Final Project---新聞立場檢索

NTU_B06902024_叛徒啦叛徒

B06902024 黃秉迦 B06902066 蔡秉辰 B06902067 許育銘 B06902074 柯宏穎

簡介

在這個資訊爆炸的時代，我們可以從非常多的平台得到各式各樣的訊息與議題。那麼多的主題與立場，沒有一個人有時間全數閱覽過，也沒有人能接受每一種議題。在時間有限的情況下，是否能達到有效地分類，且從中獲取重要的訊息？這個主題十分適合忙碌的我們去做研究，也能拿來運用在不同的領域，找到自己喜歡的主題，深入鑽研。

在這次報告中，我們將從100000篇文章中找出與20個爭議性議題最符合特定立場的文章各300篇，在新聞立場檢索技術獎金賽做評分，作為我們方法好壞的標準。

```
Query_Index,Query
q_01,通姦在刑法上應該除罪化
q_02,應該取消機車強制二段式左轉（待轉）
q_03,支持博弈特區在台灣合法化
q_04,中華航空空服員罷工是合理的
q_05,性交易應該合法化
q_06,ECFA早收清單可（有）達到其預期成效
q_07,應該減免證所稅
q_08,贊成中油在觀塘興建第三天然氣接收站
q_09,支持中國學生納入健保
q_10,支持臺灣中小學（含高職、專科）服儀規定（含髮、襪、鞋）給予學生自主
q_11,不支持使用加密貨幣
q_12,不支持學雜費調漲
q_13,同意政府舉債發展前瞻建設計畫
q_14,支持電競列入體育競技
q_15,反對台鐵東移徵收案
q_16,支持陳前總統保外就醫
q_17,年金改革應取消或應調降軍公教月退之優存利率十八趴
q_18,同意動物實驗
q_19,油價應該凍漲或緩漲
q_20,反對旺旺中時併購中嘉
```

資料預處理

由於中文大多需以「詞」為單位才能表達完整的意思，因此我們使用jieba將文章做斷詞，由於詞彙眾多，使得one-hot encoding過於龐大，因此我們使用Word2Vec做word embedding，將詞降到500維，並以此找出特定詞的相似詞。

