**工業工程與管理系**

**Selenium web crawler**

**and**

**LSTM Sentiment Analysis**

# 摘要

近年來，隨著網路不斷的發展及各種網路交流平台的蓬勃發展下，網路上的留言及評論已成為網路社群中的重要角。留言及評論是網路社群中用戶表達意見的主要途徑。通過留言和評倫，用戶可以分享自己的想法，並與其他用戶交流。可以讓用戶與其他用戶建立聯繫，促進社群的形成。通過互動，用戶可以得到他人的支持和鼓勵，從而建立起信任和友誼關係。也可以為網站、服務或產品提供反饋。通過留言和評倫，用戶可以告訴網站、服務或產品的提供者他們對其的看法，並提出建議和意見。

在本研究中，我們將從google map去尋找店家的評論，並透過selenium抓取店家的留言及星數共計13462筆，將資料進行於處理後，使用LSTM (Long Short-Term Memory)的神經網絡架構來學習並預測評論的情緒是正面或是負面。

# 目錄

[摘要 II](#_Toc134287355)

[目錄 III](#_Toc134287356)

[第一章 緒論 1](#_Toc134287357)

[1.1 研究背景與動機 1](#_Toc134287358)

[1.2 研究目的 1](#_Toc134287359)

[1.3 研究流程 1](#_Toc134287360)

[第二章 文獻探討 2](#_Toc134287361)

[2.1 Selenium 2](#_Toc134287362)

[2.2 jieba 2](#_Toc134287363)

[2.3 神經網路的結構及其學習方式 3](#_Toc134287364)

[2.4 編譯函數(complie) 4](#_Toc134287365)

[2.5 循環神經網路（Recurrent Neural Network, RNN） 6](#_Toc134287366)

[2.6 長短期記憶神經網路(Long Short-Term Memory，LSTM) 7](#_Toc134287367)

[2.7 詞向量與嵌入詞 9](#_Toc134287368)

[第三章 研究方法 10](#_Toc134287369)

[3.1 數據抓取 10](#_Toc134287370)

[3.2 數據整理 10](#_Toc134287371)

[3.3 資料預處理 11](#_Toc134287372)

[3.3.1 Label資料處理 11](#_Toc134287373)

[3.3.2 樣本資料的處理 11](#_Toc134287374)

[3.3.3 建立token字典 12](#_Toc134287375)

[3.3.4 將訓練及測試資料轉換為數字list 12](#_Toc134287376)

[3.4 建立，訓練，預測模型 12](#_Toc134287377)

[第四章 研究結果 14](#_Toc134287378)

[第五章 結論 17](#_Toc134287379)

# 緒論

* 1. **研究背景與動機**

網路上充斥著各式各樣不同的平台，如社群媒體、外送 app、網路論壇等，雖功能及目的各不相同但大多都有留言及評論的功能，不管是用戶或是平台方，再行動前往往都會參考其留言及評論，判斷是否行動亦或是做出改善。因此情緒分析模型可以幫助我們了解留言和評倫中用戶表達的情緒。通過了解用戶的情緒狀態，我們可以更好地回應他們的需求，並提供更好的服務和體驗。

* 1. **研究目的**

在本研究中，抓取了google map上特定類別店家的13000多則評論及星數，使用長短期記憶神經網路(LSTM)對於處理並計算自然語言的優勢，針對文本進行情緒分析方面的學習，並將其預測未知文本的情緒，希望藉此能夠及時對陌生的評論及留言進行篩檢，並能及時針對評論及留言提出的問題進行改善。

* 1. **研究流程**

以下為本專題的研究流程。第一章首先說明研究動機與目的；第二章探討爬蟲使用的Selenium工具、資料處理時所用到的jieba分詞器、神經網路的結構及學習方法、編譯函數(complie)及循環神經網路（RNN）、長短期記憶神經網路(LSTM)之相關文獻；第三章敘述本專題透過爬蟲所獲取的資料、數據整理、資料預處理及模型訓練的實驗設計；第四章分析與評估實驗結果；第五章針對研究結果進行總結。

# 文獻探討

## Selenium

Selenium 是一個用於自動化瀏覽器的框架，可以用於模擬人類對網站進行的操作，例如點擊按鈕、輸入文本等。Selenium 可以通過程式碼控制瀏覽器，並可以支援多種瀏覽器，例如 Chrome、Firefox、Safari 等。因此，Selenium 常用於網頁爬蟲和自動化測試等任務。使用 Selenium 爬蟲的步驟大致如下：

1. 安裝 Selenium 及適用於你使用的瀏覽器的驅動程式。
2. 建立 Selenium 連接，並打開瀏覽器。
3. 使用 Selenium 指令模擬人類對網站的操作，例如點擊按鈕、輸入文本等。
4. 獲取網頁的 HTML 內容，並使用網頁解析工具，如 Beautiful Soup，提取有用的資訊。
5. 關閉瀏覽器，並結束 Selenium 連接。

