acc 0.8125
2018-03-05T02:22:12.143517: step 156, loss 93.5, acc 0.796875
2018-03-05T02:22:40.603394: step 157, loss 62.6736, acc 0.859375
2018-03-05T02:23:09.233729: step 158, loss 68.683, acc 0.84375
2018-03-05T02:23:37.786211: step 159, loss 95.2481, acc 0.796875
2018-03-05T02:24:06.186380: step 160, loss 77.1916, acc 0.828125
2018-03-05T02:24:35.402016: step 161, loss 84.2573, acc 0.828125
2018-03-05T02:25:04.327866: step 162, loss 156.035, acc 0.6875
2018-03-05T02:25:32.932403: step 163, loss 129.984, acc 0.71875
2018-03-05T02:26:00.735833: step 164, loss 74.9827, acc 0.84375
2018-03-05T02:26:28.592181: step 165, loss 74.0086, acc 0.84375
2018-03-05T02:26:55.999748: step 166, loss 84.4667, acc 0.84375
2018-03-05T02:27:23.737884: step 167, loss 129.328, acc 0.765625
2018-03-05T02:27:29.477075: step 168, loss 49.1027, acc 0.923077
2018-03-05T02:27:56.884294: step 169, loss 146.286, acc 0.71875
2018-03-05T02:28:24.475960: step 170, loss 90.059, acc 0.828125
2018-03-05T02:28:51.910382: step 171, loss 95.6114, acc 0.8125
2018-03-05T02:29:19.444118: step 172, loss 173.496, acc 0.6875
2018-03-05T02:29:46.660367: step 173, loss 85.3663, acc 0.84375
2018-03-05T02:30:14.315757: step 174, loss 88.0353, acc 0.859375
2018-03-05T02:30:42.571624: step 175, loss 125.469, acc 0.78125
2018-03-05T02:31:11.262523: step 176, loss 69.507, acc 0.875
2018-03-05T02:31:38.744315: step 177, loss 95.0483, acc 0.828125
2018-03-05T02:32:06.211399: step 178, loss 105.413, acc 0.828125
2018-03-05T02:32:34.402221: step 179, loss 102.319, acc 0.828125
2018-03-05T02:33:03.859082: step 180, loss 87.6803, acc 0.859375
2018-03-05T02:33:33.050312: step 181, loss 137.284, acc 0.765625
2018-03-05T02:33:39.240191: step 182, loss 157.227, acc 0.692308
2018-03-05T02:34:07.160322: step 183, loss 123.832, acc 0.8125
2018-03-05T02:34:35.361823: step 184, loss 163.403, acc 0.75
2018-03-05T02:35:02.717062: step 185, loss 121.503, acc 0.8125
2018-03-05T02:35:30.159789: step 186, loss 194.097, acc 0.6875
2018-03-05T02:35:57.538275: step 187, loss 91.5916, acc 0.84375
2018-03-05T02:36:25.492026: step 188, loss 107.5, acc 0.84375
2018-03-05T02:36:52.635215: step 189, loss 162.942, acc 0.75
2018-03-05T02:37:20.652721: step 190, loss 112.515, acc 0.84375
2018-03-05T02:37:49.022385: step 191, loss 137.746, acc 0.796875
2018-03-05T02:38:16.924930: step 192, loss 102.403, acc 0.84375
2018-03-05T02:38:44.505323: step 193, loss 114.033, acc 0.828125
2018-03-05T02:39:12.266500: step 194, loss 139.048, acc 0.796875
2018-03-05T02:39:39.609527: step 195, loss 103.763, acc 0.859375
2018-03-05T02:39:45.295403: step 196, loss 48.6888, acc 0.923077
2018-03-05T02:40:13.508664: step 197, loss 105.522, acc 0.859375
2018-03-05T02:40:41.156677: step 198, loss 109.547, acc 0.859375
2018-03-05T02:41:08.724615: step 199, loss 186.57, acc 0.75
2018-03-05T02:41:36.271237: step 200, loss 172.691, acc 0.765625
```

圖 4.27 CNN 執行過程(訓練 4)

總共訓練 2800 次後停止，最終的模式準確度為 0.784946。

```
2018-03-05T20:59:02.350127: step 2750, loss 280326, acc 0.734375
2018-03-05T20:59:30.717854: step 2751, loss 177672, acc 0.828125
2018-03-05T20:59:57.597633: step 2752, loss 165031, acc 0.84375
2018-03-05T21:00:24.827035: step 2753, loss 203722, acc 0.796875
2018-03-05T21:00:51.668661: step 2754, loss 180933, acc 0.828125
2018-03-05T21:01:18.742927: step 2755, loss 236364, acc 0.78125
2018-03-05T21:01:45.758380: step 2756, loss 137976, acc 0.875
2018-03-05T21:02:12.881119: step 2757, loss 193456, acc 0.8125
2018-03-05T21:02:18.561061: step 2758, loss 326038, acc 0.692308
2018-03-05T21:02:46.027495: step 2759, loss 220981, acc 0.78125
2018-03-05T21:03:13.562020: step 2760, loss 177207, acc 0.828125
2018-03-05T21:03:40.490733: step 2761, loss 172670, acc 0.828125
2018-03-05T21:04:07.594254: step 2762, loss 308694, acc 0.71875
2018-03-05T21:04:34.767184: step 2763, loss 144948, acc 0.859375
2018-03-05T21:05:02.084096: step 2764, loss 302704, acc 0.71875
2018-03-05T21:05:29.174455: step 2765, loss 145533, acc 0.859375
2018-03-05T21:05:56.304705: step 2766, loss 249160, acc 0.75
2018-03-05T21:06:23.716545: step 2767, loss 175950, acc 0.828125
2018-03-05T21:06:50.945303: step 2768, loss 128252, acc 0.875
2018-03-05T21:07:17.859735: step 2769, loss 231724, acc 0.78125
2018-03-05T21:07:44.709972: step 2770, loss 210977, acc 0.796875
2018-03-05T21:08:11.732262: step 2771, loss 149648, acc 0.859375
2018-03-05T21:08:17.380697: step 2772, loss 173763, acc 0.846154
2018-03-05T21:08:44.518121: step 2773, loss 172645, acc 0.828125
2018-03-05T21:09:12.117403: step 2774, loss 176733, acc 0.828125
2018-03-05T21:09:39.494670: step 2775, loss 134683, acc 0.875
2018-03-05T21:10:07.075363: step 2776, loss 255993, acc 0.765625
2018-03-05T21:10:34.191449: step 2777, loss 126029, acc 0.875
2018-03-05T21:11:01.215266: step 2778, loss 287557, acc 0.71875
2018-03-05T21:11:28.722460: step 2779, loss 209651, acc 0.796875
2018-03-05T21:11:55.621846: step 2780, loss 168521, acc 0.84375
2018-03-05T21:12:22.375376: step 2781, loss 206100, acc 0.8125
2018-03-05T21:12:49.915221: step 2782, loss 137320, acc 0.875
2018-03-05T21:13:16.960776: step 2783, loss 267260, acc 0.75
2018-03-05T21:13:44.181362: step 2784, loss 290620, acc 0.734375
2018-03-05T21:14:11.127132: step 2785, loss 242939, acc 0.78125
2018-03-05T21:14:16.805063: step 2786, loss 161882, acc 0.846154
2018-03-05T21:14:44.944576: step 2787, loss 169470, acc 0.84375
2018-03-05T21:15:11.930951: step 2788, loss 169706, acc 0.84375
2018-03-05T21:15:39.130807: step 2789, loss 254165, acc 0.765625
2018-03-05T21:16:06.242851: step 2790, loss 240802, acc 0.765625
2018-03-05T21:16:33.309806: step 2791, loss 142929, acc 0.875
2018-03-05T21:17:00.515861: step 2792, loss 101955, acc 0.90625
2018-03-05T21:17:27.492768: step 2793, loss 141361, acc 0.875
2018-03-05T21:17:54.500224: step 2794, loss 258650, acc 0.75
2018-03-05T21:18:21.445225: step 2795, loss 243789, acc 0.78125
2018-03-05T21:18:48.159781: step 2796, loss 302852, acc 0.71875
2018-03-05T21:19:15.487102: step 2797, loss 223806, acc 0.796875
2018-03-05T21:19:42.530177: step 2798, loss 213067, acc 0.796875
2018-03-05T21:20:09.499778: step 2799, loss 240181, acc 0.78125
2018-03-05T21:20:15.212484: step 2800, loss 264275, acc 0.769231
```

圖 4.28 CNN 執行過程(訓練 5)

模式在每訓練 100 次時會評估模式準確度，圖 4.29 至圖 4.31 為部分評估圖片擷取。

圖 4.29 CNN 執行過程(評估 1)

圖 4.30 CNN 執行過程(評估 2)

圖 4.31 CNN 執行過程(評估 3)

圖 4.32 為 Tensorbroad 視覺化 CNN 模式訓練及測試震盪圖，橘線為訓練擺盪準確度，藍線為測試準確度，總共訓練 2800 次，最終模式準確度達 0.784946，總共耗費 26 小時。

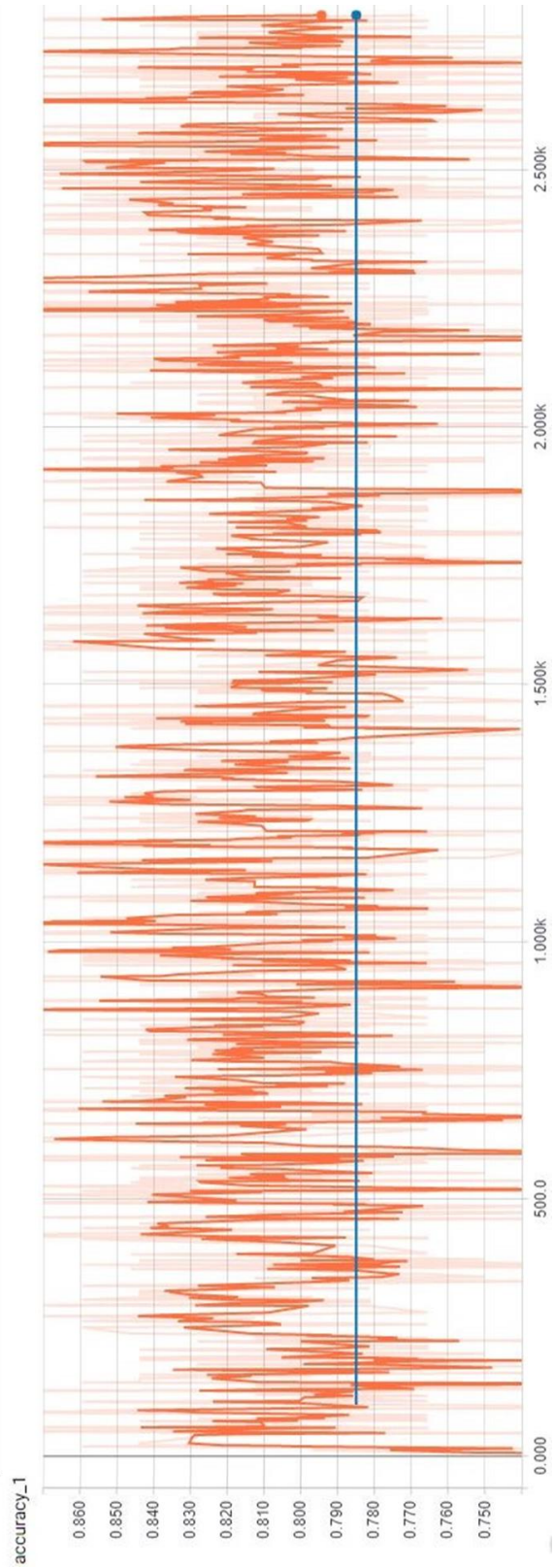

圖 4.32 CNN 模式訓練測試震盪圖

4.5.8 模式驗證

在產生卷積神經網路模型後，為了驗證模型預測之準確度，本研究以 25 筆新資料作為驗證，在人工智慧領域中，常以混淆矩陣(confusion matrix)對監督式分類演算法進行準確度評估，透過模型預測的數據與驗證的數據進行比對，使用準確率，覆蓋率和命中率等指標對模型的分類效果進行度量。

每個分類 5 筆總共 25 新資料(不含在訓練及測試內)，放入模型中進行分類預測結果如圖 4.33 至圖 4.37 所示。

