WSM Project2 Report

林宣佑
資管三 106306027
106306027@g.nccu.edu.tw

Abstract

此專案是基於 Anserini 完成的四個資料檢索模型，Anserini 是一個功能強大的工具，不僅快速且相關文件詳細。此份報告將重點著重於四種模型在不同的召回率下精確度的差異。另外則是第四個模型中 reranker 發揮了何種作用，以及各類平滑的優缺。

1 模型描述

在 anserini/target/appassembler/bin 這個資料夾下包含著各種 Anserini 可以使用的指令，其中最重要的當然是 IndexCollection 及 SearchCollection，以下分成四小節分別描述四個模型。而對於每個模型都套用於 stemmed 及 unstemmed 兩種 index。

1.1 Vector space model with BM25 weighting

向量空間模型，以 BM25 權重，並將 $k1$ 設為 2、b 設為 0.75

$$score(D,Q) = \sum_{i=1}^{n} IDF(q_i) \cdot \frac{f(q_i,D) \cdot (k_1+1)}{f(q_i,D) + k_1 \cdot (1 - b + b \cdot \frac{|D|}{avgd})}$$

1.2 Language modeling with Laplace smoothing

又稱為 add-one smoothing，目的是為了避免機率為 0，假設 MLE 如下:

$$P(w|D) = \frac{count(w,D)}{len(D)}$$

亦即，若單一字詞機率為 0 將造成整個文檔機率為 0，因此 Laplace smoothing 將以上公式改寫成:

$$P_{add-1}(w|D) = \frac{count(w,D)+1}{len(D)+|V|}$$

1.3 Language modeling with Jelinek-Mercer smoothing

若平滑可以分成 Interpolation 和 Backoff 兩種方式，Jelinek-Mercer 應屬於前者，以比例混和了對文檔以及對語料庫的估計，在此專案將 λ 設為 0.2

$$P_{JM}(w) = \lambda P + (1-\lambda)Q$$

其中 P 是來自文檔的估計概率 (max likelihood=m_i/n)

Q 是來自語料庫的估計概率 (background probability = $cf/terms$ in the corpus)

1.4 Vector space model with BM25 weighting and pseudo relevance feedback

在第四個模型中，為了強化第一個模型 (BM25)，試著套用了 reranker，為 BM25 的 pseudo relevance feedback，其用意是以原 query 為基礎，將結果排序後前 n 筆資料作為訓練集，並創造新的 query，步驟如下:

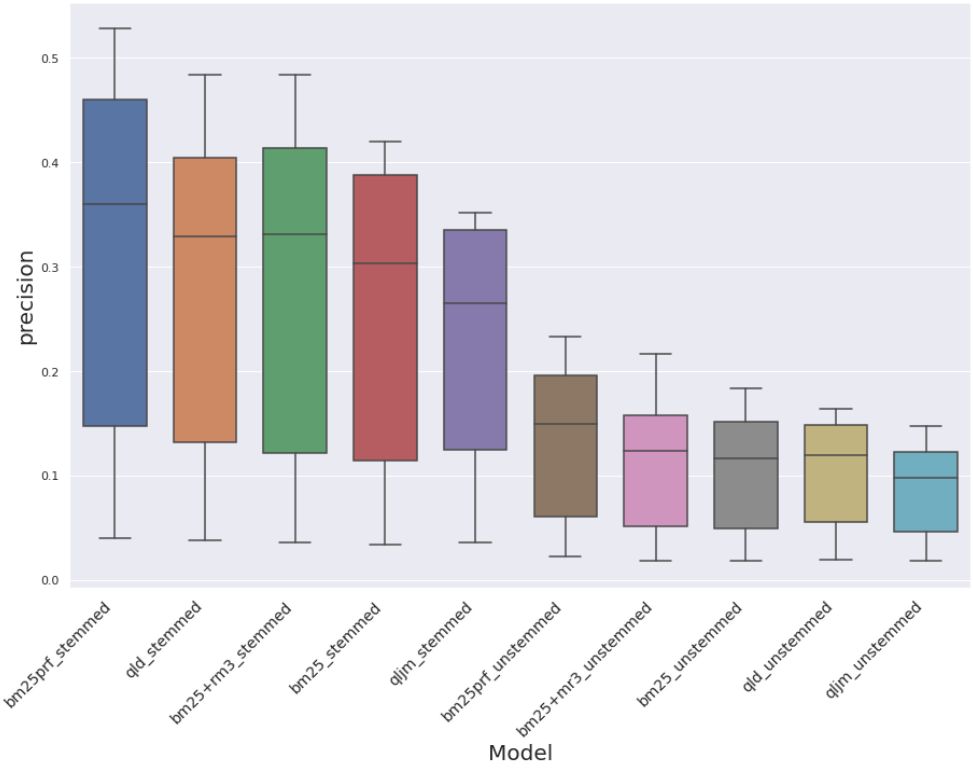
1. 用原始 query 檢索文檔
2. 取前 N 篇文檔的 term 作為 potential expansion terms
3. 為每個 potential expansion term 計算分數
4. 用前 m 個 term 創建新的 $Q_learned$
5. 用新的 query 檢索文檔

