Machine Learning Final Report
新聞立場檢索技術獎金賽

Team Member：張晁維，陳威旭，陳建成，蔡松達

Abstract

在機器學習的範疇之中，目前針對文字處理、分析的問題已經有各式各樣的模型被提出，像是從基本的RNN、LSTM，到Seq2Seq、TF-IDF、BERT不同種類的embedding方式都有其不同的成效，而這些模型提供出embedding的結果往往是透過前後文、語意、出現頻率所得到，對於後續不同的應用帶來不少便利。本實驗主要透過許多不同方法進行embedding，以方便找出與query相關性最大的幾則新聞，並利用一些模型架構試圖將立場（支持或反對）判斷清楚，以期透過受限的資料訓練過後能夠得到相當的效果。

I Introduction & Motivation

本次project的題目是給定一個議題的名稱，其中大多帶有特定立場(支持或反對)，希望能夠訓練出一個類似搜尋引擎的模型，幫助我們輸出與此議題關聯性由高到低、立場由較相同至較不同的新聞排序，此題目提供了100000則新聞連結，4742條某篇新聞與特定議題關聯性強弱的訓練資料，透過這些給定的資料進行模型訓練，期望能夠在輸入一個議題名稱時得到上述新聞排序的輸出。

在現在資訊爆炸、快速流通的時代中，各家傳播媒體在報導同樣的新聞時往往可以由不同的角度切入，由於不同媒體對於特定議題時常會有不同的觀點與看法，報導的面向往往會由不同的角度切入，尤其是針對一些具有較大爭議性的議題，例如經濟、教育、政治等等，同一個議題在不同的新聞媒體呈現出來時卻可能會有截然不同的立場跟說法。本題目希望能夠找出一個可以對於立場相關度排序的搜尋引擎，這樣的想法可以方便我們查詢特定議題中特定立場的理念與想法，不同的觀點、想法能夠促進大家對於特定爭議議題的整體了解，當自己抱持著相當鮮明的立場時，若是也能夠針對完全顛倒的立場進行探究，往往能對於該議題有更深的體會。此外，一般的搜尋引擎基本上不會明確將立場相對的新聞內容區分開來，輸入關鍵字搜尋時得到的結果往往是夾雜著不同立場的，若是能夠明確的找出立場相同的新聞內容，並將立場最相近的新聞排在搜尋結果的最前端，對於使用者將會變得更加方便，且對於研究特定議題也有相當大的助益。而我們這組對於文字處理也較有興趣，期望能夠運用所學來完成此題目。上述大致上為我們這組選擇此題目的主要動機。

Related Work

在進行model實作之前，我們有經過許多討論與嘗試，對於不同的方法進行了實作。與此相關的論文有

A.

II Data Preprocessing/Feature Engineering

我們對資料的處理主要分為兩個部分：Data Representation和Ranking Adjustment。前者實作方式有BOW(Bag of Word)、Explicit Embedding和Implicit Embedding，不同的Data Representation方式分別應用於第三部分的不同方法；後者實作方式則是對預測結果以訓練資料進行重新分群再排序。

一、Data Representation：

(一)BOW(Bag of Word)：

(二)Explicit Embedding：

Explicit Embedding方法中Embedding model和之後的分類器是分開來訓練的，如使用gensim套件的Word2Vec embedding和Doc2Vec embedding，再分別經過不同的方法改良embedding的結果。兩者共通點為皆利用jieba套件進行中文斷詞後再進行embedding，斷詞辭典使用作業六提供的中文繁體辭典。

1. gensim Word2Vec embedding：

……(陳建成)……
(1) keyword embedding：
……(陳建成)……
(2) Average embedding：
……(陳建成)……
(3) TFIDF weighted embedding：
……(陳建成)……