```
Parameters:
ALLOW_SOFT_PLACEMENT=True
BATCH_SIZE=64
CHECKPOINT DIR /runs/1520995995/checkpoints/
EVAL_TRAIN=True
LOG_DEVICE_PLACEMENT=False
VERY_NEGATIVE_DATA_FILE=./data/test_text/veryneg.txt
NEGATIVE_DATA_FILE=./data/test_text/neg.txt
NEUTRAL_DATA_FILE=./data/test_text/neu.txt
POSITIVE_DATA_FILE=./data/test_text/pos.txt
VERY_POSITIVE_DATA_FILE=./data/test_text/verypos.txt

Building prefix dict from the default dictionary
Loading model from cache /tmp/jieba.cache
Loading model cost 0.614 seconds
Prefix dict has been built successfully

Input Text…
（大紀元記者徐乃義台灣桃園報導）連結國之門到首都的機場捷運通車，對沿線環...

Evaluating…

2018-03-20 06:25:42.187312 W tensorflow/core/platform/cpu feature_uard.cc:45] The TensorFlow library wasn't compiled to
use SSE4.1 instructions, but these are available on your machine and could speed up CPU computations
2018-03-20 06:25:42.187321 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to
 use SSE4.2 instructions, but these are available on your machine and could speed up CPU computations
2018-03-20 06:25:42.187329 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to
 use AVX instructions, but these are available on your machine and could speed up CPU computations
2018-03-20 06:25:42.187336 W tensorflow/core/platform/cpu feature uard.cc:45] The Tensor Flow library wasn't compiled to
 use AVX2 instructions, but these are available on your machine and could speed up CPU computations
2018-03-20 06:25:42.187342 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to
 use FMA instructions, but these are available on your machine and could speed CPU computations