Selenium 是一個強大的自動化瀏覽器框架，可以用於網頁爬蟲和自動化測試等任務。本次爬取資料就是透過python安裝selenium套件，並運用其操控crome瀏覽器自動化抓取google map中的店家評論及星數。

## jieba

Jieba 是一個用於中文斷詞的 Python 庫。它可以將中文文本分解成詞彙序列，並支援自定義詞典。Jieba 提供了多種斷詞模式，包括精確模式、全模式和搜尋引擎模式等。此外，Jieba 還支援自定義詞典，可以讓用戶自行添加新詞或自定義詞性。使用 Jieba 的斷詞功能的基本步驟如下：

1. 安裝 Jieba 庫。
2. 引入 Jieba 庫。
3. 調用 Jieba 的斷詞函數，將中文文本分解成詞彙序列。
4. 可以選擇性地使用 Jieba 的其他功能，例如自定義詞典、計算詞頻等

Jieba 是一個強大的中文斷詞庫，提供了多種斷詞模式和自定義詞典的功能，可以幫助我們將中文文本分解成詞彙序列。此外，Jieba 還提供了其他功能，例如計算詞頻、對文本進行分類等。它是處理中文文本處理任務的首選工具之一。

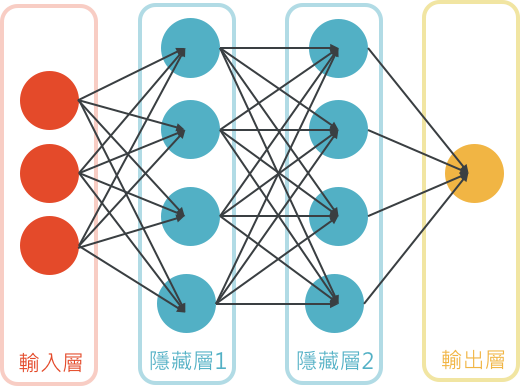


圖2.1 神經網路結構

## 神經網路的結構及其學習方式

如圖2.1所示，由多個神經元(neuron)組成神經層(layer)，再將多個神經層(layer)堆疊並連接起來組成神經網路(neural network)。

* 輸入層(input layer) : 接收輸入資料特徵的層
* 隱藏層(open layer) : 中間未與外界接觸的層
* 輸出層(output layer) : 將結果輸出到外界的層

一個神經網路要多少層、層的種類、神經層的數量以及層與層間的連接方式，都可依使用者的需求進行設計，但各個得數量並非是越多越好，比如可能造成過度配適使其面對新樣本時準確率不如預期。

神經網路的運作方式就是藉由不斷的修正各層神經元之間的權重(weight)來進行學習，藉由調整某個數入權重，即可改變其對整體輸出之影響力。如圖2.2所示，權重又有分為一般的權重(w)及偏值(b)，如果是密集層則上層每一個神經元輸出的訊息都會傳至下層的所有神經元，也就可以計算每個神經層間所有權重參數，如圖2.1中的輸入層到隱藏層1就有 3\*4+4=16 個權重參數。

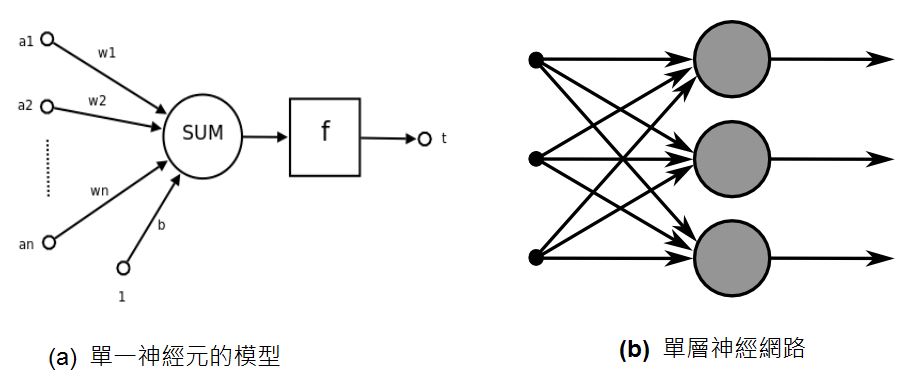
神經元除了會對輸入值做加權及偏值的線性轉換外，另外還具備一些非線性的轉換，以增加對非線性規則的學習能力，這些非線性轉換能力的來源就是啟動函數(Activation functions)，如圖2.2的f。

圖2.2 神經網路的學習方式

啟動函數(Activation functions)有很多種，例如 relu 會將小於1的輸出值都改為零、simgoid 輸出值是介於-1到1之間等等。

## 編譯函數(complie)

神經網路會依輸入資料一層層的運算並向下傳遞，接著輸出預測值(prediction)