2 模型評估

在 Anserini 中有另一種 reranker 稱為 RM3，應該也是一種 relevance feedback，我試著套用在第一個模型上並在此節一起作為比較，以下分為兩小節討論未插值以及插值後的準確率。

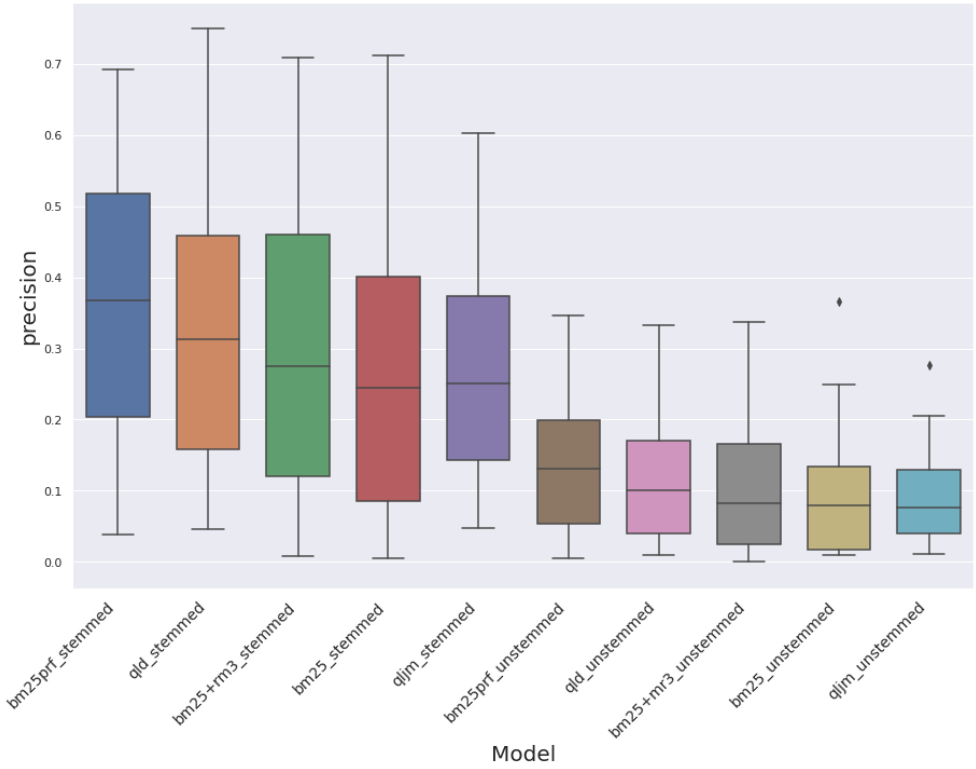
2.1 Average precision (non-interpolated)

未插值的 AP 分別是在文檔數為 5,10,15,20,30,100,200,500,1000 下的平均精確率。若文件有經過 stemming，則結果普遍較好。另一方面，全距變大許多也意味著靠前的文檔精準度明顯提高。

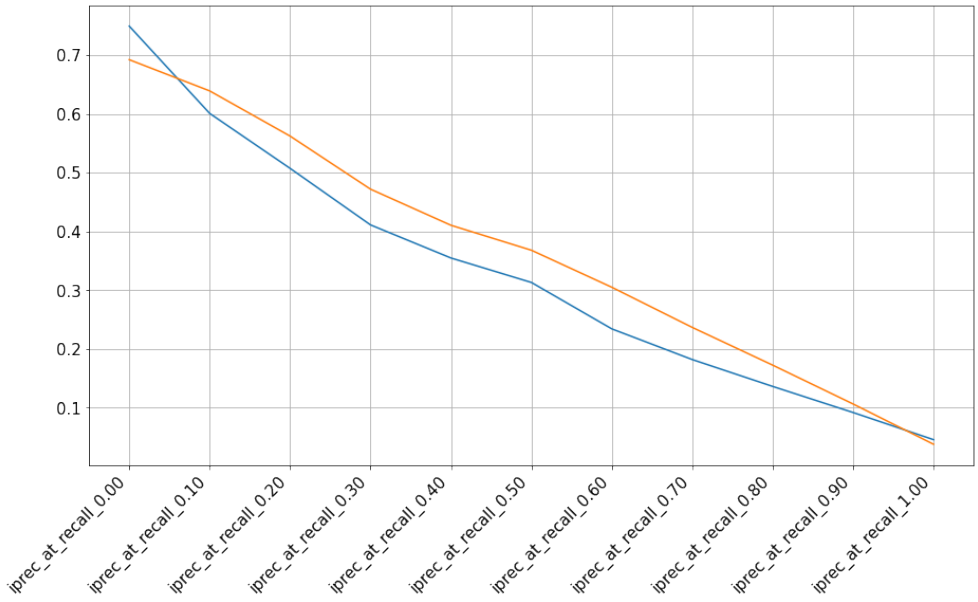


2.2 Average precision (11-point interpolated)

11 點插值的精確率是指召回率從 0.0 至 1.0 每隔 0.1 為單位的平均準確率，其值以該召回率後出現的最高準確率。在以下的圖中，我們可以看見和未插值的圖有相似的結果。