Prediction results...
total number of test examples: 1
category: very positive
Accuracy: 0.78132
```

圖 4.33 模式驗證結果(非常滿意)

圖 4.34 模式驗證結果(滿意)

圖 4.35 模式驗證結果(普通)

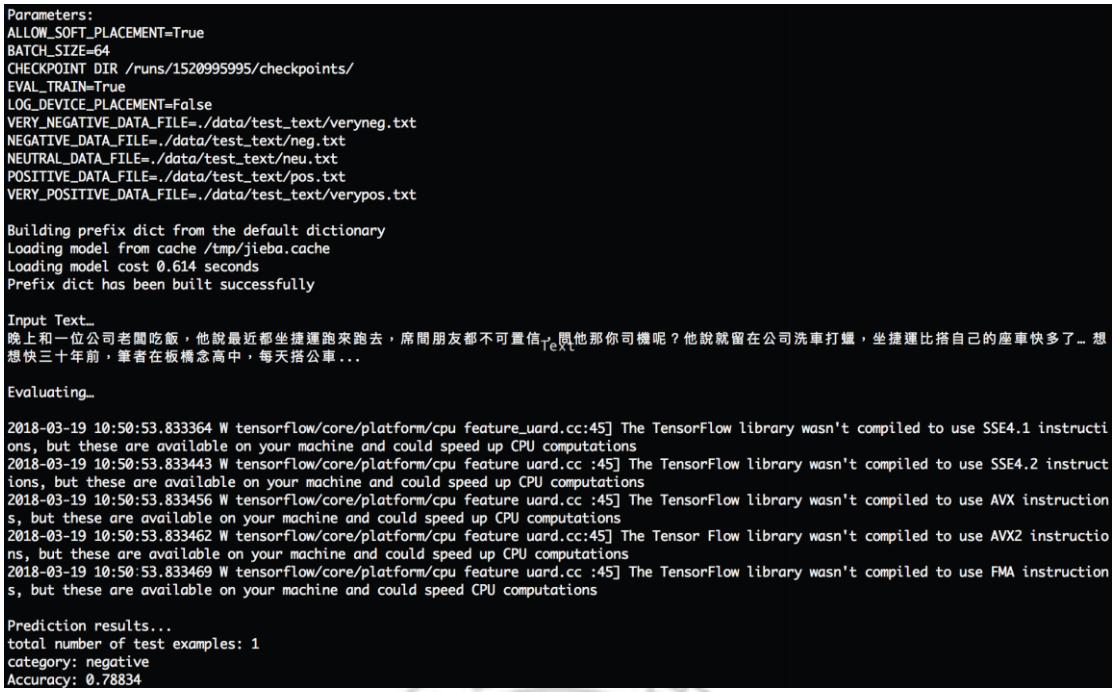

圖 4.36 模式驗證結果(不滿意)

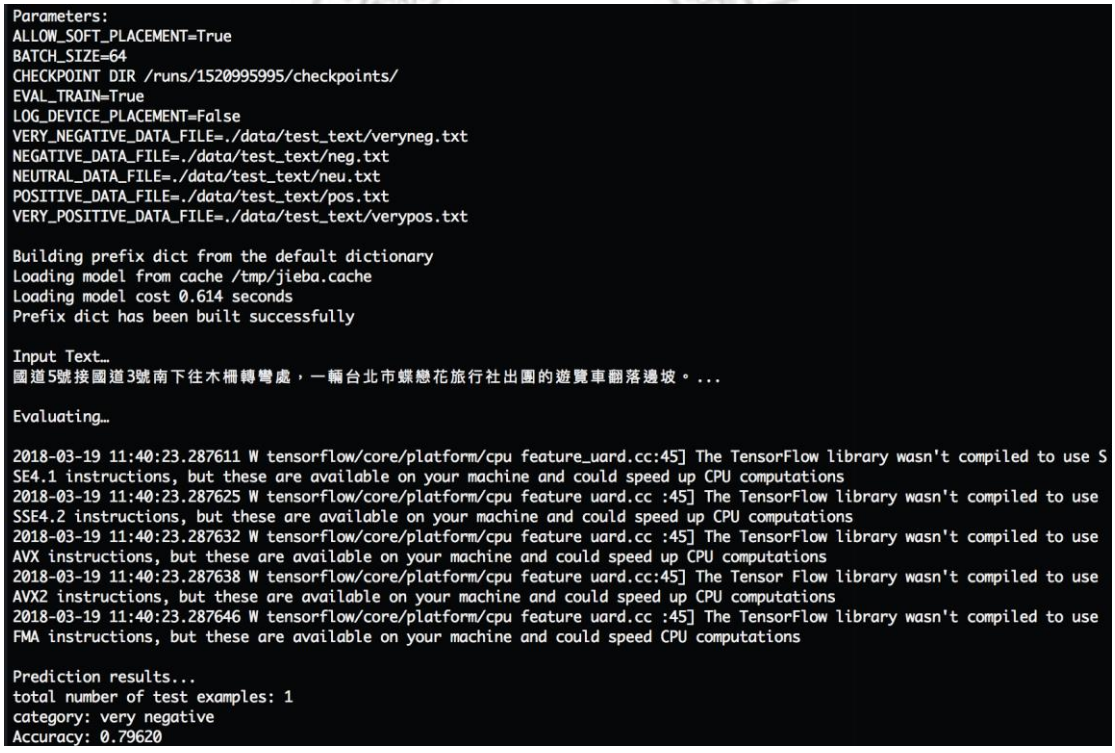

圖 4.37 模式驗證結果(非常不滿意)

經過模型預測各分類結果準確度之平均值與實際分類可列出混淆矩陣，分別如表 4.10 與表 4.11 所示，平均最高預測機率為 0.79。

表 4.10 新資料預測分類準確度

	very_pos	pos	neu	neg	very_neg
文本一	0.7962	0.79428	0.76423	0.78834	0.78132
文本二	0.78951	0.78942	0.77482	0.79114	0.77961
文本三	0.77452	0.77152	0.78712	0.77901	0.76172
文本四	0.79634	0.76891	0.78471	0.78002	0.78109
文本五	0.78561	0.77126	0.76643	0.78207	0.76001
平均值	0.788436	0.779078	0.775462	0.784116	0.77275

表 4.11 混淆矩陣

		預測值				
		Very _pos	Pos	Neu	Neg	Very _neg
實際值	Very _pos	**5 (0.79)**	0	0	0	0
	Pos	0	**5 (0.78)**	0	0	0
	Neu	0	0	**5 (0.78)**	0	0
	Neg	0	0	0	**5 (0.78)**	0
	Very _neg	0	0	0	0	**5 (0.77)**

4.6 詞頻演算法

透過前述演算法及結巴斷詞結果計算 TF-IDF 提取特徵並值修正權重，供後續 k-means 集群演算法使用，排除與本研究非相關之辭彙後，總共剩餘 41 個特徵詞，欄位分別為特徵詞、出現次數及 TF-IDF，如表 4.12 所示。

表 4.12 部分篩選完特徵詞

1	交通車	2	0.000646621
2	住宅	22	0.007112835
3	住家	8	0.002586486
4	公車	177	0.057225994
5	區間車	4	0.001293243
6	商務	6	0.001939864
7	商務型	4	0.001293243
8	商務客	2	0.000646621
9	學生	55	0.017782089
10	客運	147	0.047526673
11	專車	3	0.000969932
12	工作	176	0.056902683
13	捷運	307	0.099256385
14	校車	3	0.000969932
15	桃機	6	0.001939864
16	機捷	13	0.004203039
17	機捷線	2	0.000646621
18	火車	41	0.013255739
19	計程車	55	0.017782089
20	通勤族	10	0.003233107
21	鐵路	40	0.012932428
22	鐵道	75	0.024248303
23	國光	27	0.008729389
24	復興號	6	0.001939864
25	普悠瑪	4	0.001293243
26	高捷	3	0.000969932
27	公司	265	0.085677336
28	台鐵	207	0.066925315
29	大學	36	0.011639185
30	學校	28	0.0090527
31	家	145	0.046880052
32	上下學	2	0.000646621
33	上下班	22	0.007112835
34	上學	891	0.288069835
35	上班	170	0.054962819
36	上課	17	0.005496282

4.7 K-means 集群結果

雖然卷積神經網路建構之模式準確度尚稱優良，但僅能反映出表象結果無法探討其內部特徵，因此本研究最後使用 K-means 集群演算法，將提取之內部特徵集群，並針對集群結果做出合理解釋。

內部特徵係透過上述 TF 找出關鍵詞彙後，經過人工檢視保留與本研究相關之特徵詞彙，特徵詞共 41 個，接著利用程式比對內文出現特徵詞之次數，但中文詞彙相較英文複雜且一種詞彙有多種陳述方式(EX:司機員、司機、駕駛、駕駛員)，為了增加提取嚴謹度，將 41 個詞彙擴增至 99 個，再以人工檢視將屬性相同特徵值合併，最後利用 TF-IDF 進行特徵值權重修正，將文字資料轉換成矩陣，確認資料無問題後，最後使用 WEKA 套裝軟體執行 K-means 集群。

圖 4.38 為合併資料前之特徵詞提取，將意思相近詞彙手動加入，確保所有相關特徵詞皆被提取。

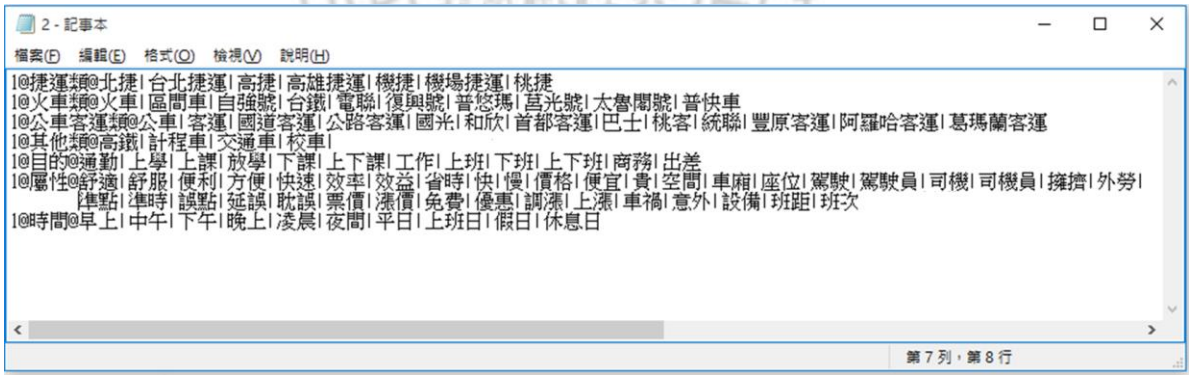

圖 4.38 擴增特徵詞

由於 K-means 於高維度資料執行有困難，且難解釋集群結果，因此將擴充詞彙按其屬性，將意思相近之特徵詞合併，將上班、下班、上下班、上課、下課、上下課合併為通勤；將快速、便利、快、慢、效率、誤點、準點合併為效率；將價格、貴、便宜、免費、優惠、漲價合併為費率；將空間、車廂、座位合併為設備；將班次、班距合併為排班；將早上、中午、下午、晚上、凌晨、夜間合併為

時段；合併結果如表 4.13 所示，最後一欄情感值為前述方法計算完畢後填入。

表 4.13 合併後部分資料

文章ID	捷運類	火車類	公車類	客運類	其他	通勤	商務	效率	舒適	費率	設備	駕駛員	意外	排班	時段	情感值
1	5	0	0	0	0	5	0	0	0	0	0	0	0	1	9	0.015981
2	0	0	0	0	2	2	0	0	0	0	0	0	0	0	0	-0.02803
3	0	0	1	0	0	5	0	0	0	0	1	0	0	0	0	-0.42717
4	0	0	3	3	0	4	0	7	0	1	0	0	0	0	2	-0.24134
5	0	1	2	0	4	1	0	7	1	0	0	0	0	0	0	0.156776
6	1	1	0	0	0	2	0	0	0	1	0	0	0	0	0	0.013269
7	0	0	0	0	1	1	0	0	0	2	0	0	0	0	2	0.073075
8	0	0	2	0	0	4	0	10	0	0	0	0	0	0	1	-0.13439
9	0	0	0	8	0	1	0	0	0	4	0	0	0	1	1	0.09543
10	0	1	0	0	0	2	0	5	0	7	0	0	0	0	4	-0.12148
11	0	4	6	0	1	1	0	4	0	0	0	0	0	4	0	0.058081
12	0	0	0	1	0	2	0	1	0	0	0	0	0	0	0	0.060035
13	0	5	5	0	0	1	0	2	0	0	0	0	0	3	0	0.042272
14	0	1	0	0	0	1	0	2	0	0	0	0	0	0	0	-0.05383
15	0	0	16	0	0	1	0	0	0	8	0	0	0	0	0	-0.24479
16	0	1	0	0	0	4	0	2	1	1	2	0	0	0	1	-0.66392
17	0	0	0	17	0	4	0	2	0	18	0	3	0	0	0	0.033109
18	0	1	2	0	0	1	0	1	0	0	0	0	0	1	0	-0.02042
19	0	1	0	0	0	2	0	0	0	1	0	0	0	0	0	0.035387
20	0	2	0	0	0	2	0	2	0	0	0	0	0	0	0	-0.06806
21	0	2	13	5	1	1	0	6	0	0	0	0	0	0	0	0.062652
22	11	0	0	0	0	1	0	2	0	0	6	8	0	0	0	0.045296
23	0	0	5	3	0	1	0	5	0	0	0	11	0	2	0	-0.31781
24	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0.021997
25	0	0	0	0	5	1	0	0	0	0	0	0	0	0	0	-0.5007
26	0	0	2	1	0	1	0	0	0	0	0	0	0	0	0	0.024648
27	0	0	1	0	0	1	0	3	0	1	0	0	0	0	0	-0.05042
28	0	0	0	0	15	1	0	4	0	0	3	11	0	0	0	-0.25904
29	0	0	0	0	2	2	0	0	0	1	0	0	0	0	0	0.075947
30	0	0	1	0	0	1	0	3	0	1	0	0	0	1	1	0.035287
31	0	1	0	0	0	2	2	7	2	1	4	4	0	0	2	-0.23431
32	0	2	0	0	0	1	0	0	0	0	0	1	0	0	0	0.008018
33	0	1	3	0	1	1	1	10	0	0	14	0	0	0	2	-1.09696
34	0	13	0	0	0	1	0	0	0	0	0	4	0	0	0	-0.06403
35	0	0	11	0	0	1	0	0	0	0	0	0	0	0	0	0.034748

為決定 K-means 最佳分群數 K，本研究以華德法(Ward's Method)搭配歐式距離進行分析，華德法是將每一個體視為一個群，將各群依序合併，合併順序照合併後群體組內總變異數大小決定。群內總變數產生最小變化量之個體優先合併，判斷最佳分群數方式有 2 種，第一種為判斷凝聚過程之中係數增加量，若係數增加量有明顯提升表示將最相似的兩群合併為一群，也會大幅度提升群之總變異量，因此不宜再繼續凝聚，第二種為觀察階層圖，經由觀察華德法連結的樹狀圖，能客觀判斷最佳分群數。

係數增加率在 3、4、5 群較趨緩，在 2 群時有較大的變化，1 群變化最劇烈，表示最佳分群數為 3，但為了確保最佳分群數正確，因此再以試誤法嘗試跑 3、4、5 群，發現最佳分群數為 4 時，能有最佳解釋能力，因此本研究選定最佳分群數為 4。

表 4.14 華德法凝聚過程表

階段	群數	係數	係數變化量	係數增加率
1096	8	64992.687	-	-
1097	7	70112.378	5119.691	7.88%
1098	6	75866.661	5754.283	8.21%
1099	5	81732.033	5865.371	7.73%
1100	4	88153.370	6421.337	7.86%
1101	3	94907.615	6754.246	7.66%
1102	2	104776.983	9869.368	10.40%
1103	1	119664.623	14887.640	14.21%

本研究將集群結果彙整成表 4.15 及表 4.16，說明如下:

集群 1 占總樣本 4.53%，搭乘運具以其他(計程車、交通車、校車)(1.7%)、火車類(2.322)為主，旅次目的以商務(0.36%)為主，情感值低於平均為-0.2321(普通滿意)，此群關注的變數有：效率(4.74%)、舒適(2.08%)、設備(4.1%)、是否發生意外(0.2%)、時段(3.4%)；較不關注的變數有：費率(0.37%)、駕駛員(0.5%)、排班(0.59%)。

集群 2 筆數最多占總樣本 85.42%，搭乘運具以捷運(1.14%)為主，旅次目的以通勤(2.98%)為主，情感值高於平均為 0.12(普通滿意)，此群沒有關注的變數；較不關注的變數有：效率(1.81%)、舒適(0.04%)、費率(1.29%)、設備(0.41%)、駕駛員(0.56%)、意外(0.09%)、排班(0.08%)、時段(1.25%)。

集群 3 占總樣本 3.44%，搭乘運具以公車類(2.58%)、客運類(12.05%)為主，旅次目的以通勤(2.73%)為主，情感值低於平均為-0.247(普通滿意)，此群關注的變數有：效率(2.13%)、設備(0.59%)、駕駛員(2.26%)、意外(0.29%)、排班 (0.37%)；較不關注的變數有：舒適(0.12%)、費率(0.33%)、時段(0.73%)。

集群 4 占總樣本 6.61%，搭乘運具以火車類(8.04%)、公車類(2.75%)為主，旅次目的以通勤(2.79%)為主，情感值低於平均為-0.2089(普通滿意)，此群關注的變數有：效率(4.19%)、駕駛員(0.64%)、排班(2.27%)、時段(0.99%)；較不關注的變數有：舒適(0.04%)、費率(0.29%)、設備(0.56%)、意外(0.17%)。

表 4.15 分群比較表(*表示平均值是最大或最小的一群)

分群	第1群	第2群	第3群	第4群
大於變數平均值	Var1.火車類 Var4.其他類* Var6.商務* Var7.效率* Var8.舒適* Var10.設備* Var12.意外 Var14.時段*	筆數* Var0.捷運類* Var5.通勤* Var15.情感值*	Var2.公車類 Var3.客運類* Var5.通勤 Var7.效率 Var10.設備 Var11.駕駛員* Var12.意外* Var13.排班	Var1.火車類* Var2.公車類* Var5.通勤 Var7.效率 Var11.駕駛員 Var13.排班* Var14.時段
小於變數平均值	筆數 Var0.捷運類 Var2.公車類* Var3.客運類* Var5.通勤* Var9.費率 Var11.駕駛員* Var13.排班 Var15.情感值	Var1.火車類* Var2.公車類 Var3.客運類 Var4.其他類 Var6.商務* Var7.效率* Var8.舒適* Var9.費率 Var10.設備* Var11.駕駛員 Var12.意外* Var13.排班* Var14.時段*	筆數* Var0.捷運類* Var1.火車類 Var4.其他類 Var6.商務 Var8.舒適 Var9.費率 Var14.時段 Var15.情感值*	筆數 Var0.捷運類 Var3.客運類 Var4.其他類* Var6.商務 Var8.舒適 Var9.費率* Var10.設備 Var12.意外 Var15.情感值

表 4.16 K-means 分群特徵分布(粗體表示該變數最大或最小值)

變項	全部資料	第1群	第2群	第3群	第4群
筆數	1104(平均: 276)	50	**943**	**38**	73
var0: 捷運類 (Avg.)	1.0942	0.8	**1.1362**	**0.3425**	0.7684
var0: 捷運類 (Std.)	4.5834	2.0396	4.7977	3.2861	5.9753
var1: 火車類 (Avg.)	1.4149	2.322	**1**	1.18	**8.0411**
var1: 火車類 (Std.)	3.3049	2.2063	3.2153	1.3955	8.4081
var2: 公車類 (Avg.)	1.0389	**0.58**	0.8266	2.5789	**2.7534**
var2: 公車類 (Std.)	2.6545	1.0971	1.9479	5.1841	5.7261
var3: 客運類 (Avg.)	0.7482	**0.2972**	0.6027	**12.053**	0.3
var3: 客運類 (Std.)	2.4917	0.755	0.8205	4.8609	1.3006
var4: 其他 (Avg.)	0.6929	**1.7013**	0.6233	0.5105	**0.4808**
var4: 其他 (Std.)	2.1567	1.1517	1.8408	1.4125	4.9248
var5: 通勤 (Avg.)	2.7355	**2.68**	**2.9771**	2.7368	2.7877
var5: 通勤 (Std.)	3.1156	2.3953	3.2297	2.0734	2.7572
var6: 商務 (Avg.)	0.0643	**0.36**	**0.0137**	0.0168	0.0454
var6: 商務 (Std.)	0.482	0.9749	0.4334	0.7411	0.1162
var7: 效率 (Avg.)	2.0833	**4.74**	**1.808**	2.1263	4.1918
var7: 效率 (Std.)	3.806	5.4619	3.2384	2.7481	7.6044
var8: 舒適 (Avg.)	0.1458	**2.08**	**0.0411**	0.1195	0.0444
var8: 舒適 (Std.)	0.5329	1.0741	0.2059	0.603	0.1985
var9: 費率 (Avg.)	1.4475	0.3714	1.291	0.3263	**0.2945**
var9: 費率 (Std.)	3.7262	3.4928	3.2729	9.9986	4.9765
var10: 設備 (Avg.)	0.5906	**4.1**	**0.4087**	0.5993	0.5616
var10: 設備 (Std.)	1.5021	4.0902	0.9702	1.0696	1.1819
var11: 駕駛員 (Avg.)	0.635	**0.5**	0.5645	**2.2632**	0.6391
var11: 駕駛員 (Std.)	3.0645	1.2689	3.1308	3.4541	1.8395
var12: 意外 (Avg.)	0.1712	0.2	**0.0959**	**0.2895**	0.1672
var12: 意外 (Std.)	0.6972	0.5292	0.6986	1.049	0.3378
var13: 排班 (Avg.)	0.2111	0.12	**0.0774**	0.3684	**2.274**
var13: 排班 (Std.)	0.7684	0.5879	0.3101	0.625	1.8076
var14: 時段 (Avg.)	0.9882	**3.4**	**1.2466**	0.7263	0.9922
var14: 時段 (Std.)	2.6721	6.3498	1.9408	1.5975	2.9784
var15: 情感值 (Avg.)	-0.1187	-0.2321	**0.1171**	**-0.247**	-0.2089
var15: 情感值 (Std.)	0.3128	0.4518	0.3126	0.1262	0.2178

4.8 管理意涵

根據表 4.15 與 4.16 之集群結果可看出，不同旅次目的及搭乘運具之關注變數不盡相同,因此本研究回顧相關文獻,針對不同目的及運具之可能不滿意項目，提出對應之改善方案，以提高滿意度之情感值。

搭乘火車類族群較關注：是否能準時到達目的(效率)、搭乘過程是否舒適(舒適)、使用運具設備是否完善(設備)、排班、搭乘運具時段(時段)。搭乘捷運之通勤族，則較無特別關注的變數，表示捷運於各方面表現均尚滿意。

搭乘公車、客運族群較關注：是否能準時到達目的(效率)、使用運具設備是否完善(設備)、駕駛員、是否發生意外(意外)、排班。搭乘其他(計程車、校車、交通車)族群較關注：是否能準時到達目的(效率)、駕駛員、排班、搭乘運具時段(時段)。綜合上述，本研究提出相對應之改善方案可彙整成表 4.17。

表 4.17 各運具滿意度之改善方案

運具	關注變數	改善方案
捷運類(北捷、機捷、高捷)	無	目前無需改善
火車類(台鐵、高鐵)	效率、設備、排班、時段、舒適	1. 搭乘時段 通勤時段集中，乘客量大，須加開班次消化大量通勤民眾 2. 乘客過多或個別因素導致列車誤點 乘客速度慢可能是車站動線不佳，且列車應提早到站，給乘客更多時間上下車 3. 車廂整潔 通勤時段乘客較多，易造成環境髒亂，且廁所使用量大，於通勤時段須加強整體車廂清潔 4. 提升並保養車內軟硬體設備 定期保養車內座椅、車上跑馬燈損毀、無線網路連線品質等 (2015, 成大研究)