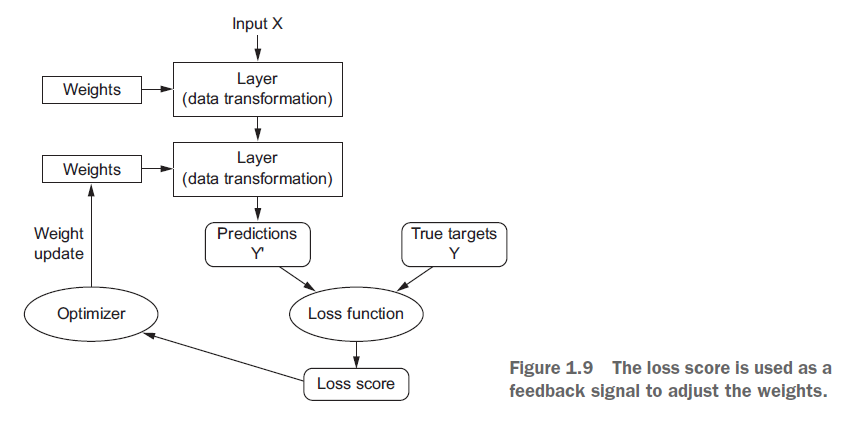
* 損失函數(loss function) : 會計算標準答案與預測值之間的差距以作為損失值(loss)。
* 優化器(optimizer) : 將依據損失值由下而上的修正各層的權重，並將損失值降到最低。
* 評量準則(metrics) : 用來評量模型訓練的成效(準確率)，但並不會參與模型的訓練，但可讓我們直觀的了解成效如何。

圖2.3 神經網路訓練流程

如圖2.3所示，經由大量資料不斷來回重複訓練，損失值將不斷變小(也就是越來越準)，最後就可能達到我們所期望的準確率。如果訓練效果不好，調整損失函數及優化器也是其中一個選項。而以下為根據文獻對損失函數(loss function)、優化器(optimizer)及評量準則(metrics)在選擇上的參考整理。

表2.1 損失函數(loss function)

|  |  |  |
| --- | --- | --- |
| 問題類型 | 輸出層的啟動函數 | 損失函數 |
| 二元分類 | Sigomid | losses.binary­­­­­­­­­­­­\_crossentropy |
| 多元分類 | Softmax | losses.categorical­­­­­­­­­­­­\_crossentropy |
| 多標籤分類 | Sigmoid | losses.binary­­­­­­­­­­­­\_crossentropy |
| 迴歸直分類 | None | Losses.mse |
| 迴歸求0到1值 | Sigmoid | Losses.mse 或 losses.binary­­­­­­­­­­­­\_crossentropy |

優化器(optimizer)

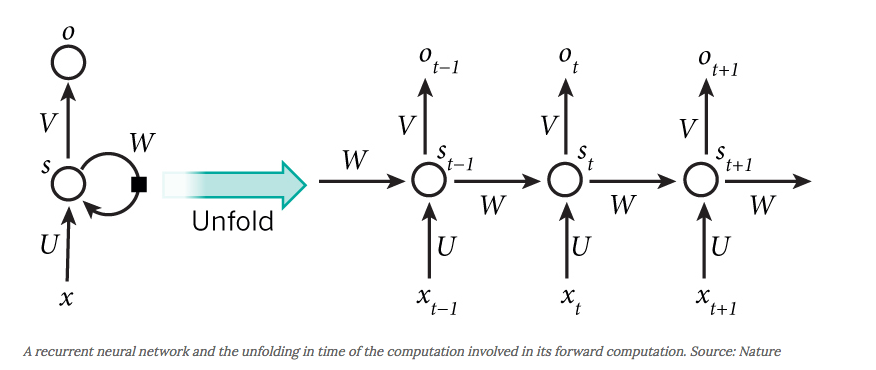
* 如果資料是稀疏的(例如0很多1很少)，建議使用自適應學習速率的優化器，例如RMSprop、Adagrad、Adadelta 或 Adam。
* 對於較多層或較複雜的神經網路，推薦使用自適應學習速率的優化器 RMSprop。
* RMSprop,Adadelta、Adam 在很多情況下效果相近。
* Adam、Nadam 就是在 RMSprop 的基礎上分別加上了 momentum、Nesterov momentum 動量，因此在某些情況下，Adam 或 Nadam可能會有比較好的表現。
* 基本款的 SDG 雖然比較有可能達到梯度的最低點，但參數比較難調整，且會比其他演算法花更多時間訓練。

表2.2 評量準則(metrics)

|  |  |
| --- | --- |
| 模型所使用損失函數 | 對應的 acc 評量準則 |
| losses.binary­­­­­­­­­­­­\_crossentropy或模型只輸出一個數值(二元或迴歸預測) | metrics.binary­­­­­­­­­­­­\_accuracy |
| losses.sparse\_categorical­­­­­­­­­­­­\_crossentropy | metrics.sparse\_categorical­­­­­­­­­­­­\_ accuracy |
| 其他損失函數 | Metrics. categorical­­­­­­­­­­­­\_ accuracy |

