Programming HW1 Report

資工碩一 R07922004 吳星耀

1. Describe your VSM (e.g., parameters…)

   我首先先建立文件 id 與檔名的辭典、詞的 id 與詞的對應辭典。接著，我從 invert-file 中求出實際有出現在文件中的詞的辭典、每個詞在每個文件中的對應詞頻(c(t, d))，以及由每個詞在每個文件對應詞頻求出每個文章的長度。由長度計算出 avdl (average doc length)。我用 c(t, d)算出 Okapi TF，並用 normalizer 做 normalize，使得：

   $$\mathrm{TF}_{\mathrm{doc}}(t,d) = \frac{k * c(t,d)}{c(t,d)+k} * \frac{1}{1-b+b*\frac{\mathrm{doclen}(d)}{\mathrm{avdl}}}$$

   其中經過調整參數，我發現 b=0.287, k=1.2 的效果較好。

   接著我做 query expansion。我使用了 query 裡面的 concepts，並且將每個 concept 切出 unigram、bigram 與英文等作為 query 的關鍵字，每個關鍵字維度的大小為此關鍵字在 concepts 中出現的頻率。例如"英語教學"，我會切成"英"、"語"、"教"、"學"、"英語"、"語教"、"教學"，接著我將這些字中沒有出現在先前存的辭典中的刪除，剩下的作為 query 的 term，query 在這些字的維度的值加 1。

   我算出每個字的 IDF，以$\mathrm{TF}_{\mathrm{query}}(t,q) = \frac{(k+1)c(t,q)}{c(t,q)+k}$作為 query 的 TF，然後一個對一個 term，一個文章的分數則用以下方式計算：

   $$\mathrm{score}(t,d) = \mathrm{TF}_{\mathrm{doc}}(t) * \mathrm{TF}_{\mathrm{query}}(t,q) * \mathrm{IDF}(t)$$

   將 query 中的每個 term 所得到的分數總合，即是這個 doc 得到的分數。最後依照分數高低來排序 doc，沒有另外設 threshold。

   這個方法能在 kaggle public score 拿到 0.81477，private 拿到 0.71756。

2. Describe your Rocchio Relevance Feedback (e.g., how do you define relevant documents, parameters…)

   我將由 SVM 得到的排名前 15 名作為 relevant document，100 名後 15 名作為 non-relevant document。接著我將 relevant document 中所有的包含的 term 都作為有關的 term，將他們的 TF 乘上 beta=0.3 除以 15 加到新的 query 中，non-relevant document 中所有的包含的 term 都作為無關的 term，將他們的 TF 乘上 gamma=0.3 除以 15，新的 query 都減去這個值，最後把原本的 query 乘上 alpha = 1 加到新的 query 中，再對這個新的 query 用 SVM 重新排名。雖然在 kaggle 的 public score 只有拿到 0.79838，在 private 卻會上升到 0.75642(private leaderboard 會是第四名)。由於在 public 成績較低而沒有選擇這個結果，非常可惜。

3. Results of Experiments, including:
   - MAP value under different parameters of VSM
   - With Feedback vs. without Feedback
   - Other experiments you tried

關於參數調整，我嘗試了不同組 b 與 k，發現 b、k 在 0.287、1.2 的時候表現最好。

我調整在 rocchio feedback 的 alpha、beta、gamma，我固定 alpha=1，取的 relevant 與 non-relevant 數量為 8，測試 beta、gamma 的不同對 MAP 的關係。下表為結果：

beta、gamma	0.1	0.2	0.3	0.4
MAP	0.73267	0.73524	0.74135	0.7369

我在實驗中 beta 與 gamma 都彼此相同，這是因為我覺得相關的資料跟不相關的資料帶來的影響應該是等效的。

另外固定 alpha=1、beta=gamma=0.3，觀察 feedback 時使用不同 relevant 與 non-relevant 的數量對 MAP 的影響：

Number of relevant & non-relevant data	5	8	10	15
MAP	0.74285	0.74135	0.72957	0.74586

相對於 alpha、beta、gamma，feedback 的數目對 MAP 的影響較沒有規則，我最後取 15 作為參數。同樣的因為我覺得相關的資料跟不相關的資料帶來的影響應該是等效的，正負相關的資料我取的數目相同。

我比較了在有無 Feedback 的分數，如果 Feedback 不會增加新的 term，只是調整每個 term 的權重，所計算出的排名在 kaggle 上分數會完全一致。如果 Feedback 會增加新的 term，training set 計算出的排名會上升，但是在 kaggle 的 public score 雖然有上升，但 private 卻是下降的

我試過將 TF normalization 的方法改為$\mathrm{TF}_{\mathrm{doc}}(\mathrm{t},\mathrm{d}) = \frac{\mathrm{k}*\mathrm{c}(\mathrm{t},\mathrm{d})}{\mathrm{c}(\mathrm{t},\mathrm{d})+\mathrm{k}*(1-\mathrm{b}+\mathrm{b}*\frac{\mathrm{doclen}(\mathrm{d})}{\mathrm{avdl}})}$的方式來計算 TF，這個方法在 public dataset 得到的成績較差，但是在 private set 的成績較好。我以自己寫的 MAP 來調整參數，發現原本的 TF 計算公式在 b=0.7 的時候表現很差，在 0.3 時表現較好，但是在這個公式下卻是 b=0.7 表現較好。此外這個公式在 kaggle 的 public score 會較低為 0.79962(vs 0.81477)，但 private score 會上升至 0.73157(vs 0.71756)。

我也試了將 Non-relevant 的資料由 100 往後取一定名次改成取倒數幾名，但是在 MAP 表現反而下降。

4. Discussion: what you learned from the work.
   我從這次的作業中發現其實網路資料檢索所用的 model 都需要許多參數的調整，必須花費很多的心思。SVM 的參數非常多，很難調整，而且每個參數只要些微調整，在 testing set 的分數就會差異很大。在這次作業我的 rocchio feedback 的 beta 與 gamma 都是設相同的值，有時間的話想試試看設不同的值看看結果。