公車類 客運類(國道/市區客運)	效率、設備、駕駛員、意外、排班	1. 提升並保養車內軟硬體設備 定期保養車內老舊座椅、車上燈光昏暗、影音設備軟體更新、空調舒適度等 2. 按時發車不脫班 可透過最佳化配置班表，並規範司機員準時進站離站 3. 加強司機教育訓練 司機不和善、乘客抱怨未回應、未供乘客搭乘相關資訊、司機不專心行駛、車子行駛發生意外 (2013, 陳宏佳)
其他類(計程車、交通車、校車)	效率、舒適、設備、意外、時段	1. 提升並保養車子軟硬體設備 定期保養車內設備及車體外觀 2. 加強司機教育訓練 司機駕車平穩並準時將乘客送至目的地

第五章 結論

本研究係針對日常交通下社群媒體進行文本挖掘，首先透過建置社群媒體文本資料庫，並將日常交通之社群媒體文本進行情感分析，以瞭解民眾對於日常交通下各運具評論之情緒，接著使用人工智慧演算法之 CNN 建構多元情緒決策情感分析模型。透過實證分析，本研究建立一套社群媒體輿情分析程序，運用文本挖掘及深度學習分類技術，分析民眾日常通勤通學搭乘公共運具之滿意度，並建構滿意度評級尺度，找出民眾搭乘不同公共運具之整體情緒，最後使用集群演算法探討社群媒體輿情內部特徵。然而此研究成果僅代表部份民眾之真實意見，可供決策者了解問題所在。本研究歸納出以下重要結論及建議：

5.1 結論

一、本研究應用文本挖掘技術建立日常交通下社群媒體文本資料庫，接著透過結巴(jieba)斷詞系統及詞頻演算法(Term-Frequency)，初步找出特徵詞彙，再經由詞頻演算法及逆向檔案頻率法(Inverse Document Frequency)修正權重，經由人工檢視後，找出與本研究相關之特徵詞彙共 41 個。

貢獻在於：未來若有交通通勤相關之社群媒體中文文本挖掘，可參考本研究之分析流程，針對樣本文本資料進行蒐集、預處理、標準化過程，後完成特徵選擇，節省以往研究探索之時間。

二、本研究根據台灣大學中文情感極性辭典(NTUSD)比對文本出現之情感詞彙，經由 TF-IDF 修正權重後，將情感特徵詞加總得到各文本整體情感值，並建立滿意度區間(共五評級)，找出各公共運具情感值，其結果如下：

1. 各公共運具滿意度區間多落於普通滿意，就合併觀察集群結果與社群媒體文本內容，可知通勤通學民眾對於公共運具的使用有其必要性與依賴

性，故雖常因擁擠、誤點等因素產生負面情感，但負面情感之表現幅度較小且多，整體上民眾仍尚滿意公共運具服務，惟造成負面情感之因素對於管理與改善營運效益上，仍具參考價值。

2. 大部分每月負文本累積數多於正，根據陳亭愷(2015)與蔡易辰(2016)相關研究結果，得知社群媒體文本在情感表現上偏向負面屬於常見之情況，惟分析上應針對造成負面情感之構成因素進行探討，如此可利於提出改善意見。

3. 情感值擺盪幅度由大到小依次為：其他類(計程車、交通車、校車)、火車類(台鐵、高鐵)、公車客運類、捷運類，其中其他類造成情感值擺盪幅度較大推測原因為其他類主要係以商業目的為主，一旦於搭乘運具過程中受到效率、舒適、設備、意外、時段某一關注變數影響，造成其商業損失或失敗較通勤通學族群則有更劇烈情緒反應。

三、本研究使用深度學習之卷積神經網路建立情感識別模型，將文本依照滿意度區間歸納完後，輸入機器自動訓練，訓練 2800 次後收斂，在五個滿意度區間下，模式準確度達 0.784946。

貢獻在於：本研究建置之 CNN 情感識別模型，可應用於未來相關議題之新文本分析，除了可透過過去用於訓練之資料進行新文本之情感區間落點判別之外，亦可將新文本作為樣本資料進行訓練，豐富本模型之情感判別能力，使本研究建立之社群媒體挖掘與情感分析通用程序具有擴展性，此可供決策者藉由模型的修正與改良而能快速掌握網路輿情正、負面情感趨勢之即時資訊。

四、CNN 模式準確度尚稱優良，但僅能反映出表象結果無法探討其內部特徵，因此本研究最後使用 K-means 演算法，將提取之內部特徵集群，並針對集群結果提供合理解釋。集群分析結果如下：

集群 1 之旅次目的為商務，且搭乘運具為計程車、交通車，情感區間落於普通滿意，商務旅次較關注：是否能準時到達目的(效率)、搭乘過程是否舒適(舒適)、使用運具設備是否完善(設備)、是否發生意外(意外)、搭乘運具時段(時段)；商務旅次較不關注：運具之價格(費率)、駕駛員、排班。

集群 2 為搭乘捷運之通勤族，筆數最多，情感值區間落於普通滿意，此群沒有關注的變數，各項變數對捷運通勤族皆無太大影響。

集群 3 為搭乘公車、客運之通勤族，情感值區間落於普通滿意，此群較關注：是否能準時到達目的(效率)、使用運具設備是否完善(設備)、駕駛員、是否發生意外(意外)、排班；較不關注的變數有：搭乘過程是否舒適(舒適)、運具之價格(費率)、搭乘運具時段(時段)。

集群 4 為搭乘火車、公車之通勤族，情感值區間落於普通滿意，此群較關注：是否能準時到達目的(效率)、駕駛員、排班、搭乘運具時段(時段)；較不關注的變數有：搭乘過程是否舒適(舒適)、運具之價格(費率)、使用運具設備是否完善(設備)、是否發生意外(意外)。

綜合上述結果，不論通勤或商旅旅次皆關注是否能準時到達目的，顯示民眾最為重視搭乘大眾運具之效率；而搭乘火車類、公車類、客運類之通勤族皆關注駕駛員及排班，顯示民眾重視資訊的掌握與服務及乘車安全性的好壞，可知若特定時間的排班常有誤點情況時，調整到站時間資訊的呈現有望減少民眾的負面情感表現，而駕駛員的素質、服務態度、應對能力等，對於民眾的影響亦相當直接，故對於駕駛員系統性的服務訓練亦有望提升民眾的正面情感；最後，目前所有通勤族群對於費率的關注程度皆不明顯，顯示目前我國公共運具在費率的制定上皆受民眾肯定，若未來產生費率調漲之政策需求時，亦可經由社群媒體輿情分析加以驗證，決定費率是否調漲，以此輔助相關單位進行決策。

5.2 建議

一、本研究之樣本來源目前以社群媒體為主，如：FB、PTT、各大新聞網站下方之留言內容等，考慮民眾發言內容受對象與事件的不同有所差異，未來可針對不同面向之社群媒體文本進行分析與應用，如：觀光旅遊(遊覽車服務、高鐵N日遊、火車觀光旅行等)、交通政策(重大節日費率調整、班次加開等)、社會議題(政治人物輿情民調、抗爭事件)等，皆可依據本研究流程進行分析與應用。

二、目前本研究所建置之爬蟲系統抓取對象以社群媒體網站為主，考慮未來應用的擴展與分析面向的增加，社群媒體文本的來源亦可持續增加，如：Tripadvisor、Twitter、Trivago 等大型討論網站下之留言內容進行抓取，以利分析內容更加全面。

三、由於本研究實證人工智慧之 CNN 演算法可應用於社群媒體文本之情感識別分析流程，後續相關研究亦可嘗試以本流程與模型加以改進，依據需求調整人工智慧演算法內容(如:增強學習演算法)針對不同領域而擴充。

參考文獻

中英文文獻

1. 劉文良(2007)，網際網路行銷(第二版)，台北：碁峰資訊。

2. 張斐章、張麗秋(2015)，類神經網路導論原理與應用，蒼海圖書資訊股份有限公司(第二版) 。

3. 鄭捷(2016)，機器學習-算法原理與編程實踐，北京：電子工業出版社。

4. 陳亭愷(2015)，國道計程電子收費實施後之網路社群媒體文本情感分析研究，淡江大學運輸管理學系運輸科學碩士班碩士論文。

5. 蔡易辰(2016)，三元決策理論應用於國道計程收費議題之情感分析研究，淡江大學運輸管理學系運輸科學碩士班碩士論文。

6. 洪士耕(2011)，社群媒體之關係行銷研究－以7-Eleven之Facebook粉絲專頁使用者為例，中國文化大學新聞暨傳播學院學新聞學系碩士論文。

7. 交通部統計處(2017)，民眾日常使用運具狀況調查摘要分析。

8. 交通部統計處(2017)，交通部施政措施滿意度調查摘要分析。

9. 楊雅惠(2011)，使用媒體豐富理論與網路互動理論解釋企業使用社群媒體行為，國立高雄第一科技大學資訊管理系碩士論文。

10. 李嘉洲(2016)，應用深度學習於財經新聞來源對股價趨勢預測之研究，淡江大學資訊管理學系碩士在職專班碩士論文。

11. 李宏毅(2016)，「什麼是深度學習」，*數理人文*，第十期。

12. 伊寶才，王文通，王立春(2015)，「深度學習綜述」，*北京工業大學學報* ，Vol.41，No.1。

13. 孫志軍，薛磊，許陽明，王正(2012)，「深度學習研究綜述」，*計算機應用研*

究 ，Vol.29，No.8。

14. 郭俊桔(2012)，使用情緒分析於圖書館使用者滿意度評估之研究，國立中興大學圖書資訊學研究所碩士學位論文。

15. 謝衫蒂(2014)，應用機器學習與多辭典的中英雙語意見分析之研究，淡江大學資訊管理學系在職專班碩士論文。

16. 蔡國永, 夏彬彬(2016)，「基於卷積神經網路的圖文融合媒體情感預測」，*計算機應用*，第二期，頁428-431。

17. 梁軍, 柴玉梅, 原慧斌, 昝紅英, 劉銘(2014)，「基於深度學習的微博情感分析」，*中文信息學報*，第五期，頁155-161。

18. 孫瑛澤, 陳建良, 劉峻杰, 劉昭麟, 蘇豐文(2010)，中文短句之情緒分類，第12屆自然語言與語音處理研討會中華民國計算語言學學會、國立國際暨南大學資工系與電機系。

19. 方濱興、許進、李建華(2014)，在線社交網路分析，北京：電子工業出版社。

20. 于冬梅(2009)，情感計算關鍵技術研究，東華大學信息科學與技術學院博士論文。

21. 鄒以勤(2012)，顧客滿意、服務特性與回購意願關係之研究：以台灣高速鐵路為例，國立成功大學交通管理科學系碩士論文。

22. 陳建成(2008)，「台灣高鐵服務品質及服務價值與顧客滿意度關係之實證研究」，*數據分析期刊*，3卷3期，頁95-116。

23. Arisoy, E., Sainath, T., Kingsbury, B., Ramabhadran, B. (2012), "Deep Neural Network Language Models," *In Proceeding of the Joint Human Language Technology Conference and the North American Chapter of the Association of Computational Language (HLT-NAACL) Workshop.*

24. Cambria, E. and White, B. (2014), "Jumping NLP curves: A Review of Natural Language Processing Research," *IEEE Computational Intelligence*

Magazine, Vol. 9, No. 2, 48–57.

25. Chen, Y.(2015),"Convolutional Neural Networks for Sentence Classification," A thesis presented to the University of Waterloo in fulfillment of the thesis requirement for the degree of Master of Mathematics in Computer Science.

26. Ellison, N. B.(2007),"The Benefits of Face book Friends: Social Capital and College Students Use of Online Social Network Sites," *Journal of Computer-Mediated Communication*, 12, 1143-1168.

27. Kim, Y. M., N.W. G.(2008),"Politics as Friendship: The Impact of Online Social Networks on Young Voters' Political Behavior," *Proceeding of Conference on International Communication Association*, 22-26 May 2008, Montreal: TBA.

28. Kushin, M. J.(2010),"Did Social Media Really Matter? College Students' Use of Online Media and Political Decision Making in the 2008 Election," *Mass Communication and Society*,13, 5: 608-630.

29. Severyn, A. & Moschitti, A. (2015), "Training Deep Convolutional Neural Network for Twitter Sentiment Classification," *The 9th International Workshop on Semantic Evaluation Proceedings of SemEval*, 2015, 464-469.

30. Soujanya, P., Erik, C., Iti, C. & Amir, H. (2017), "Convolutional MKL Based Multimodal Emotion Recognition and Sentiment Analysis", *2016 IEEE 16th International Conference on Data Mining*.

31. Strapparava, C. & Mihalcea, R. (2008),"Learning to Identify Emotions in Text," *SAC '08 Proceedings of the 2008 ACM symposium on Applied Computing*, 1556-1560.

32. Wu, C.-H. and Liang, W.-B. (2011), "Emotion Recognition of Affective Speech based on Multiple Classifiers Using Acoustic-prosodic Information and Semantic Labels," *IEEE Transactions on Affective Computing*, Vol. 2, No. 1, 10–21.

33. Yassine, M. & Hajj, H. (2010),“A Framework for Emotion Mining from Text in Online Social Networks,” *ICDMW '10 Proceedings of 2010 IEEE International Conference on Data Mining Workshops*,1136-1142.

34. Zuo, Z., Shuai, B., Wang, G., Liu, X., Wang, X., Wang, B., & Chen, Y. (2015), ”Convolutional Recurrent Neural Networks: Learning Spatial Dependencies or Image Representation,” *2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops*, Boston, MA, USA, 18–26.

網路文獻

1. Brandon(2016)，卷積神經網路的運作原理，https://brohrer.mcknote.com/zh-Hant/how_machine_learning_works/how_convolutional_neural_networks_work.html

2. 王十二的(2017)，NLP與文本分類跟我們有什麼關係，http://www.jianshu.com/p/12de34cec389

3. 紅黑聯盟(2017)，深度學習筆記五：卷積神經網絡CNN，https://read01.com/zh-tw/RE37gy.html#.WgktBWiCxhF

4. CSDN博客(2016)，對卷積神經網絡(CNN)的簡單理解，https://read01.com/zh-tw/2zA0Po.html#.Wgktw2iCxhF

5. 維基百科，深度學習，https://zh.wikipedia.org/wiki/%E6%B7%B1%E5%BA%A6%E5%AD%A6%E4%B9%A0

6. 中研院智財技轉處，中文斷詞系統，http://ckipsvr.iis.sinica.edu.tw/

7. 資策會創研所(2017)，台灣人特愛用FB、Line平均擁4個社群帳號，https://www.find.org.tw/market_info.aspx?n_ID=9095

8. 林志傑(2015)，結巴中文斷詞，https://speakerdeck.com/fukuball/jieba-jie-ba-zhong-wen-duan-ci

9. Denny Britz(2015)，Implementing a CNN for Text Classification in TensorFlow，http://www.wildml.com/2015/12/implementing-a-cnn-for-text-classification-in-tensorflow/

10. 陳勇汀(2011)，Weka的K-Means分群演算法使用教學，http://blog.pulipuli.info/2016/12/wekak-meanssimplekmeans-clustering-with.html

11. Mr. Opengate(2016)，AI-Ch16.5類神經網路實作-TensorFlow-介紹與入門教學，http://mropengate.blogspot.tw/2016/10/ai-ch165-tensorflow.html

12. 王浩全(2015)，社群運算兩三事，https://www.facebook.com/notes/hao-chuan-wang/%E7%A4%BE%E7%BE%A4%E9%81%8B%E7%AE%97-social-computing-%E5%85%A9%E4%B8%89%E4%BA%8B/10153032839723256/

13. Kobe Chen(2016)，3分鐘搞懂深度學習到底在深什麼，https://panx.asia/archives/53209

14. 大數據文摘(2017)，手把手範例+代碼：一文帶你上手Python網頁抓取神器BeautifulSoup庫，https://buzzorange.com/techorange/2017/08/04/python-scraping/

15. 詹雨安(2018)，Deep Learning with Python by François Chollet，https://www.facebook.com/permalink.php?story_fbid=2244278798932232&id=100000504010703

16. 中國雲計算(2017)，用深度學習(CNN RNN Attention)解決大規模文本分類問題-綜述和實踐，https://kknews.cc/zh-tw/tech/85qkzrq.html

17. YG(2017)，Tensorflow文本分類-Python深度學習，https://hk.saowen.com/a/d1306fa248106f8af843fc2b16dbb73374e12a1055bbf5af878e0faeaf074580

附錄 A 模式驗證結果