## 循環神經網路（Recurrent Neural Network, RNN）

RNN是一種專門用於處理時序數據的神經網路架構，它在處理自然語言時非常有用。在傳統的神經網路中，訓練過程中的輸入和輸出是互相獨立的不會保留任何狀態，也就是說這些神經網路沒有記憶能力，但對於自然語言處理來說，這種方法有一些局限性。因為自然語言句子可能非常長，並且每個詞的含義可能受到前面和後面的詞的影響。

不同於傳統神經網路，RNN是一種擁有記憶的神經網路，能夠累積過去輸出的資料來分析目前的資料，及處理序列資料。RNN擁有三種權重，U是隱藏權重，V是輸出權重，W是時步權重如下圖2.4所示。

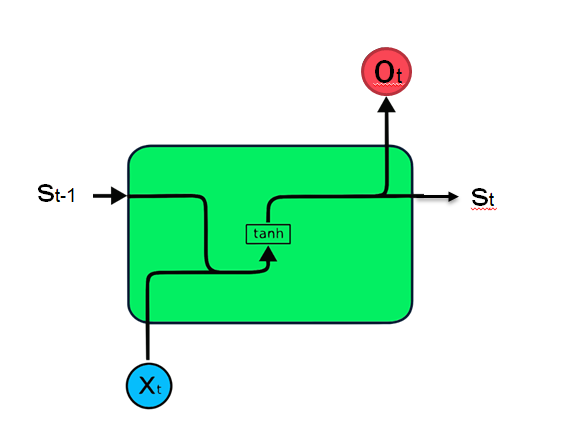
在圖2.4展開的循環神経網路中，下標t就是時歩，每一個時歩都是一個全連接的三層神經網路。假設現在有一個序列資料，在每一個時步t有一個輸入向量xt(輸入層)，神經網路會輸出向量(輸出層），輸入向量除了xt，選需要將上一時歩t-1 的隱藏層輸出st-1合併後才是輸入資料，隱藏層的輸出st, 除了輸出至ot(輸出層），同時還輸出至下一個時歩 t+1 作為輸入，如圖2.5所示。

圖2.5 RNN單元

圖2.4 RNN的隱藏層展開

圖2.5是循環神經網路的組成單元，以此例的隱藏層啟動函數是 Tanl函数，每一個時歩的輸入除了目前時歩t，還有上一個時歩t+1的際藏尾輸出合併輪人來一起預測St，換向話說，循環神經網路的每一個時步t不只有當前資料，還包含之前時步的所有累積資訊（即權重 W)，這也是為什麼循環神經網路擁有記憶能力，這使得RNN非常適合用於自然語言處理任務，例如文本分類、機器翻譯、語音識別等。 RNN能夠保留前面詞的信息，並根據當前輸入的詞對其進行修改，這有助於更好地理解整個句子的含義。

此外，RNN還具有許多其他優點，例如它能夠處理不定長度的序列輸入，並且可以通過將隱藏層的輸出連接到自身來進行多次循環，以便對長期依賴性進行建模。這對於自然語言處理非常有用，因為許多自然語言句子具有長期依賴性，即一個詞的含義可能受到遠處詞的影響。

## 長短期記憶神經網路(Long Short-Term Memory，LSTM)

LSTM改良自循環神經網路(RNN)，是一種擁有長期記憶的神經網路，因為RNN會產生梯度消失(Vanishing Gradient Problem)及梯度爆炸(Exploding Gradient Program)的問題，造成其只能更新最近幾個時步的權重，也就是只有短期記憶，所以實務上不直接使用RNN。

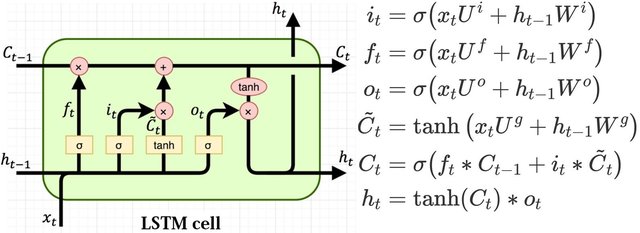
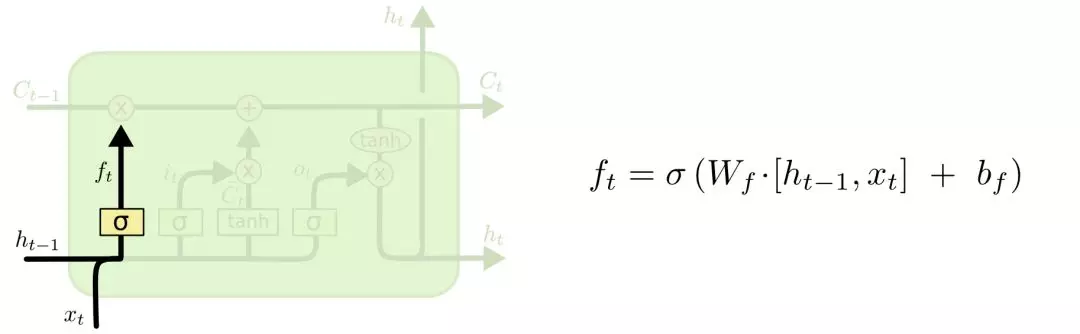
LSTM在結構上與RNN並沒有什麼不同，同樣都有時步迴圈差別在於RNN的隱藏層只有一層神經層，而LSTM的隱藏層則是一個LSTM單元(LSTM cell)，這個單元不只有一個神經層而是四個神經層，如圖2.6所示，最上面那條為長期記憶線也稱單元狀態(cell Satae)，下面四個黃色方框為4個神經層。

