B04902060　周良冠

Information Retrieval Programming Assignment 2

I. 探索資料

官方給的資料集是每篇新聞的url，可以從url得知新聞的來源。出於好奇心，於是就分析了資料集的新聞是從哪些網站爬下來的，結果如下表所示。（至於這有什麼用我也不清楚，但很好奇為什麼中國時報佔大宗，可能因為文章比較好爬，或是有合作？）

新聞來源分佈	
中國時報	43535
蘋果相關	22358
自由時報	19311
聯合報	7869
tvbs	6927

II. 實驗進程

1. Okapi

首先觀察助教提供的simple baseline，是很簡單的TFIDF，於是稍微修改成okapi，成績有顯著提升到0.2，調參數後可以到0.204。

觀察到在立場分析的應用上，stopword可能會是不必要的noise，剔除stopword後，搭配okapi些微提升至0.207。

觀察jieba的斷詞，發現有些query被斷得很奇怪，例如「中國學」、「生納入」。於是我試著自己手動調整成合理的斷詞，但成績竟然退步。我認為這可能是因為助教提供的inverted-file也是用jieba斷的，本身就存在錯誤。

2. Language Model

試了以上方法後，發現VSM Okapi很難超越strong baseline。於是實作老師上課所教的language model，用document生成query的機率大小當作ranking的依據。其中也使用了smoothing的技巧，種類包括Dirichlet和Jelinek-Mercer。但這方法效果很差，分數大約都落在0.6左右。

3. FastText

為了突破strong baseline，上網下載已經預訓練好的fasttext詞向量，訓練語料是中文維基百科，而詞向量有300維。我使用的方式是，直接將document和query出現的詞的向量總和起來，並且取cosine相似度，成績為0.12，並沒有很好。我有聽同學使用相同方法且成績達到0.24，但他的詞向量有經過比賽的新聞文本fine-tune過，而我並沒有這麼做。

4. Peak

最後為了過strong baseline，使用了類似偷看答案的方式。其實一開始我就有發現TD.csv中和我們要求的query有5個是重複的，我就把分數從3到1排序，並且直接從rank1往下排。藉由這個方法，最後5筆query的分數大幅提升，總分數直接提升到0.35。

實驗結果	
Simple baseline	0.1556762

B04902060　周良冠

實驗結果	
Okapi	0.2043089
Okapi w/o stopwords	0.2074112
LM (Dirichlet)	0.0621598
LM (Jelinek-Mercer)	0.0578911
FastText (cosine)	0.1206317
Okapi (peak)	**0.3520132**