國立臺北大學電機工程學系專題報告

基於 Word2Vec 的情緒分析規則探勘

Data mining of sentiment based on word2vec.

組員：黃子育

指導老師：吳柏翰 老師 林家洨 老師

執行期間：104 年 7 月至 105 年 6 月

1. 摘要

現代科技日新月異,網路也日漸蓬勃,也就在網路上產生大量的資訊。本專題目標是利用網路上的眾多評論，分析出大眾普遍在網路上表達的意見，對於產品、決策的正負面評價，以利於日後在政策、決策上的決定。

關鍵字：word2vec、情緒分析、資料探勘、Jieba

2. 簡介

現在科技日新月異，網路也日漸蓬勃,隨之而生的大量資訊就可以利用來分析大眾的意見,以利於公司對於產品的分析,更有利於日後公司在決策上的決定。

消費者在購買產品後,心中會因為產品的優缺點而對產品進行評價,而企業為了要獲取更大的利潤,通常要透過收集這些評價資訊,進而使商品可以做進一步的改善。近來,網路科技發達,使得消費者習慣更容易從各種不同管道抒發自己對於產品的看法,所以本篇研究,將以文字探勘與情緒分析的做法,建構一套情緒分析系統,將網路上論壇的大量評論進行分析,並以知名車輛品牌為例,進行品牌、正負面情緒等等的分析。

3. 專題進行方式

本專題主要是利用 Jieba、word2vec、mobile01 的十萬則評論、台大開放的情緒字典等等資源，進行情緒分析。

首先,我們利用相關套件,將 mobile01 上的十萬則評論從網路上爬下來，之後以 jieba 斷詞將爬下來的評論進行斷詞,去除標點符號等等相關與情緒較無關的字眼，再將評論放入 word2vec 把詞彙轉換為一個個的詞向量，以利計算，最後將相關字詞丟入並觀察結果，看吐出的字詞的相關性，並與情緒字典做出的語言模型結果做比較，找出兩者的特性，找出各自的優缺點。

此專題為一人獨立完成並會與吳柏翰老師與實驗室同學們討論，並在每周的 meeting 與老師討論進度,完成整個情緒分析的系統。

4. 主要成果與評估

Jieba 分詞：

Jieba 分詞是一個中文分詞程式,他是開放程式碼的程式，根據引用的字典，支持檢體與繁體的分詞，因為詞為最小且有意義的單位，因此斷詞可以說是整個自然語言處理最基礎的工作，所以這是我們的第一個工作。而 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。對於辨識新詞則使用了 HMM 模型（Hidden Markov Model）及 Viterbi 算法來辨識出來。

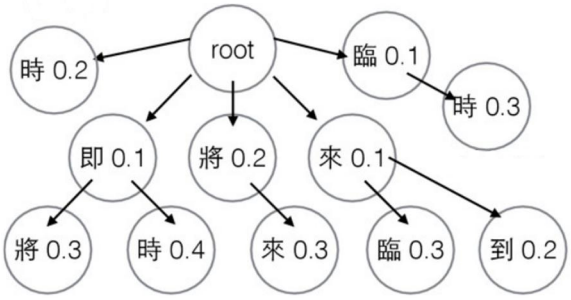


圖 4-1 Trie 樹

這個是我們引入字典後第一個做的事情，我們將字典裡所有的詞進行統計，依據詞頻將每個字接續的字的機率統計出來，畫出我們的 Trie 樹。也就是”將”後面接”來”的機率為 0.3，”來”後面接”到”的機率為 0.2，統計完整顆樹後我們進行下一個步驟。