圖2.6 LSTM單元

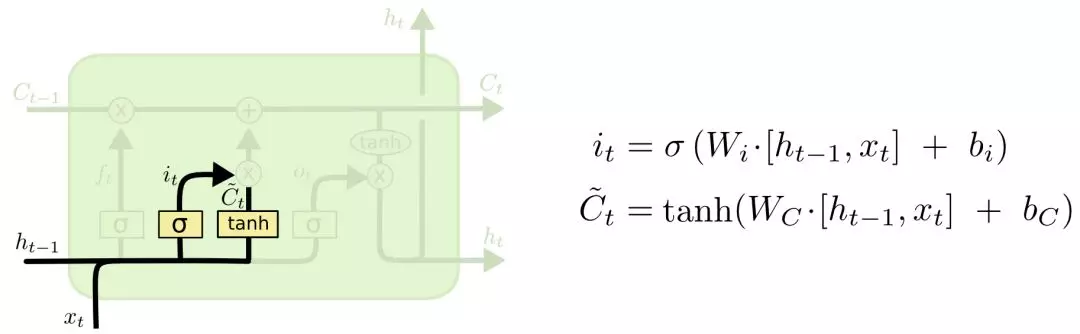
四種神經層依啟動函數分為兩種:

* Sigmiod神經層 : 為了實現閘門，我們需要一組介於 0 和 1 之間的數值，使用 Sigmoid 函數 將值控制在 0~1 之間以控制資料通過比例，然後與資料進行相乘。
* Tanh 神經層 : Tanh函數的處理，確保了數值介於 1 和 -1 之間，並透過此神經層取得欲通過的候選資料資料。

LSTM的記憶力就是依靠閘門來篩選資料，自動學習資料的保留即刪除，能夠保留面對現在問題所需的長期資料。而構成LSTM並讓其建立長期記憶力有三個，分別是輸入閘、遺忘閘及輸出閘。

* 遺忘閘(Forget Gate) : 將用不到的資料從長期記憶線中刪除，保留需要的資料。

(2.1)

* 輸入閘(Input Gate) : 用於更新長期記憶線的資料，將資料進行新增或替換。

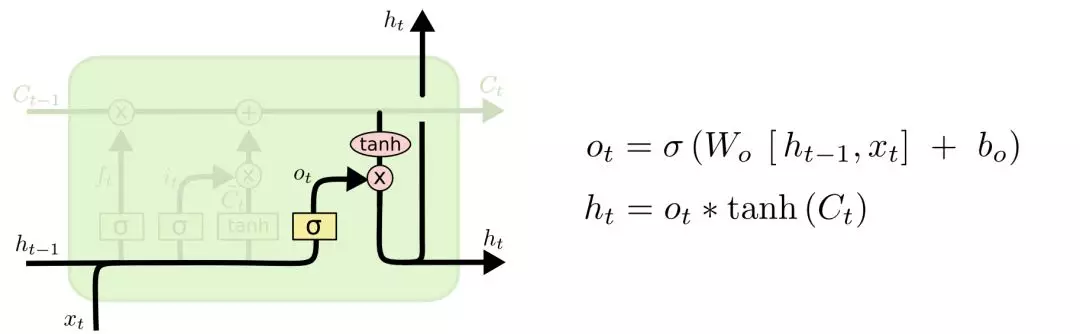
(2.3)

(2.2)

圖2.8 遺忘閘(Forget Gate)

* 輸出閘(Output Gate) : 從長期記憶線中取得並篩選出之後需要用到的資料，從輸出層輸出及輸出至下一個時步。

圖2.9 輸入閘(Input Gate)



(2.5)

(2.4)

圖2.9 輸出閘(Output Gate)

## 詞向量與嵌入詞

詞向量（Word Vector）或稱為詞嵌人(Word Embedding）也是一種文字資料向量化的方法，可以將單字嵌入一個幅點數的數學空間中。假設：現在有10.000 個不同單字（詞庫），分别使用 One-hot 編碼和詞向量（使用 200 個神經元的隱藏層）執行 文字資料向量化的差異，如下所示：