```
Parameters:
ALLOW_SOFT_PLACEMENT=True
BATCH_SIZE=64
CHECKPOINT DIR /runs/1520995995/checkpoints/
EVAL_TRAIN=True
LOG_DEVICE_PLACEMENT=False
VERY_NEGATIVE_DATA_FILE=./data/test_text/veryneg.txt
NEGATIVE_DATA_FILE=./data/test_text/neg.txt
NEUTRAL_DATA_FILE=./data/test_text/neu.txt
POSITIVE_DATA_FILE=./data/test_text/pos.txt
VERY_POSITIVE_DATA_FILE=./data/test_text/verypos.txt

Building prefix dict from the default dictionary
Loading model from cache /tmp/jieba.cache
Loading model cost 0.614 seconds
Prefix dict has been built successfully

Input Text…
【勁報記者羅蔚舟／新竹報導】中華大學同學們5/16日在高鐵新竹站大廳上演一...

Evaluating…

2018-03-20 06:26:51.721451 W tensorflow/core/platform/cpu feature_uard.cc:45] The TensorFlow library wasn't compiled to
use SSE4.1 instructions, but these are available on your machine and could speed up CPU computations
2018-03-20 06:26:51.721463 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to
 use SSE4.2 instructions, but these are available on your machine and could speed up CPU computations
2018-03-20 06:26:51.721470 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to
 use AVX instructions, but these are available on your machine and could speed up CPU computations
2018-03-20 06:26:51.721478 W tensorflow/core/platform/cpu feature uard.cc:45] The Tensor Flow library wasn't compiled to
 use AVX2 instructions, but these are available on your machine and could speed up CPU computations
2018-03-20 06:26:51.721482 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to
 use FMA instructions, but these are available on your machine and could speed CPU computations

Prediction results...
total number of test examples: 1
category: very positive
Accuracy: 0.77961
```

圖 A 模式驗證結果(非常滿意 2)

```
Parameters:
ALLOW_SOFT_PLACEMENT=True
BATCH_SIZE=64
CHECKPOINT DIR /runs/1520995995/checkpoints/
EVAL_TRAIN=True
LOG_DEVICE_PLACEMENT=False
VERY_NEGATIVE_DATA_FILE=./data/test_text/veryneg.txt
NEGATIVE_DATA_FILE=./data/test_text/neg.txt
NEUTRAL_DATA_FILE=./data/test_text/neu.txt
POSITIVE_DATA_FILE=./data/test_text/pos.txt
VERY_POSITIVE_DATA_FILE=./data/test_text/verypos.txt

Building prefix dict from the default dictionary
Loading model from cache /tmp/jieba.cache
Loading model cost 0.614 seconds
Prefix dict has been built successfully

Input Text…
      光是想到開始收拾行李，前往機場我就感到開心！       我跟外子都是上班族...