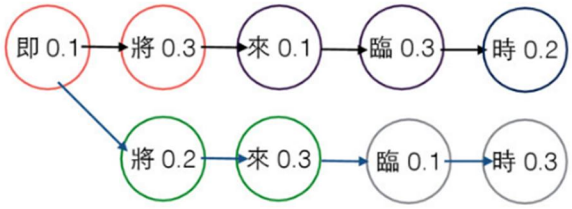


圖 4-2 有向無環圖

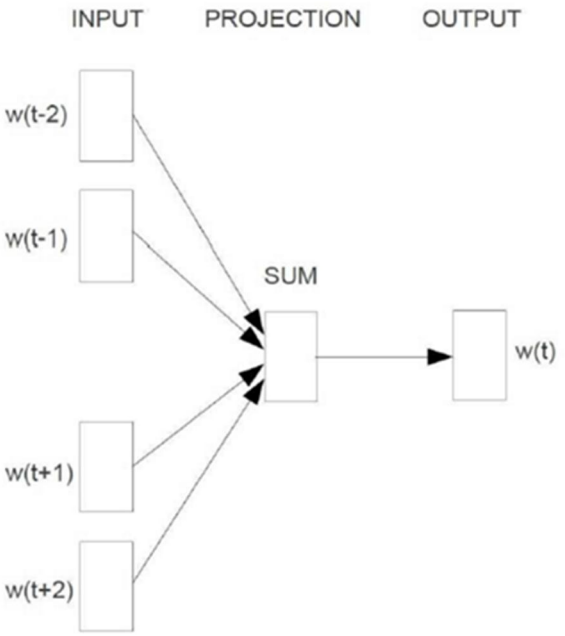
這是我們再分詞上的第二個步驟，這個步驟相我們所輸入的詞彙，依據所有的可能性進行斷詞，本次的例子為”即將來臨時”，依據其可能性，我們可以斷成”即將”、”來臨”、”時”，或者”即”、”將來”、”臨時”，這兩種可能性，再依據這些可能，找查剛剛所建立的 Trie 樹，將 Trie 樹上面所對應的機率，帶入有向無環圖中，將所有機率連乘，圖中的結果就是兩個分支，所以得到兩個機率的結果，我們取機率高的結果為最後的結果，也就是所有分詞可能中，最有可能為正確語意意義的分詞結果。

Word2vec：

接下來我們利用 word2vec 將剛剛斷好的詞轉為詞向量，將詞彙數值化，以方便計算。

Word2vec 是一款能將詞轉換為實數值向量的高效工具，為一群用來產生詞向量的模型。輸入為斷開的詞語，且輸入的順序是不重要的。這些模型為淺層和雙層神經網路，用來訓練以重新建構語言學之詞文本，網路以詞表現，並且需猜測相鄰位置的輸入詞，輸出的詞向量可用於做自然語言處理的相關工作，像是聚類、找同義詞、詞性分析。輸出的向量有線性關係，以下是 word2vec 上的一個經典例子，在訓練後的詞向量中我們可以發現 vector('Paris')-vector('France')+vector('Italy')≈vector('Rome')。有這樣特殊的線性關係。

Word2vec 主要使用到 continuous bag-of-words 模型以及 continuous skip-gram 模型。而詞向量與語言模型是同時間完成的。



CBOW 模型

圖 4-3 CBOW 模型

這個模型主要分為三層，輸入層、投影層和輸出層。輸入層輸入單字向量，在 CBOW 模型中，是在已知當前詞 Wt 的上下文 Wt-2、Wt-1、Wt+1、Wt+2 的條件下，預測當前詞 Wt 。投影層為輸入向量的加總。輸出層對應到一顆 Huffman 樹，他是以語料中出現過的詞當葉子節點，以各詞在語料中出現的次數當權值所構成的 Huffman 樹。

Huffman 樹：

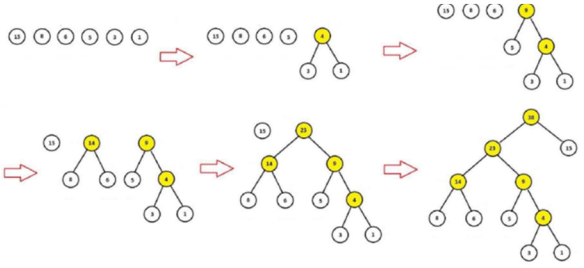


圖 4-4 Huffman 樹

赫夫曼編碼，是一種用於無損資料壓縮的熵編碼演算法。通過評估符號出現機率的方法得到編碼，出現機率高的字母使用較短的編碼，反之出現機率低的則使用較長的編碼，這使編碼之後的字串的平均長度、期望值降低，從而達到無失真壓縮資料的目的。