* One-hot 編碼需要使用程式碼轉換單字成為向量；詞向量是建立神經網路來自行學習單字的詞向量。
* One-hot 編碼建立的是一個高維度的稀疏矩陣（每一個向量長 10,000，其中只有一個1、其他都是 0），以此例是 10,000× 10,000，即 10,000 個單字，每一個單字是長度 10,000 的向量；詞向量是低維度浮點數的緊密矩陣，因為隱藏層是 200 個神經元，可以壓縮成 10,000×200，10,000 個軍字，每一個單字是長度 200 的向量。

詞向量就是將原來詞庫（Lexicon）中每一個單字的高維度 One-hot 編碼的向量（10,000），轉换成低維度浮點數向量（200），不只如此，因為詞向量是透過神經網路白行學習來建立，還可以自動建立單字之間上下文關係，即單字的意義。

使用一個向量來表示每一個詞(vector representation)，如此一來，就能把一段由許多詞組成的文句，轉換成一個個詞向量來表示，並把這樣數值化的資料，送到模型裡做後續的應用。一組好的詞向量，會使意思相似的詞在向量空間上比較靠近彼此，甚至詞義上的關聯可以用詞向量在空間中的關係來表示。

Keras 提供 Embedding層可以幫助我們在將文字內容轉換成整數索引值後轉換成固定尺寸的層向量，而 Embedding層一定是 Sequential 模型的第一層。

# 研究方法

1. **數據抓取**

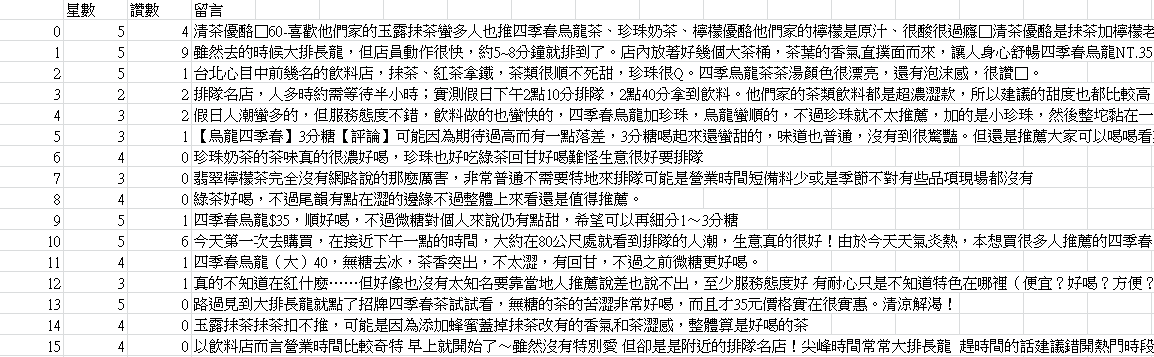
我們使用 Selenium 的 Python 包中的 Webdriver 函數從 Google 地圖共抓取了 13462 條評論。在我們製作的爬蟲工具中，可以輸入想要搜尋的關鍵詞及抓取店家數量，並且可以自動將沒有留言的評論從資料中整筆刪除。我們依序搜尋台北、新北及高雄的飲料店，並分別向各個類別抓取了個評論的星數、讚數及留言，而後將三個類別的資料合併成一個 Excel 檔，如圖3.1所示。

圖3.1 抓取星數、讚數、評論的資料

1. **數據整理**

而後我們將數據資料只保留留言做為樣本數據、星數作為標籤數據。我們將1星、2星及3星替換成0作為負面情緒留言，4星及5星替換成1作為正面情緒留言。再來由於有部分外國留言及符號，因此將留言去除中文以外的文字這樣做是為了更好地分析中文文本的情緒，因為中文文本的詞彙和語法結構比較統一，而非中文文本則可能存在詞彙和語法結構的差異，而後再將因此空白的留言整筆資料進行移除。

並且我們發現資料的正面有10092筆、負面資料有3331筆。由於兩者比例過於懸殊，為了避免模型˙可能會對某一種情緒偏好，導致分析的結果不夠准確，因此我們決定隨機取正面資料5000筆，及負面資料3000並將兩者合併共計800筆作為這次 LSTM 模型所需的資料。

1. **資料預處理**
2. **Label資料處理**

為了要符合訓練模型的格式，我們需要將label資料序列化，轉為[0 1]代表1正面評論，[1 0]代表0負面評論。

1. **樣本資料的處理**

為了提供訓練後用來測試的資料，因此我們將自己將資料切分，8成的資料(6400)為訓練資料，2成的資料(1600)為測試資料。

而由於中文不像英文一個一個單字都是分開的，因此要使用一些工具，來協助斷詞，這裡我們選擇使用結巴(jieba)，我們用其將訓練及測試資料的句子進行分詞，如圖3.2所示。

圖3.2 用jieba將中文留言分詞

難喝服務差欠錢一顆星都嫌多

['難', '喝', '服務', '差', '欠錢', '一', '顆', '星', '都', '嫌多']