Evaluating…

2018-03-20 06:28:43.361211 W tensorflow/core/platform/cpu feature_uard.cc:45] The TensorFlow library wasn't compiled to
use SSE4.1 instructions, but these are available on your machine and could speed up CPU computations
2018-03-20 06:28:43.361218 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to
 use SSE4.2 instructions, but these are available on your machine and could speed up CPU computations
2018-03-20 06:28:43.361223 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to
 use AVX instructions, but these are available on your machine and could speed up CPU computations
2018-03-20 06:28:43.361228 W tensorflow/core/platform/cpu feature uard.cc:45] The Tensor Flow library wasn't compiled to
 use AVX2 instructions, but these are available on your machine and could speed up CPU computations
2018-03-20 06:28:43.361234 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to
 use FMA instructions, but these are available on your machine and could speed CPU computations

Prediction results...
total number of test examples: 1
category: very positive
Accuracy: 0.76172
```

圖 B 模式驗證結果(非常滿意 3)

```
Parameters:
ALLOW_SOFT_PLACEMENT=True
BATCH_SIZE=64
CHECKPOINT DIR /runs/1520995995/checkpoints/
EVAL_TRAIN=True
LOG_DEVICE_PLACEMENT=False
VERY_NEGATIVE_DATA_FILE=./data/test_text/veryneg.txt
NEGATIVE_DATA_FILE=./data/test_text/neg.txt
NEUTRAL_DATA_FILE=./data/test_text/neu.txt
POSITIVE_DATA_FILE=./data/test_text/pos.txt
VERY_POSITIVE_DATA_FILE=./data/test_text/verypos.txt

Building prefix dict from the default dictionary
Loading model from cache /tmp/jieba.cache
Loading model cost 0.614 seconds
Prefix dict has been built successfully

Input Text…
雙北1280元大眾運輸定期票開賣滿1週，台北市交通局今天表示，截至昨天，預購數達2萬...

Evaluating…

2018-03-20 06:29:56.873116 W tensorflow/core/platform/cpu feature_uard.cc:45] The TensorFlow library wasn't compiled to
use SSE4.1 instructions, but these are available on your machine and could speed up CPU computations
2018-03-20 06:29:56.873126 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to
 use SSE4.2 instructions, but these are available on your machine and could speed up CPU computations
2018-03-20 06:29:56.873137 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to
 use AVX instructions, but these are available on your machine and could speed up CPU computations
2018-03-20 06:29:56.873143 W tensorflow/core/platform/cpu feature uard.cc:45] The Tensor Flow library wasn't compiled to
 use AVX2 instructions, but these are available on your machine and could speed up CPU computations
2018-03-20 06:29:56.873149 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to
 use FMA instructions, but these are available on your machine and could speed CPU computations

Prediction results...
total number of test examples: 1
category: very positive
Accuracy: 0.78109
```

圖 C 模式驗證結果(非常滿意 4)

```
Parameters:
ALLOW_SOFT_PLACEMENT=True
BATCH_SIZE=64
CHECKPOINT DIR /runs/1520995995/checkpoints/
EVAL_TRAIN=True
LOG_DEVICE_PLACEMENT=False
VERY_NEGATIVE_DATA_FILE=./data/test_text/veryneg.txt
NEGATIVE_DATA_FILE=./data/test_text/neg.txt
NEUTRAL_DATA_FILE=./data/test_text/neu.txt
POSITIVE_DATA_FILE=./data/test_text/pos.txt
VERY_POSITIVE_DATA_FILE=./data/test_text/verypos.txt

Building prefix dict from the default dictionary
Loading model from cache /tmp/jieba.cache
Loading model cost 0.614 seconds
Prefix dict has been built successfully

Input Text…
近年外國觀光客變多，就有一名網友發現不論是網紅、一般遊客或是留學生，都只會待在台北...

Evaluating…

2018-03-20 06:31:42.912151 W tensorflow/core/platform/cpu feature_uard.cc:45] The TensorFlow library wasn't compiled to
use SSE4.1 instructions, but these are available on your machine and could speed up CPU computations
2018-03-20 06:31:42.912160 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to
 use SSE4.2 instructions, but these are available on your machine and could speed up CPU computations
2018-03-20 06:31:42.912164 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to
 use AVX instructions, but these are available on your machine and could speed up CPU computations
2018-03-20 06:31:42.912169 W tensorflow/core/platform/cpu feature uard.cc:45] The Tensor Flow library wasn't compiled to
 use AVX2 instructions, but these are available on your machine and could speed up CPU computations
2018-03-20 06:31:42.912173 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to
 use FMA instructions, but these are available on your machine and could speed CPU computations

Prediction results...
total number of test examples: 1
category: very positive
Accuracy: 0.76001
```

圖 D 模式驗證結果(非常滿意 5)

```
Parameters:
ALLOW_SOFT_PLACEMENT=True
BATCH_SIZE=64
CHECKPOINT DIR /runs/1520995995/checkpoints/
EVAL_TRAIN=True
LOG_DEVICE_PLACEMENT=False
VERY_NEGATIVE_DATA_FILE=./data/test_text/veryneg.txt
NEGATIVE_DATA_FILE=./data/test_text/neg.txt
NEUTRAL_DATA_FILE=./data/test_text/neu.txt
POSITIVE_DATA_FILE=./data/test_text/pos.txt
VERY_POSITIVE_DATA_FILE=./data/test_text/verypos.txt

Building prefix dict from the default dictionary
Loading model from cache /tmp/jieba.cache
Loading model cost 0.614 seconds
Prefix dict has been built successfully

Input Text…
圖、文／1111今年農曆春節，台灣鐵路工會因為勞動法一例一休所衍生...

Evaluating…

2018-03-20 06:19:57.679123 W tensorflow/core/platform/cpu feature_uard.cc:45] The TensorFlow library wasn't compiled to
use SSE4.1 instructions, but these are available on your machine and could speed up CPU computations
2018-03-20 06:19:57.679137 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to
 use SSE4.2 instructions, but these are available on your machine and could speed up CPU computations
2018-03-20 06:19:57.679142 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to
 use AVX instructions, but these are available on your machine and could speed up CPU computations
2018-03-20 06:19:57.679159 W tensorflow/core/platform/cpu feature uard.cc:45] The Tensor Flow library wasn't compiled to
 use AVX2 instructions, but these are available on your machine and could speed up CPU computations
2018-03-20 06:19:57.679166 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to
 use FMA instructions, but these are available on your machine and could speed CPU computations

Prediction results...
total number of test examples: 1
category: positive
Accuracy: 0.78942
```

圖 E 模式驗證結果(滿意 2)

```
Parameters:
ALLOW_SOFT_PLACEMENT=True
BATCH_SIZE=64
CHECKPOINT DIR /runs/1520995995/checkpoints/
EVAL_TRAIN=True
LOG_DEVICE_PLACEMENT=False
VERY_NEGATIVE_DATA_FILE=./data/test_text/veryneg.txt
NEGATIVE_DATA_FILE=./data/test_text/neg.txt
NEUTRAL_DATA_FILE=./data/test_text/neu.txt
POSITIVE_DATA_FILE=./data/test_text/pos.txt
VERY_POSITIVE_DATA_FILE=./data/test_text/verypos.txt

Building prefix dict from the default dictionary
Loading model from cache /tmp/jieba.cache
Loading model cost 0.614 seconds
Prefix dict has been built successfully

Input Text…
盼了20年，桃園機場捷運今（2）日起展開試營運，立法院長蘇嘉全在...

Evaluating…

2018-03-20 06:21:03.187291 W tensorflow/core/platform/cpu feature_uard.cc:45] The TensorFlow library wasn't compiled to
use SSE4.1 instructions, but these are available on your machine and could speed up CPU computations
2018-03-20 06:21:03.187303 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to
 use SSE4.2 instructions, but these are available on your machine and could speed up CPU computations
2018-03-20 06:21:03.187315 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to
 use AVX instructions, but these are available on your machine and could speed up CPU computations
2018-03-20 06:21:03.187323 W tensorflow/core/platform/cpu feature uard.cc:45] The Tensor Flow library wasn't compiled to
 use AVX2 instructions, but these are available on your machine and could speed up CPU computations
2018-03-20 06:21:03.187331 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to
 use FMA instructions, but these are available on your machine and could speed CPU computations

Prediction results...
total number of test examples: 1
category: positive
Accuracy: 0.77152
```

圖 F 模式驗證結果(滿意 3)

```
Parameters:
ALLOW_SOFT_PLACEMENT=True
BATCH_SIZE=64
CHECKPOINT DIR /runs/1520995995/checkpoints/
EVAL_TRAIN=True
LOG_DEVICE_PLACEMENT=False
VERY_NEGATIVE_DATA_FILE=./data/test_text/veryneg.txt
NEGATIVE_DATA_FILE=./data/test_text/neg.txt
NEUTRAL_DATA_FILE=./data/test_text/neu.txt
POSITIVE_DATA_FILE=./data/test_text/pos.txt
VERY_POSITIVE_DATA_FILE=./data/test_text/verypos.txt

Building prefix dict from the default dictionary
Loading model from cache /tmp/jieba.cache
Loading model cost 0.614 seconds
Prefix dict has been built successfully

Input Text…
（中央社記者陳葦庭台北8日電）台鐵產業工會在除夕到初三期間發起依法休假...

Evaluating…

2018-03-20 06:22:12.681541 W tensorflow/core/platform/cpu feature_uard.cc:45] The TensorFlow library wasn't compiled to
use SSE4.1 instructions, but these are available on your machine and could speed up CPU computations
2018-03-20 06:22:12.681552 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to
 use SSE4.2 instructions, but these are available on your machine and could speed up CPU computations
2018-03-20 06:22:12.681561 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to
 use AVX instructions, but these are available on your machine and could speed up CPU computations
2018-03-20 06:22:12.681569 W tensorflow/core/platform/cpu feature uard.cc:45] The Tensor Flow library wasn't compiled to
 use AVX2 instructions, but these are available on your machine and could speed up CPU computations
2018-03-20 06:22:12.681574 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to
 use FMA instructions, but these are available on your machine and could speed CPU computations

Prediction results...
total number of test examples: 1
category: positive
Accuracy: 0.76891
```

圖 G 模式驗證結果(滿意 4)

```
Parameters:
ALLOW_SOFT_PLACEMENT=True
BATCH_SIZE=64
CHECKPOINT DIR /runs/1520995995/checkpoints/
EVAL_TRAIN=True
LOG_DEVICE_PLACEMENT=False
VERY_NEGATIVE_DATA_FILE=./data/test_text/veryneg.txt
NEGATIVE_DATA_FILE=./data/test_text/neg.txt
NEUTRAL_DATA_FILE=./data/test_text/neu.txt
POSITIVE_DATA_FILE=./data/test_text/pos.txt
VERY_POSITIVE_DATA_FILE=./data/test_text/verypos.txt

Building prefix dict from the default dictionary
Loading model from cache /tmp/jieba.cache
Loading model cost 0.614 seconds
Prefix dict has been built successfully

Input Text…
以站務員為主的台鐵產業工會在除夕到初三期間發起「依法休假」，導致部分車站...