圖 4-5 Huffman 樹-1

首先我們先將所有的字詞進行詞頻統計

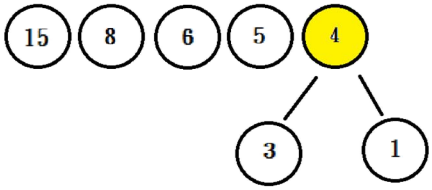


圖 4-6 Huffman 樹-2

再來，我們取最小的兩個節點將加得到新的節點。

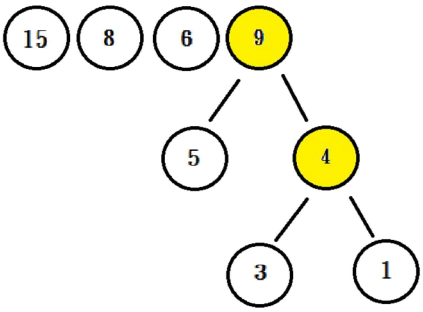


圖 4-7 Huffman 樹-3

連續重複上述步驟直到剩下最後一個節點。

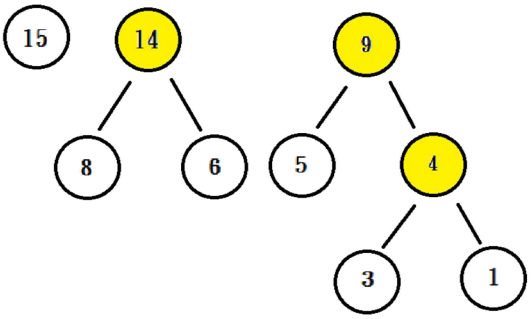


圖 4-7 Huffman 樹-3

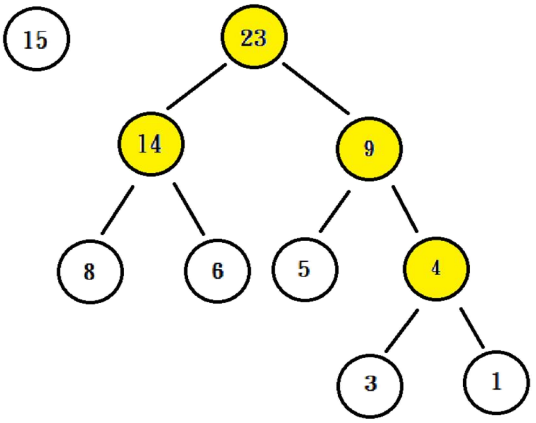


圖 4-8 Huffman 樹-4

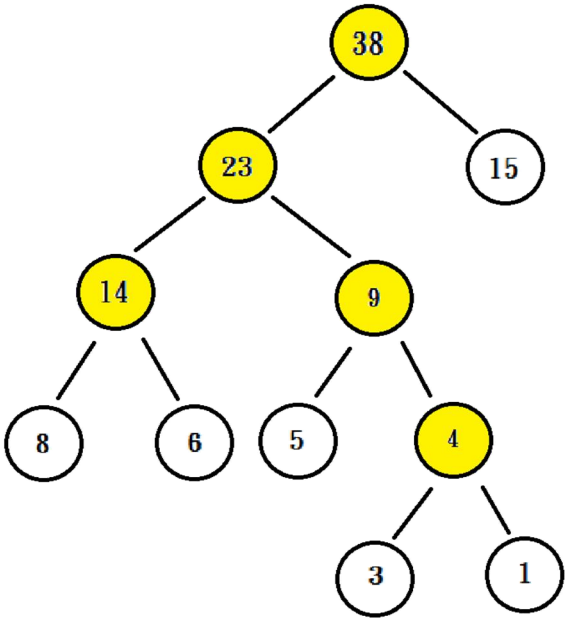


圖 4-9 Huffman 樹-5

$$\sum_{w \in \mathcal{C}} \log p(w|Context(w)),$$

圖 4-10 目標函數

CBOW 的最終目的就是讓這個目標函數取得最大值。

$$p(w|Context(w)) = \prod_{j=2}^{l^w} p(d_j^w|\mathbf{x}_w),$$

圖 4-11 函數展開內容

而其中的條件機率 P 展開為上述式子。

$$\sigma(\mathbf{x}_w^\top) = \frac{1}{1+e^{-\mathbf{x}_w^\top}},$$

圖 4-12 sigmoid 函數

根據 Huffman 樹的結果，我們可以分為正負兩個分支，以左分支為正，右分支為負面情緒，或整相反，可以自由定義。而上面為正面情緒所取得的機率。