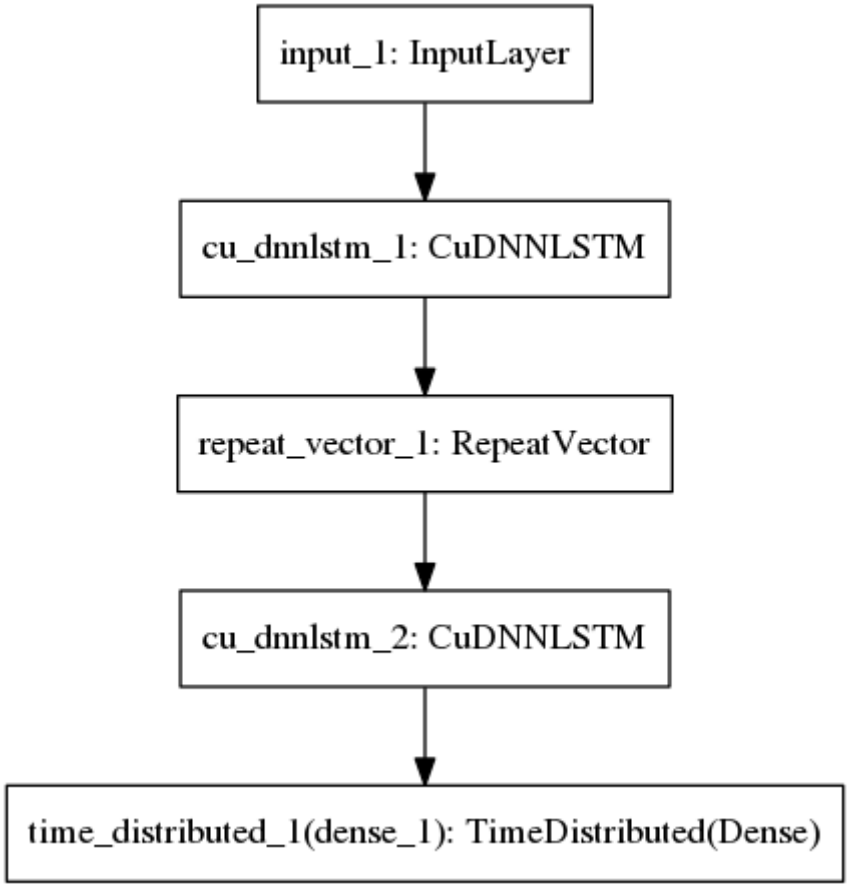
方法

1. 使用autoencoder將文章與議題立場encode，以此找出文章和立場的相似度。

詳細流程及使用參數如下：

1. 輸入資料處理：
   因為將文章詞數做平均後，其值約落在400附近，故我們將文章和query的句子都切成長度為400個詞的句子。在此處，我們將"query句子"也視為文章，並將其用和文章相同之方式一起放入autoencoder。而若有不足400字詞的文章則在後面補上"。"，直到達400字詞為止。
2. Autoencoder：
   取10%的資料量做validation，並train 30 個 epochs。架構如下圖：

- Encoder:
  $Input \rightarrow CuDNNLSTM(192)$
- Decoder:
  $RepeatVector() \rightarrow CuDNNLSTM(192) \rightarrow TimeDistributed(Dense(200))$
- 詳細架構如下圖：