Evaluating…

2018-03-20 06:23:50.714152 W tensorflow/core/platform/cpu feature_uard.cc:45] The TensorFlow library wasn't compiled to
use SSE4.1 instructions, but these are available on your machine and could speed up CPU computations
2018-03-20 06:23:50.714161 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to
 use SSE4.2 instructions, but these are available on your machine and could speed up CPU computations
2018-03-20 06:23:50.714169 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to
 use AVX instructions, but these are available on your machine and could speed up CPU computations
2018-03-20 06:23:50.714172 W tensorflow/core/platform/cpu feature uard.cc:45] The Tensor Flow library wasn't compiled to
 use AVX2 instructions, but these are available on your machine and could speed up CPU computations
2018-03-20 06:23:50.714181 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to
 use FMA instructions, but these are available on your machine and could speed CPU computations

Prediction results...
total number of test examples: 1
category: positive
Accuracy: 0.77126
```

圖 H 模式驗證結果(滿意 5)

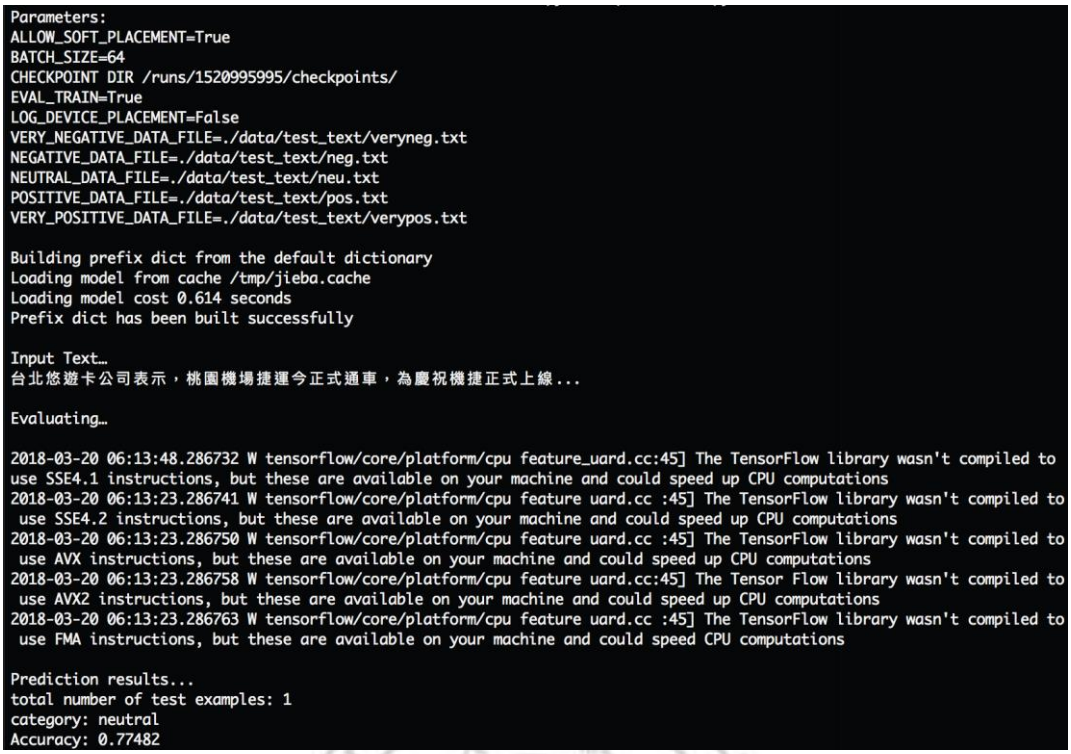

圖 I 模式驗證結果(普通 2)

圖 J 模式驗證結果(普通 3)

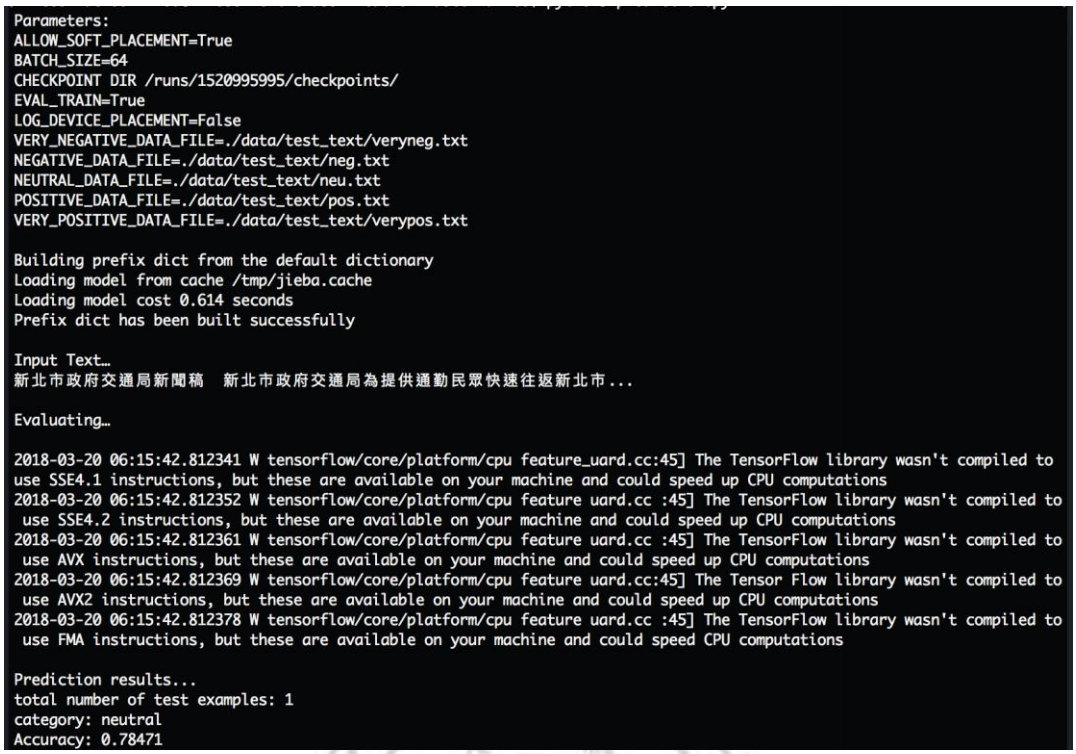

圖 K 模式驗證結果(普通 4)

圖 L 模式驗證結果(普通 5)

```
Parameters:
ALLOW_SOFT_PLACEMENT=True
BATCH_SIZE=64
CHECKPOINT DIR /runs/1520995995/checkpoints/
EVAL_TRAIN=True
LOG_DEVICE_PLACEMENT=False
VERY_NEGATIVE_DATA_FILE=./data/test_text/veryneg.txt
NEGATIVE_DATA_FILE=./data/test_text/neg.txt
NEUTRAL_DATA_FILE=./data/test_text/neu.txt
POSITIVE_DATA_FILE=./data/test_text/pos.txt
VERY_POSITIVE_DATA_FILE=./data/test_text/verypos.txt

Building prefix dict from the default dictionary
Loading model from cache /tmp/jieba.cache
Loading model cost 0.614 seconds
Prefix dict has been built successfully

Input Text…
LTN 養不活捷運？ 賀陳旦：地方政府可拿受惠的錢支應 2017-05-16 23:01 〔記者鄭瑋奇 / 台北報導〕行政院通...

Evaluating…

2018-03-19 11:30:53.734103 W tensorflow/core/platform/cpu feature_uard.cc:45] The TensorFlow library wasn't compiled to use S
SE4.1 instructions, but these are available on your machine and could speed up CPU computations
2018-03-19 11:30:53.734172 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to use
SSE4.2 instructions, but these are available on your machine and could speed up CPU computations
2018-03-19 11:30:53.734189 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to use
AVX instructions, but these are available on your machine and could speed up CPU computations
2018-03-19 11:30:53.734201 W tensorflow/core/platform/cpu feature uard.cc:45] The Tensor Flow library wasn't compiled to use
AVX2 instructions, but these are available on your machine and could speed up CPU computations
2018-03-19 11:30:53.734214 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to use
FMA instructions, but these are available on your machine and could speed CPU computations

Prediction results...
total number of test examples: 1
category: negative
Accuracy: 0.79114
```

圖 M 模式驗證結果(不滿意 2)

```
Parameters:
ALLOW_SOFT_PLACEMENT=True
BATCH_SIZE=64
CHECKPOINT DIR /runs/1520995995/checkpoints/
EVAL_TRAIN=True
LOG_DEVICE_PLACEMENT=False
VERY_NEGATIVE_DATA_FILE=./data/test_text/veryneg.txt
NEGATIVE_DATA_FILE=./data/test_text/neg.txt
NEUTRAL_DATA_FILE=./data/test_text/neu.txt
POSITIVE_DATA_FILE=./data/test_text/pos.txt
VERY_POSITIVE_DATA_FILE=./data/test_text/verypos.txt

Building prefix dict from the default dictionary
Loading model from cache /tmp/jieba.cache
Loading model cost 0.614 seconds
Prefix dict has been built successfully

Input Text…
（更新：新增動新聞）桃園國際機場捷運A8站，今天早上近6時許，一輛聯結車未注意限高...

Evaluating…

2018-03-19 11:33:02.554431 W tensorflow/core/platform/cpu feature_uard.cc:45] The TensorFlow library wasn't compiled to use S
SE4.1 instructions, but these are available on your machine and could speed up CPU computations
2018-03-19 11:33:02.554457 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to use
SSE4.2 instructions, but these are available on your machine and could speed up CPU computations
2018-03-19 11:33:02.554469 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to use
AVX instructions, but these are available on your machine and could speed up CPU computations
2018-03-19 11:33:02.554481 W tensorflow/core/platform/cpu feature uard.cc:45] The Tensor Flow library wasn't compiled to use
AVX2 instructions, but these are available on your machine and could speed up CPU computations
2018-03-19 11:33:02.554490 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to use
FMA instructions, but these are available on your machine and could speed CPU computations

Prediction results...
total number of test examples: 1
category: negative
Accuracy: 0.77901
```

圖 N 模式驗證結果(不滿意 3)

```
Parameters:
ALLOW_SOFT_PLACEMENT=True
BATCH_SIZE=64
CHECKPOINT DIR /runs/1520995995/checkpoints/
EVAL_TRAIN=True
LOG_DEVICE_PLACEMENT=False
VERY_NEGATIVE_DATA_FILE=./data/test_text/veryneg.txt
NEGATIVE_DATA_FILE=./data/test_text/neg.txt
NEUTRAL_DATA_FILE=./data/test_text/neu.txt
POSITIVE_DATA_FILE=./data/test_text/pos.txt
VERY_POSITIVE_DATA_FILE=./data/test_text/verypos.txt

Building prefix dict from the default dictionary
Loading model from cache /tmp/jieba.cache
Loading model cost 0.614 seconds
Prefix dict has been built successfully

Input Text…
新聞主內容 全球禽流感疫情一度爆發，交通部函請大眾運輸「禁止旅客攜帶禽鳥類寵物搭乘」，因此捷運、高鐵都是禁止的...