$$1-\sigma(\mathbf{x}_w^\top),$$

圖 4-13 sigmoid 函數-1

負面情緒的機率也就是以 1 去相減。

Sigmmoid 函數是拿來用在二元分類上常用的函數，選擇適當的閥值，就可以將函樹所產生的數值，進行適當的二元分類。

$$\text{第 1 次:} \quad p(d_2^w|\mathbf{x}_w) = 1-\sigma(\mathbf{x}_w^\top);$$

$$\text{第 2 次:} \quad p(d_3^w|\mathbf{x}_w) = \sigma(\mathbf{x}_w^\top);$$

$$\text{第 3 次:} \quad p(d_4^w|\mathbf{x}_w) = \sigma(\mathbf{x}_w^\top);$$

$$\text{第 4 次:} \quad p(d_5^w|\mathbf{x}_w) = 1-\sigma(\mathbf{x}_w^\top),$$

圖 4-13 sigmoid 函數-2

經過一連串的正負的機率相乘，就可以得到我們的目標函數，而選擇正面或負面，取決於 Huffman 上面的左右分支。

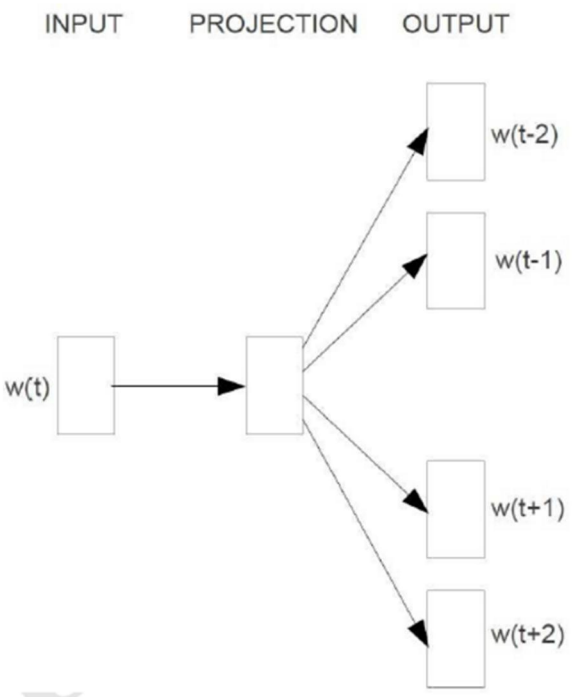


圖 4-14 skip gram 函數

Skip gram 的概念與 CBOW 基本上是一樣的，差異點在 Skip gram 是已知已知當前詞 Wt 的條件下，預測當前詞 Wt 的上下文 Wt-2、Wt-1、Wt+1、Wt+2。所以模型長相跟 CBOW 相反。