3. 選取相近文章：
   取$cosine\ similarity$，找數值最高之300個新聞即為我們要的輸出。

2. 使用tf-idf判斷議題立場的某些詞彙在文章中的重要性。

利用tf-idf計算詞彙與文章的關聯程度。

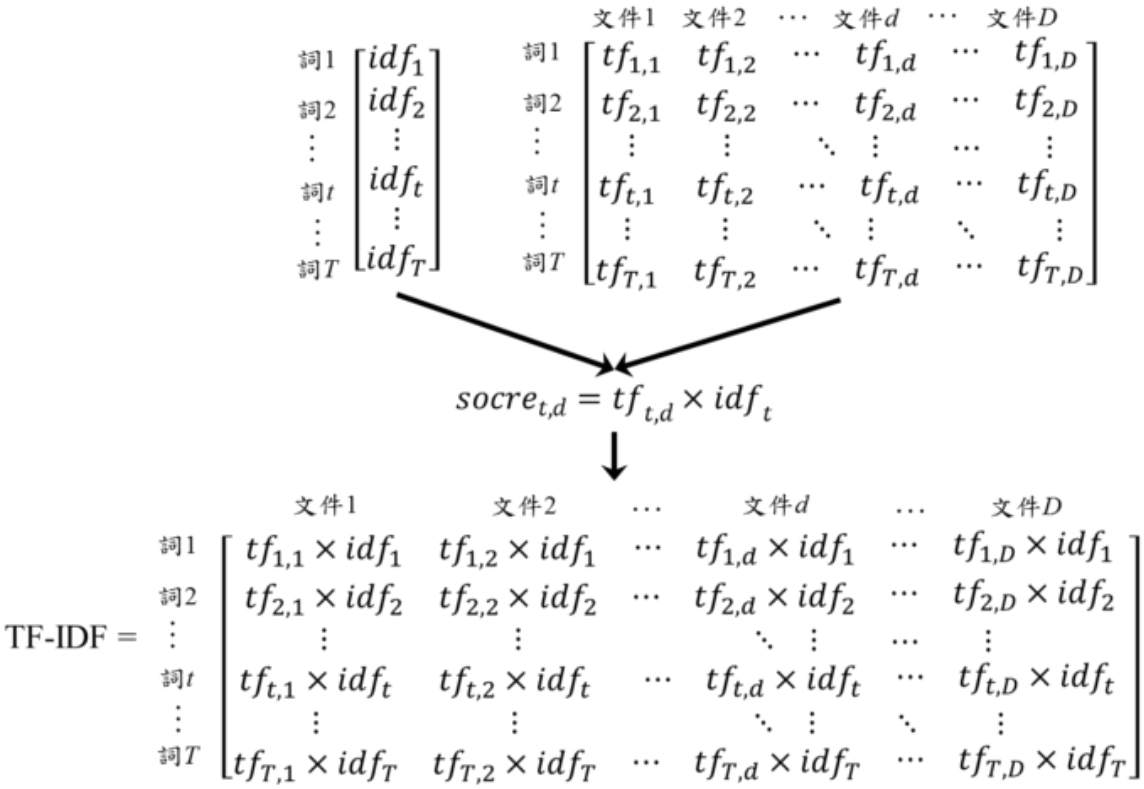
對於某一文章a、某個詞彙q，計算$tf(a,q) \times idf(q)$

$tf(a,q)$: 詞彙q在文章a的出現頻率

$idf(q)$: 考慮所有文章中有出現詞彙q的篇數，計算$\log \frac{\text{總文章數}}{\text{出現文章數}}$。

此可用來簡單地計算一個詞的重要度，如`的`、`是`這些詞，幾乎每篇文章都會出現，故在此情形下，這些詞的idf算出來就會很接近0。才不會讓那些常出現的詞影響到我們欲輸出的結果。