Evaluating…

2018-03-19 11:34:58.324463 W tensorflow/core/platform/cpu feature_uard.cc:45] The TensorFlow library wasn't compiled to use S
SE4.1 instructions, but these are available on your machine and could speed up CPU computations
2018-03-19 11:34:58.324487 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to use
SSE4.2 instructions, but these are available on your machine and could speed up CPU computations
2018-03-19 11:34:58.324495 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to use
AVX instructions, but these are available on your machine and could speed up CPU computations
2018-03-19 11:34:58.324502 W tensorflow/core/platform/cpu feature uard.cc:45] The Tensor Flow library wasn't compiled to use
AVX2 instructions, but these are available on your machine and could speed up CPU computations
2018-03-19 11:34:58.324517 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to use
FMA instructions, but these are available on your machine and could speed CPU computations

Prediction results...
total number of test examples: 1
category: negative
Accuracy: 0.78002
```

圖 O 模式驗證結果(不滿意 4)

```
Parameters:
ALLOW_SOFT_PLACEMENT=True
BATCH_SIZE=64
CHECKPOINT DIR /runs/1520995995/checkpoints/
EVAL_TRAIN=True
LOG_DEVICE_PLACEMENT=False
VERY_NEGATIVE_DATA_FILE=./data/test_text/veryneg.txt
NEGATIVE_DATA_FILE=./data/test_text/neg.txt
NEUTRAL_DATA_FILE=./data/test_text/neu.txt
POSITIVE_DATA_FILE=./data/test_text/pos.txt
VERY_POSITIVE_DATA_FILE=./data/test_text/verypos.txt

Building prefix dict from the default dictionary
Loading model from cache /tmp/jieba.cache
Loading model cost 0.614 seconds
Prefix dict has been built successfully

Input Text…
看一下地圖就知道為什麼東港不要台鐵東港支線而是要高捷紅線延伸 紅線延伸幾乎是走捷徑，台鐵那還要繞遠路...

Evaluating…

2018-03-19 11:36:42.875621 W tensorflow/core/platform/cpu feature_uard.cc:45] The TensorFlow library wasn't compiled to use S
SE4.1 instructions, but these are available on your machine and could speed up CPU computations
2018-03-19 11:36:42.875640 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to use
SSE4.2 instructions, but these are available on your machine and could speed up CPU computations
2018-03-19 11:36:42.875651 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to use
AVX instructions, but these are available on your machine and could speed up CPU computations
2018-03-19 11:36:42.875658 W tensorflow/core/platform/cpu feature uard.cc:45] The Tensor Flow library wasn't compiled to use
AVX2 instructions, but these are available on your machine and could speed up CPU computations
2018-03-19 11:36:42.875669 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to use
FMA instructions, but these are available on your machine and could speed CPU computations

Prediction results...
total number of test examples: 1
category: negative
Accuracy: 0.78207
```

圖 P 模式驗證結果(不滿意 5)

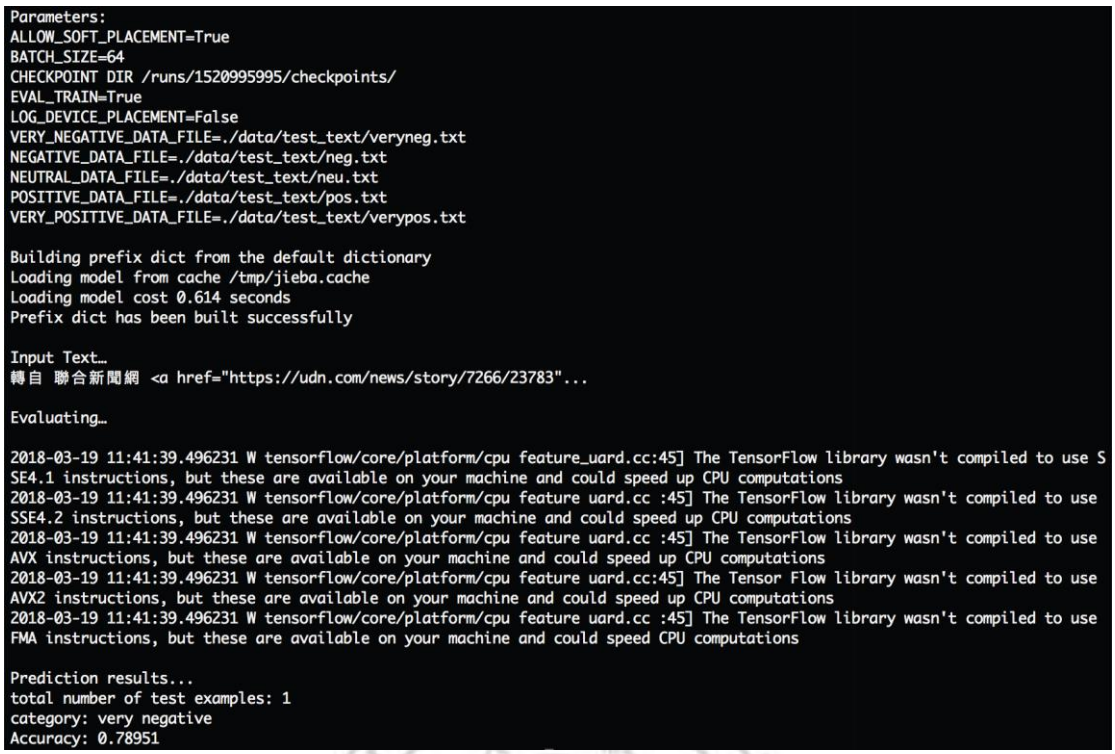

圖 Q 模式驗證結果(非常不滿意 2)

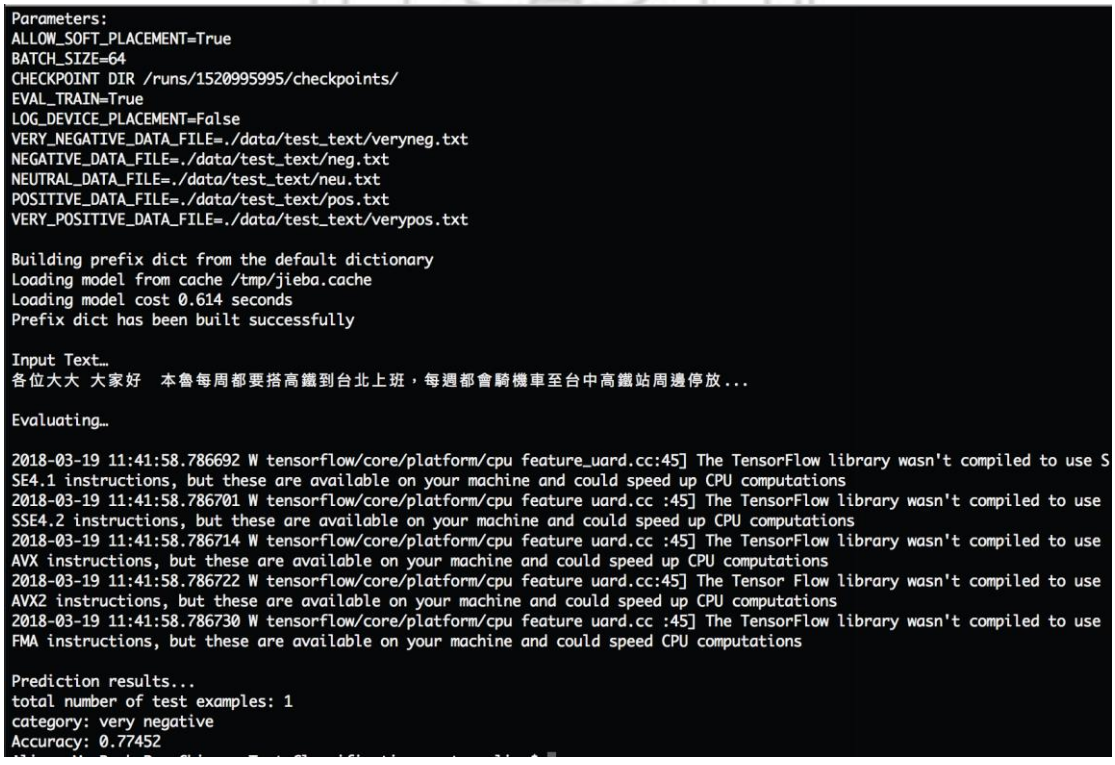

圖 R 模式驗證結果(非常不滿意 3)

```
Parameters:
ALLOW_SOFT_PLACEMENT=True
BATCH_SIZE=64
CHECKPOINT DIR /runs/1520995995/checkpoints/
EVAL_TRAIN=True
LOG_DEVICE_PLACEMENT=False
VERY_NEGATIVE_DATA_FILE=./data/test_text/veryneg.txt
NEGATIVE_DATA_FILE=./data/test_text/neg.txt
NEUTRAL_DATA_FILE=./data/test_text/neu.txt
POSITIVE_DATA_FILE=./data/test_text/pos.txt
VERY_POSITIVE_DATA_FILE=./data/test_text/verypos.txt

Building prefix dict from the default dictionary
Loading model from cache /tmp/jieba.cache
Loading model cost 0.614 seconds
Prefix dict has been built successfully

Input Text…
<!--@IMAGE_3173931_CENTER_0@-->又有遊覽車發生傷亡慘重重大事故...

Evaluating…

2018-03-19 11:42:27.217732 W tensorflow/core/platform/cpu feature_uard.cc:45] The TensorFlow library wasn't compiled to use S
SE4.1 instructions, but these are available on your machine and could speed up CPU computations
2018-03-19 11:42:27.217741 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to use
SSE4.2 instructions, but these are available on your machine and could speed up CPU computations
2018-03-19 11:42:27.217750 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to use
AVX instructions, but these are available on your machine and could speed up CPU computations
2018-03-19 11:42:27.217763 W tensorflow/core/platform/cpu feature uard.cc:45] The Tensor Flow library wasn't compiled to use
AVX2 instructions, but these are available on your machine and could speed up CPU computations
2018-03-19 11:42:27.217771 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to use
FMA instructions, but these are available on your machine and could speed CPU computations

Prediction results...
total number of test examples: 1
category: very negative
Accuracy: 0.79634
```

圖 S 模式驗證結果(非常不滿意 4)

```
Parameters:
ALLOW_SOFT_PLACEMENT=True
BATCH_SIZE=64
CHECKPOINT DIR /runs/1520995995/checkpoints/
EVAL_TRAIN=True
LOG_DEVICE_PLACEMENT=False
VERY_NEGATIVE_DATA_FILE=./data/test_text/veryneg.txt
NEGATIVE_DATA_FILE=./data/test_text/neg.txt
NEUTRAL_DATA_FILE=./data/test_text/neu.txt
POSITIVE_DATA_FILE=./data/test_text/pos.txt
VERY_POSITIVE_DATA_FILE=./data/test_text/verypos.txt

Building prefix dict from the default dictionary
Loading model from cache /tmp/jieba.cache
Loading model cost 0.614 seconds
Prefix dict has been built successfully

Input Text…
補上後續跟回應質疑部分，我想這應該還是可以回文　首先針對擦撞部分...

Evaluating…

2018-03-19 11:42:59.712452 W tensorflow/core/platform/cpu feature_uard.cc:45] The TensorFlow library wasn't compiled to use S
SE4.1 instructions, but these are available on your machine and could speed up CPU computations
2018-03-19 11:42:59.712461 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to use
SSE4.2 instructions, but these are available on your machine and could speed up CPU computations
2018-03-19 11:42:59.712468 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to use
AVX instructions, but these are available on your machine and could speed up CPU computations
2018-03-19 11:42:59.712477 W tensorflow/core/platform/cpu feature uard.cc:45] The Tensor Flow library wasn't compiled to use
AVX2 instructions, but these are available on your machine and could speed up CPU computations
2018-03-19 11:42:59.712482 W tensorflow/core/platform/cpu feature uard.cc :45] The TensorFlow library wasn't compiled to use
FMA instructions, but these are available on your machine and could speed CPU computations

Prediction results...
total number of test examples: 1
category: very negative
Accuracy: 0.78561
```

圖 T 模式驗證結果(非常不滿意 5)