專題實際操作：

```
請問SUV第三排的舒適度及空間如何?
為什台灣擁有40年經驗的車廠，確輸給不到10年的Tesla?
Luxgen7 suv豪華型和尊爵型各有哪些配備?
Luxgen7 suv豪華型和尊爵型各有哪些配備?
Luxgen7 suv豪華型和尊爵型各有哪些配備?
比原廠更靚的SUV運動版大包開箱
比原廠更靚的SUV運動版大包開箱
Luxgen7 suv豪華型和尊爵型各有哪些配備?
為什台灣擁有40年經驗的車廠，確輸給不到10年的Tesla?
LUXGEN 5 RF-Design Sport Styling Package 空力套件版
luxgen 5 sedan 新年繼續改
luxgen 5 sedan 新年繼續改
luxgen 5 sedan 新年繼續改
luxgen 5 sedan 新年繼續改
luxgen 5 sedan 新年繼續改
```

圖 4-15 mobile01 raw data

我們以 mobile 上的十萬則評論和情緒字典當作輸入進行實驗，比較何者訓練出來的結果有較佳的表現。

```
請問 SUV 第三排 的 舒適度 及 空間 如何
為 什台灣 擁有 40 年 經驗 的 車廠 ， 確 輸給 不到 10 年 的 Tesla
Luxgen7 suv 豪華型 和 尊爵型 各有 哪些 配備
Luxgen7 suv 豪華型 和 尊爵型 各有 哪些 配備
Luxgen7 suv 豪華型 和 尊爵型 各有 哪些 配備
比 原廠 更靚 的 SUV 運動版 大包 開箱
比 原廠 更靚 的 SUV 運動版 大包 開箱
Luxgen7 suv 豪華型 和 尊爵型 各有 哪些 配備
為 什台灣 擁有 40 年 經驗 的 車廠 ， 確 輸給 不到 10 年 的 Tesla
LUXGEN 5 RF - Design Sport Styling Package 空力 套件 版
luxgen 5 sedan 新年 繼續 改
luxgen 5 sedan 新年 繼續 改
luxgen 5 sedan 新年 繼續 改
luxgen 5 sedan 新年 繼續 改
luxgen 5 sedan 新年 繼續 改
luxgen 5 sedan 新年 繼續 改
```

圖 4-16 jieba 斷詞後結果

運行 jieba 的程式，幫助我們斷詞。

0

```
INFO: PROGRESS: at 3.32% examples, 75793 words/s, in_qs
INFO: PROGRESS: at 6.37% examples, 76183 words/s, in_qs
INFO: PROGRESS: at 9.84% examples, 74121 words/s, in_qs
INFO: PROGRESS: at 12.74% examples, 69242 words/s, in_q
INFO: PROGRESS: at 15.97% examples, 69716 words/s, in_q
INFO: PROGRESS: at 19.71% examples, 71789 words/s, in_q
INFO: PROGRESS: at 23.79% examples, 74526 words/s, in_q
INFO: PROGRESS: at 28.08% examples, 77781 words/s, in_q
INFO: PROGRESS: at 32.90% examples, 80504 words/s, in_q
INFO: PROGRESS: at 37.61% examples, 82968 words/s, in_q
INFO: PROGRESS: at 42.49% examples, 84927 words/s, in_q
```

圖 4-17 word2vec 轉換過程

將斷好詞的文本當作輸入，丟入 word2vec 中幫我們把文字轉為向量。

```
>>> result=model.most_similar(u"保固")
>>> for e in result:
...  print e[0], e[1]
...
快過 0.924601256847
零件 0.733630597591
不會 0.691097795963
單單 0.689484536648
不二價 0.688739895821
趕快 0.686941325665
很爛 0.68683642149
只是 0.683317780495
而是 0.680120646954
說好 0.675670862198
```

圖 4-18 結果

訓練完畢後，我們可以開始提問相關字詞，找查跟這個詞最相關的詞是什麼。以下先以評論為例。

```
>>> result=model.most_similar(u"車展")
>>> for e in result:
...  print e[0], e[1]
...
增 0.854473829269
車廠 0.816890001297
第四 0.807511508465
上海 0.805728435516
節目 0.804571390152
公佈 0.800516963005
新聞 0.785292208195
程式 0.784702301025
95.9 0.775212407112
本屆 0.762710690498
```