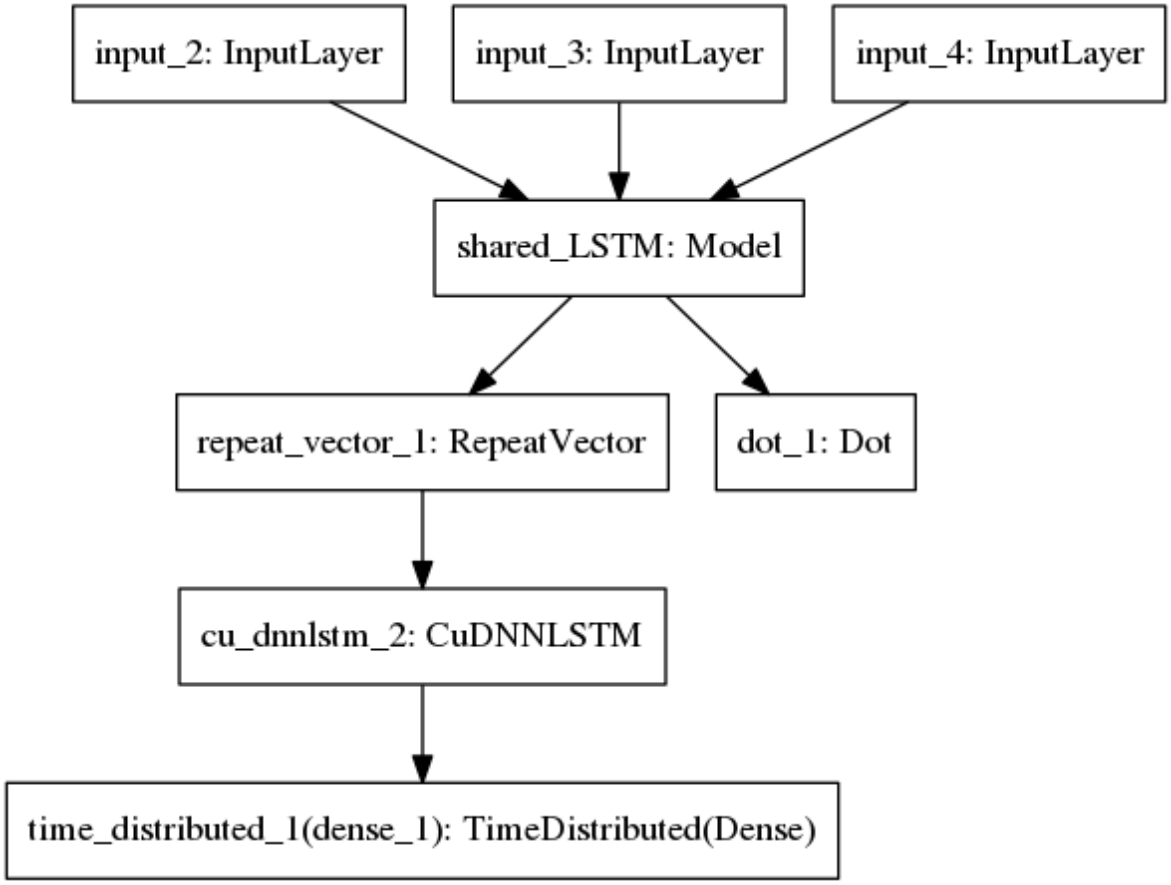
$tf\text{-}idf$的示意圖如下：



3.Autoencoder 搭配 training data

我們將原始文章和訓練資料一起丟入相同的$encoder$。接著，再分別將encode完的原始文章丟入$decoder$，得出$output1$；encode完的訓練資料丟入$cosine\ similarity$，得出$output2$。

詳細model圖如下：



其中：$input_2$為新聞文章和query句子，$input_3$為訓練資料中的query句子，$input_4$為訓練資料中的對應新聞文章。

而在訓練資料中，我們有進行以下處理：

- 將 Relevance = 0 的資料刪除，因為我們並不知道0所代表的是"完全不相關" 還是 "持反面意見"。
- 將Relevance之值除以三，因為$cosine$的值介於[-1, 1]之間。
- 將訓練資料中的query改成相反的論述，並將relevance改成負的，此不但可以獲得更多訓練資料，也可以讓判定value時有小於零之data。(e.g. 下列的其中一筆資料：**"支持陳前總統保外就醫，news_064209,2"** 將會被改成 **"反對陳前總統保外就醫，news_064209,-0.66"**)

而此方法目前會遇到memory error的問題，我們仍在嘗試使用其他方式或尋找其他資源來解決此問題，希望此方法能在日後有不錯的表現。

實驗與討論

Autoencoder

此部分成效並不佳，結果呈現於底下之實驗結果中；而成效不佳之原因我們將放在結論中進行說明。

tf-idf

- 提升performance之方式：

  我們主要着墨於此方法。我們先利用$jieba$將全部文章做分詞，跑遍所有詞，用來計算idf，同時我也會記下這個詞在哪篇文章出現過，出現過幾次，方便後面做使用。

  tf的部分，我先計算每篇文章的總詞數。也因為我在計算idf時便已經儲存每個詞在哪個文章出現幾次，我只要在計算分數時，用那個數字去除以文章的總字數，我便可得到詞頻。

  預處理完這些詞後，我們將所有的分詞丟入$word2vec$，目的是為了找出這些詞的近似詞。$tf\text{-}idf$的方法，主要就是做字串匹配，給予不同的權重。至於我們該丟如哪些詞給他找？我們原先只將20條$query$做分詞，一個個丟入去做計算。不過明顯地，這樣的統計量是絕對不足的，因此我們便採取找近似詞的方法，一同下去做計算。我們直接使用$gensim$的$word2vec$，再利用$most\ similar$去找。

  以上可用下列式子表示：

  $$score(Q,a)=\Sigma_{q\in Q_{sim}[:n]}tf(q,a)\times idf(q)\times s$$

  其中：Q代表query的句子；q表示句子中之單詞；$Q_{sim}[:n]$表示將Q丟入word2vec後，前n個相近的句子；s表示相似度。

- 過程

  如前面所述，資料量非常的龐大。若我們直接將上圖的$tf\text{-}idf$矩陣儲存起來，至少會有600000×100000筆資料，無論是儲存上或是計算上(memory error)都會造成很多困擾。因此我們採取以時間換取空間的方法，分開計算後再相乘。過程中也有遇到了一些演算法的問題，如計算idf時，利用$dictionary$能達到$O(1)$的複雜度，並在traverse的過程邊跑邊存，把執行時間從一個月降低到幾十分鐘左右$(O(n^2)\rightarrow O(n))$。