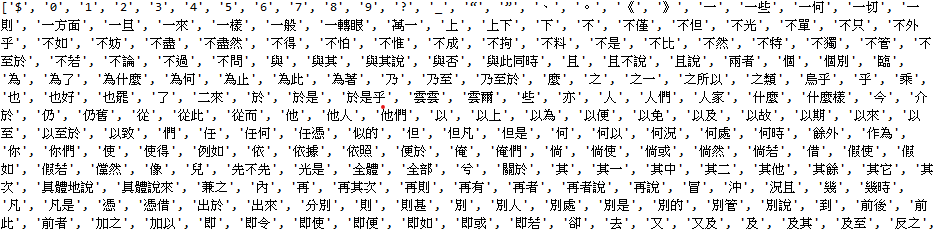
我們在網路上下載停用詞後將其轉為繁體中文，取得對於訓練資料無意義的用詞與符號，存成txt檔，接著讀取文字檔內容，並將各個停用詞以斷行符號\n分割，設定其為stopWords，部分如圖3.3所示，用其將一些無意義的詞及符號等進行移除。

圖3.3 部分停用詞展示

1. **建立token字典**

我們使用使用Tokenizer建立大小為1000的字典，接著透過fit\_on\_texts()方法將訓練的留言資料中，依照文字出現次數排序，而前1000個常出現的單字將會列入token字典中。

1. **將訓練及測試資料轉換為數字list**

接著，透過token的texts\_to\_sequences()方法將訓練及測試資料轉換為數字list，如圖3.4所示。

圖3.4 將文字轉為數字list

也許是高雄飲品種類最豐富的飲料店

[209, 51, 550, 694, 2, 40]

此外，由於keras只接受長度一樣的list輸入，因此必須使用sequence的pad\_sequences()方法，將序列後的訓練及測試資料長度限制在100，表示當list長度超過100時，會自動切斷多出來的內容，反之list長度小於100時便會自動補0，直到長度為100，如圖3.5所示。

圖3.5 將資料填滿或裁減至長度100

[ 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 595 10 5 8 295 45 50 903 681 6 13 6 3

26 130 866 6 773 7 460 753 27 10 439 173 42 595 361 24 104 211

10 280 611 3 6 20 6 626 1 339]

1. **建立，訓練，預測模型**

接著開始建立模型，而過程就是不斷嘗試是不同參數並修正，找出找出可以得得最高 val\_accuracy 的設定，而我們的神經層堆疊如下。

* 第一層 Embedding 層，並測試 output\_dim 輸出維度為多少，而input\_dim 輸入維度則是與前面設定的字典大小相同為1000，input\_length也與前面設定序列長度相同100。
* 第二層 dropout 層，Dropout 比例越大，則模型中被捨棄的神經元越多，對過擬合的抑制越強。但是，如果 Dropout 比例過大，則模型的表現可能會過於保守，無法很好地擬合數據集，因此Dropout 比例也是我們的測試對象。
* 第三層加入隱藏層，我們會測試並設定其神經元數量，其中激活函數設定為relu，表示資料會捨去負數，並介於0到無限大區間。
* 第四層再加一個 dropout 層，並測試其 Dropout 比例，這裡比例與第三層 dropout 層的比例相同來測試。
* 加入輸出層，並設定輸出為2個神經元，並定義激活函數為sigmoid表示資料為0或1。

在我們實驗完模型各個參數的準確率後我們將會接著測試模型的batch\_size 及週期，嘗試模型是否能有更高的準確度及穩定性。最後在使用測試資料進行預測測試。

# 研究結果

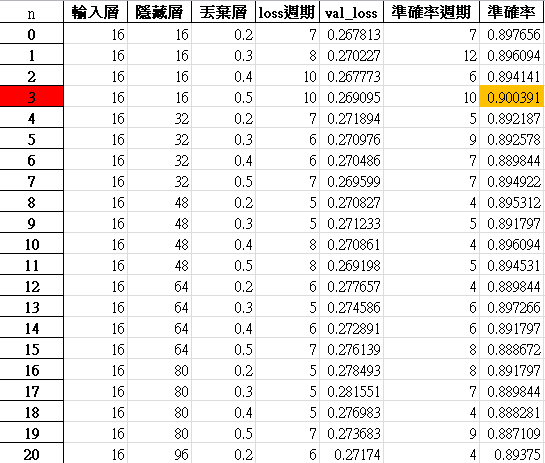
我們透過測試Embedding 層、隱藏層及 dropout 層的組合，試圖尋找合適的模型參數。Embedding 層及隱藏層測試神經元數有16、32、48、64、80、96、112、128，dropout 層測試比例有0.2、0.3、0.4、0.5，batch\_size統一使用100，共計256種組合，部分如圖4.1所示。