圖 4-19 結果-1

```
>>> result=model.most_similar("luxgen")
>>> for e in result:
...  print e[0], e[1]
...
引以 0.651146233082
為傲 0.649157345295
納智傑 0.640007615089
你 0.632611572742
你會 0.622309207916
說聲 0.61246830225
爛 0.597100496292
保固 0.592600941658
降 0.590692937374
小小 0.590583205223
```

圖 4-20 結果-2

我們可以發現，以評論訓練出來的結果相當的好，相關性很高，連英文跟中文

的相關辭彙都可以找出來。

接著我們看字典訓練的結果

```
>>> result=model.most_similar(u"英俊")
>>> for e in result:
...  print e[0], e[1]
...
喧嚷的 0.191532313824
使碎裂 0.184567078948
刻薄的 0.179777443409
抒發 0.179080232978
打耳光 0.170517712831
不可愛 0.166952252388
最高 0.162761792541
失效 0.156733453274
喋喋不休地講話 0.15637640655
豪華的 0.156299352646
```

圖 4-21 結果-3

```
>>> result=model.most_similar(u"酒醉")
>>> for e in result:
...  print e[0], e[1]
...
苛刻 0.186927571893
基礎薄弱 0.174567013979
蠢話 0.171918794513
有得瞧 0.170480147004
逆境 0.166295751929
無法預料 0.165853306651
未完全發展的 0.165104150772
興沖沖 0.159420341253
劈啪 0.159055486321
暗中 0.158660337329
```

圖 4-22 結果-4

```
>>> result=model.most_similar(u"消遣")
>>> for e in result:
...  print e[0], e[1]
...
不忠實的 0.197165384889
引以為榮 0.181686624885
餓死 0.175708919764
有膽量的 0.175355672836
粗糙 0.174215525389
欣賞 0.172500491142
浪費的 0.167452245951
無語 0.166208744049
甘願 0.161510720849
勝任的 0.160968005657
```

圖 4-23 結果-5

```
>>> result=model.most_similar(u"侵略者")
>>> for e in result:
...  print e[0], e[1]
...
幹醮 0.207492768764
屌屁 0.190478697419
否認 0.177099272609
做作 0.17352090776
感激的 0.168456986547
流行 0.164877578616
加罪於 0.163023918867
值得嚮往的 0.160962358117
淺的 0.160062640905
克制 0.159669592977
```

圖 4-24 結果-6

我們可以發現,效果相對於評論而言,相關性低了很多,結果不是很理想,但他卻訓練出了原本字典裏面不應該出現的詞彙,也就是自主產生了新的詞彙,這對我們在情緒分析上,有重大的效果。

5. 銘謝

在這一年中,感謝所有老師與同學。在這一年中,因為專題製作與系上學習的知識差異較大,讓我對資訊領域的相關知識有了更進一步的了解,而吳柏翰老師帶來了很多目前最新的技術,讓我們在學習上跟得上時代,也讓我們在學習上可以有許多新的想法,不再像以前學習教科書一樣,總是學習過去十年以上的知識與科技,雖然知識準確性極高,已經經過無數學者的千錘百鍊,確保是正確的,但學習極新的知識也是一種全新的體驗與挑戰,在學習的過程中,因為科技與技術過於新穎,別人提出的想法與理論可能是錯誤的,如何找出其中的奧妙便是其中的樂趣,也是以前我們讀取教科書所沒有的經驗,相信在以後,這將是的重要的技能與經驗,謝謝老師與同學在這一年的教學相長,讓我在這年有所成長。

6. 參考文獻

- Bengio 《A Neural Probabilistic Language Model》，JMLR 2003
- Tomas Mikolov (Word2vec)Distributed Representations of Words and Phrases and their Compositionality
- Tomas Mikolov, Linguistic Regularities in Continuous Space Word Representations, 2013
- Tomas Mikolov, Efficient Estimation of Word Representations in Vector Space, 2013
- Kumar Ravi a, Vadlamani Ravi, survey on opinion mining and sentiment analysis: Tasks, approaches and applications, 2015