實驗結果：

以下是單純用原query句子去做$tf\text{-}idf$和做了調整後的表現比較：

處理方式	performance
無相似詞	0.331
3個相似詞、200維	0.408
3個相似詞、500維	0.395
autoencoder(cos similarity)	5.5e-5
autoencoder(distance)	5.48e-5

其實純粹用$tf\text{-}idf$下去做計算，最好的performance大概到0.26左右而已。以上所有的實驗，我們均有利用了主辦方所給的$training\ data$，直接將有相關性的文章(相關度$3 \rightarrow 1$)，優先塞入欲輸出的答案裡，此舉可使最好的performance上升到0.408。

不過這方法比較沒辦法判斷立場，我們也嘗試過，將反義詞(ex. 若此篇文章為同意某個議題，將`反對`一起丟進去找反義詞，並將找到的文章乘上一個懲罰數字)。不過效果並沒有比較好，因此便取消了這項計算。

結論

1. 在此次報告中，我們發現使用傳統的文字探勘技術+word2vec可得到的perfomance較使用autoencoder來得好上許多。
2. 造成使用autoencoder表現較不佳之原因可能有以下：
    1. 因為不足400字的文章在後面補上"。"，若autoencoder會把很多句號的那些文章判定為相近的文章，則performance自然就很差。
    2. 可能在autoencoder中，我們encoder輸出為192維有點太大或太小，使得autoecoder的表現較差。不過太大記憶體與時間需要更大量，會需要更多的運算資源。
    3. cosine similarity和distance不一定是最好的判定相近之方式，但因主辦方給定的training data過少，故要train出一個model去取代掉此兩種方式並不是一件容易的事情，很有可能一不小心就造成overfitting。
3. Tf-idf雖然表現比較好，但依然存在某些問題：
    1. 較不能判斷立場的問題。我們去找`喜歡`，可能會找到`不喜歡`等詞，我們也沒辦法完全防止這些否定詞一定要用`討厭`等詞來表示。
    2. 有可能句子會有"雙重否定"的使用(e.g. `我並不是討厭...`)這種情況應該較偏向正面的"喜歡"之意，但判定時難以對這種情形進行判定。
    3. idf可以表現出稀有性，但不一定能表現出重要性。舉例來說，像是`同意動物實驗`這個query之中的`實驗`一詞。我們可以合理的認定，`實驗`在全部文章中會出現的比例並不高，但可能出現像是`化學實驗`這種詞彙出現。此時他應該判定是無關，但因為idf之值很高，故會造成此方法將其誤判成有關的新聞。

參考資料

- https://github.com/ntu-csie-irlab/News_Stance
- http://www.cc.ntu.edu.tw/chinese/epaper/0031/20141220_3103.html

- https://medium.com/@chih.sheng.huang821/%E6%A9%9F%E5%99%A8%E5%AD%B8%E7%BF%92%E6%87%89%E7%94%A8-%E5%9E%83%E5%9C%BE%E8%A8%8A%E6%81%AF%E5%81%B5%E6%B8%AC-%E8%88%87-tf-idf%E4%BB%8B%E7%B4%B9-%E5%90%AB%E7%AF%84%E4%BE%8B%E7%A8%8B%E5%BC%8F-2cddc7f7b2c5
- https://github.com/keras-team/keras/issues/10333?fbclid=IwAR1hQPXL6gRq9C1ncItTfd6PQbcF2kbIKw_TPPARowthA6IjoDvqq6Q3CBA