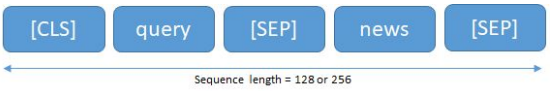
2. gensim Doc2Vec embedding：
取得所有新聞資料後先用jieba.cut函式進行斷詞，再將每篇斷詞過後的新聞資料重組成gensim Doc2Vec模型需要的TaggedDocument物件，調整Doc2Vec需要的模型參數(min_count, vector_size等等)後對Doc2Vec模型進行訓練，再利用此模型取得每篇新聞資料與查詢題目的embedding vector。查詢題目的embedding vector取得方式為將斷詞後的查詢題目傳入訓練完畢的Doc2Vec模型的infer_vector() method，便會回傳該文章(句子)依照此Doc2Vec模型得到的embedding vector；而新聞資料的embedding vector則是呼叫Doc2Vec模型的物件docvecs，用index的方式得到該篇新聞資料的embedding vector。

(三)Implicit Embedding：
Implicit Embedding方法為將Embedding Vector直接接著輸入NN(Neural Network)或是其他分類器進行相關性，和其他Embedding方式有所區別故分開討論。

此方法主要為使用BERT Pretrained Model來實作Sentence Embedding，使用BERT Tokenizer 將查詢題目與新聞資料依照BERT模型輸入方式進行預處理[1]，採用BERT模型Pre-training 任務中NSP(Next Sentence Prediction)方式得到每個查詢題目與新聞資料的Embedding Vector。我們認為BERT進行NSP的任務可以類比於訓練資料中查詢題目與新聞資料的相關性。

欲使用BERT模型進行NSP任務我們需要將查詢問題與新聞資料tokenize後串接成sequence length小於512的陣列，而實作上BERT Pretrained Model輸入的sequence length只要大於256我們的硬體都無法負荷，故我們輸入BERT的sequence length僅有嘗試總長度128與256，如下圖所示。

圖一：BERT模型輸入sequence



二、Ranking Adjustment：

經過一些方法(如第三部分所述)後我們得到各篇新聞資料對各個查詢題目的立場與相關性的排序，觀察查詢題目與訓練題目的相關性與相似程度之後發現可以利用訓練題目已知的相關性來改善我們對相同題目的排序結果。

實作方法有兩種，其一是直接讀取任一方法排序後的結果比較查詢題目與訓練題目的相似度，紀錄這些和訓練題目相似的查詢題目並且找出該訓練題目與訓練資料中新聞資料標記之相關性(Relevance)，取相關性大於0的新聞資料，將這些新聞資料依據相關性做降序排序後做為輸出結果，如有不足三百名的部分則是再加入原本查詢題目使用方法得到的排序結果的前幾名補足三百名的排序結果，如果出現補足三百名的某則新聞資料已經由訓練資料依據相關性排入的情形，那麼該則新聞資料會被略過，轉而觀察下一順位的另一則新聞資料，直到補足三百名為止。

此種實作方是的優點是快速且方便實作，由於直接使用各查詢題目對各新聞資料的排序結果，可以很方便的接在各種搜尋方法後面進行結果修正。然而此方法的缺點也是顯而易見的，即不考慮先前方法所得到的排序結果依據為何便直接由訓練資料內相關性高的新聞資料取代，可能導致具有高相關性的資料被排序至低順位。

第二種實作方法則是取得全部新聞資料對各查詢題目排序的依據，即演算結果的相似度(cosine similarity等等)，以及訓練資料中和查詢題目相似之訓練題目對應新聞資料與其相關性，對一相似訓練題目的查詢題目的所有排序新聞資料而言，如有出現在訓練題目內的新聞資料則將其演算結果的相似度加上訓練題目與該新聞資料的相似度，而如果該新聞在訓練資料中相似度為0則將其演算結果相似度乘以0.5，全部相似題目內的新聞資料相似度都重新處理過後再進行排序。

此方法利用訓練資料的相似度對預測結果做分群的，可以保留原本演算結果得到的相似度關係，如相似度判斷為較高的新聞資料加上其原本在訓練資料中對相同查詢題目的相似度仍可以維持其相似程度，而若是沒有出現在訓

[1] BERT模型架構與輸入資料預處理方式請參考VI-1。

練資料中而演算結果為高相似程度的新聞資料在最後的排序中仍有1左右的排序相關性。

III Methods

此部分將會介紹我們曾經使用過並且較具有代表性的分析方法，配合不同的Data Representation有不同的Similarity Analysis方法；某些Similarity Analysis方法則不使用第二部分中提及的Data Representation方式，直接使用rule-based的方法對資料進行相關性的排序。