圖4.1 實驗組合及數據結果

我們從中選擇了在測試中擁有最高 val\_accuracy 的4個組合，依測試順序 n 為第3、131、203、235號組合，並進行進一步測試，如圖4.2所示。

圖4.2 進一步測試的數據

我們隨後又將這4個組合進行多次測試及繪圖，週期10、batch\_size : 100可以發現神經元數過多反而會造成過度配適況發生，由左到右分別為131、203及235的 accuracy 及val\_accuracy 所繪製的圖，如圖4.3所示。

最終選擇較為穩定的3號模型，下圖4.4為 accuracy 及 val\_accuracy圖表，圖4.5為一到十週期的 val\_loss 及 val\_accuracy 數值。



圖4.3 n:131、n:203、n:235 acc及val\_acc曲線圖

接下開始測試 batch\_size 及最佳的週期數，batch\_size 我們測試了80、90、100、110、120、130、140、150，batch\_size 在140中取得了較好的成績，val\_accuracy來到了0.9，也決定將週期訂為10，如圖4.6所示。

圖4.5 n:3 10週期的val\_loss及val\_acc

圖4.4 n:3 acc及val\_acc取線圖

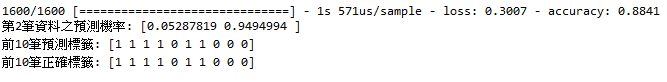
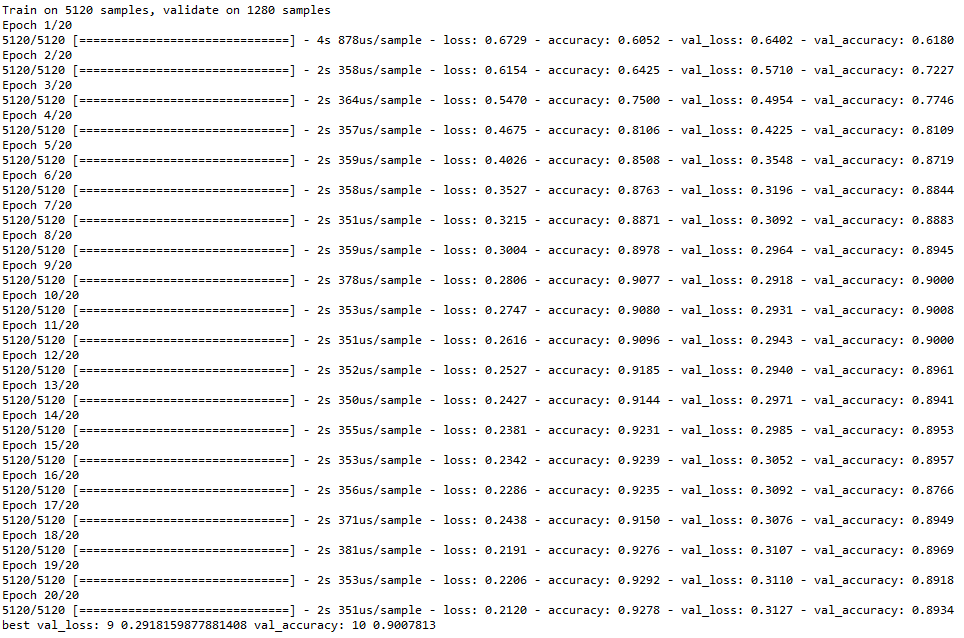
最後我們將訓練的模型進行處存並用來預測測試資料的準確率，並將前十筆的正確答案與預測出的結果進行比對，看看遇到陌生資料時的情緒判斷能力，可以看到預測準確度也有0.884，而前十筆資料預測也是全對的。如圖4.7所示。

圖4.6 batch\_size:140測試結果

我們也有試著運用較有趣的文字雲方式查看較常出現的詞彙。如圖4.8所示。

圖4.8 文字雲查看常用字

圖4.7 最終訓練模型預測準確度



# 結論

LSTM (長短期記憶網絡) 是一種人工神經網絡，具有記憶能力，可以用於處理和預測序列數據。它通常用於自然語言處理和文本分類任務，包括情緒分析。我們做完本次研究後，可以從結果中看到它對未知留言情緒分析的準確率來到了88%以上，並且通過不同測試不同神經原及參數組合也可以看到並非越多神經元就越， batch\_size 的設置也與模型訓練的準確度及穩定度有著相互間的取捨，經過不斷嘗試及觀察才能找到相對較佳的模型準確率。

然而，我們也清楚的理解對於情緒分析這項任務我們還有很多能夠精進的空間且尚未嘗試的方法，例如在文獻中使用的 TF-IDF 進行文本像量化並且使用隨機森林 (Random forest) 的機械學習模型，或者使用word2vec, seq2seq, attention, transformer 等這些優秀的深度學習模型。

而在未來我們也期望能持續精進這些技術，開發出情緒分析的更多用途如市場調查及即時留言審查等，並嘗試更多不同的相關應用，如搜尋引擎、問答及聊天機器人、翻譯及創作類的文本生成等，都是現在正在發展並有望更加精進的技術。