一、Doc2vec + cosine similiarity：

先用gensim同時找出每篇文章和query的document vector，再用每篇文章的vector和query的vector做cosine similiarity，拿和各個query最相近的前300出來輸出。

二、關鍵字搜尋法：

針對文章中，所有出現的jieba分詞進行搜尋，假設有出現與query相同的特定的詞，就給那篇文章加分，此外也針對部分query進行限制，若是特定詞彙沒有出現就算得分很高也會將分數歸零，例如當query為"通姦應該除罪化"，但其實很多新聞時常報導有關於外遇的新聞，這些新聞往往與除罪化議題毫無關聯，因此必須設定若是沒有出現"除罪"詞彙時即便"通姦"出現多次，分數也將會歸零，以排除不具關聯性的新聞。最後統計出每則新聞對於特定的query所得的分數後，將前300高分的新聞排序出來，即為所求答案。

三、關鍵字搜尋法改進版：

新增自訂的jieba字典來做分詞，使jieba分詞時，可以分出我想要的詞來做加減評分，然後每次做完就打開結果的文章，看是否有太多錯誤的文章出現，有的話，就在更動自訂的jieba字典和更動評分方式，經過多個iteration後，基本上可以達到非常好的效果。和reinforcement learning有異曲同工之妙。

四、TF-IDF Bag of words by sklearn library：

先經過jieba斷詞處理，接著使用scikit-learn套件的Vectorizer和TfidfTransformer，將原本的字串轉換成one-hot vector之後，再根據每個文件每個單詞出現的詞頻（term frequency，TF）與反向文件頻率（inverse document frequency，IDF），得到權重過後的BOW向量（約10萬×57萬維的sparse matrix）。
接著將Query set的每個query分別與news進行cosine similarity直接比對。

五、Average Word Embedding：

將所有news、TD與QS合併起來當做一個大corpus，經過jieba斷詞處理後使用Gensim的Word2Vec套件，embed到400維的向量上，然後將每個news document出現的文字之embedding vectors直接進行平均，得到10萬×400維的matrix。接著與前面相同，將Query set的每個query分別與news進行cosine similarity直接比對。

六、Binary Stance Classification：

基於前敘方法多為考慮查詢題目與新聞資料的相似度
Data Representation為採用Implicit Embedding方式

IV Experiment and Discussion

一、Doc2vec + cosine similiarity：

準確率：0.0124075

可能問題：Gensim使用的Doc2Vec演算法，對於每個document的embedding的時候容易造成嚴重震盪，因此無法得到穩定的embedding，進而無法在後續的比對上得到很好的效果。

二、關鍵字搜尋法：

準確率：0.1358650

可能問題：只有對於文本出現的關鍵字進行搜尋評分，無法對於立場進行判斷，此外這裡所使用的詞彙僅限於query中有出現的詞彙，因此有些新聞如果使用相似詞則無法得到分數，也是可能的因素之一。

三、關鍵字搜尋法改進版：

準確率：

可能問題：經過多次iteration後，最終結果得到非常好的成長，只是較缺乏通用性，於Conclusion詳細解說

四、TF-IDF Bag of words by sklearn library：

準確率：0.1918345

可能問題：完全沒有考慮到training data，純粹就文件本身的資料特性去做unsupervised統計。另外，因為資料量實在太大，容易將工作站（Google Colaboratory）的14G RAM全部使用殆盡而導致工作站多次崩潰。

五、Average Word Embedding：

準確率：0.1646697

可能問題：一樣完全沒有考慮到training data，word embedding的training也是unsupervised的。另外就是每個詞語的權重不應該是完全相同的。

V Conclusion

1. 其實暴力搜尋法+新增自訂jieba字典，就可以得到很好的效果，只是較為缺乏通用性，即每次有新的query出現，就需要重新新增字典和找出新的特定字來做加減分。在真正的大量搜尋時較沒那麼好。

VI Reference

1. BERT: https://arxiv.org/pdf/1810.04805.pdf