但出現了一個有趣的問題：bm25prf 和 qld 哪個模型比較好？以下將對兩者做更進一步的比較：



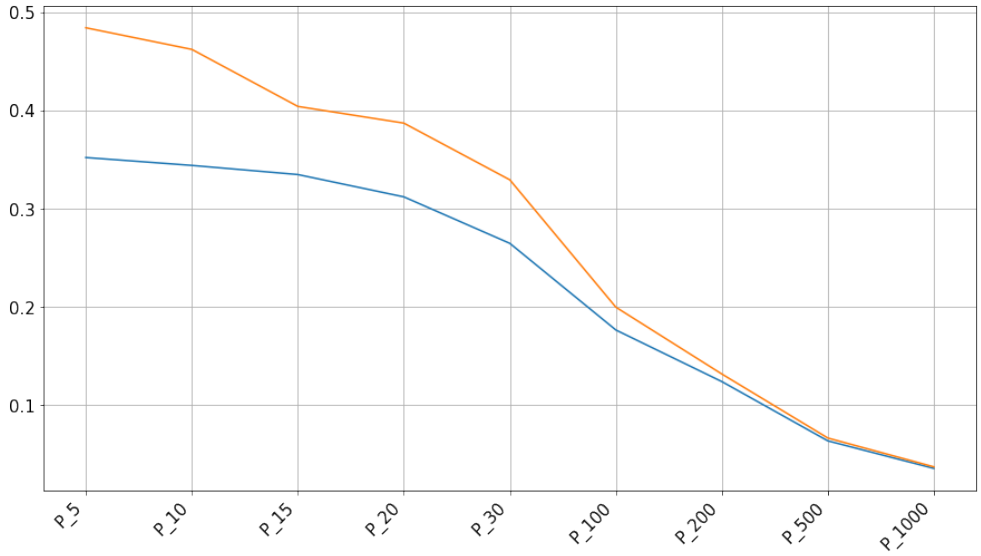
這裡我認為，兩個模型對於不同的使用者來說各有所長，若以上圖來解釋，前者整體表現較好，也就是說若使用者著重於大量的相關文件搜尋，則 bm25prf 應該更適合它；後者則適合需要更快速得到一些相關文件的使用者。

3 平滑以及字根化比較

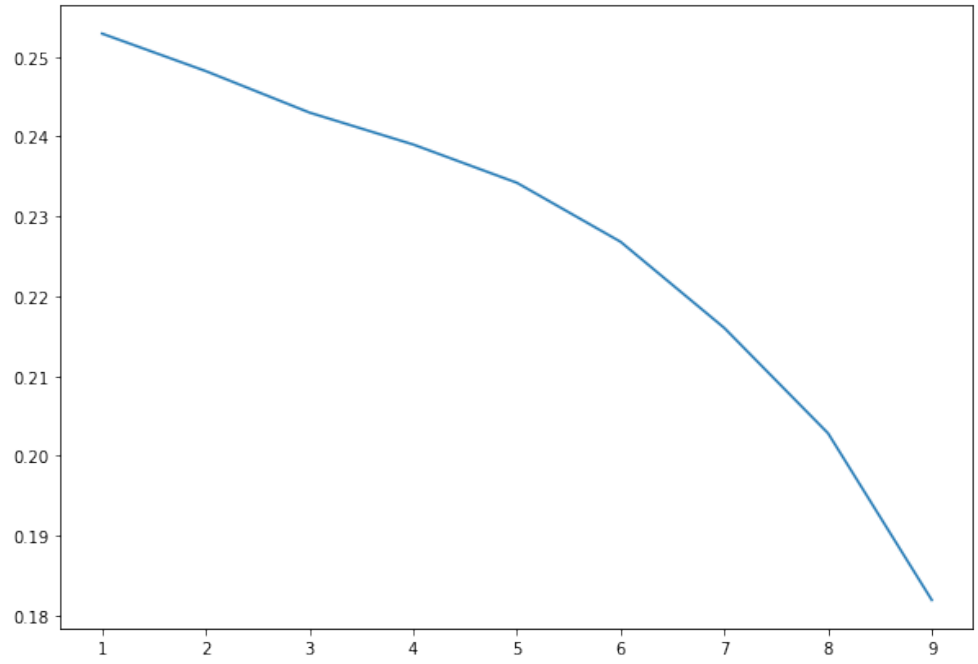
在這節中會比較語言模型的兩種平滑方式，如同第一節介紹模型所提到的，分別為 Laplace smoothing 做比較，其變化如下圖與 Jelinek-Mercer smoothing，以及去除字根與否的差別。

3.1 平滑

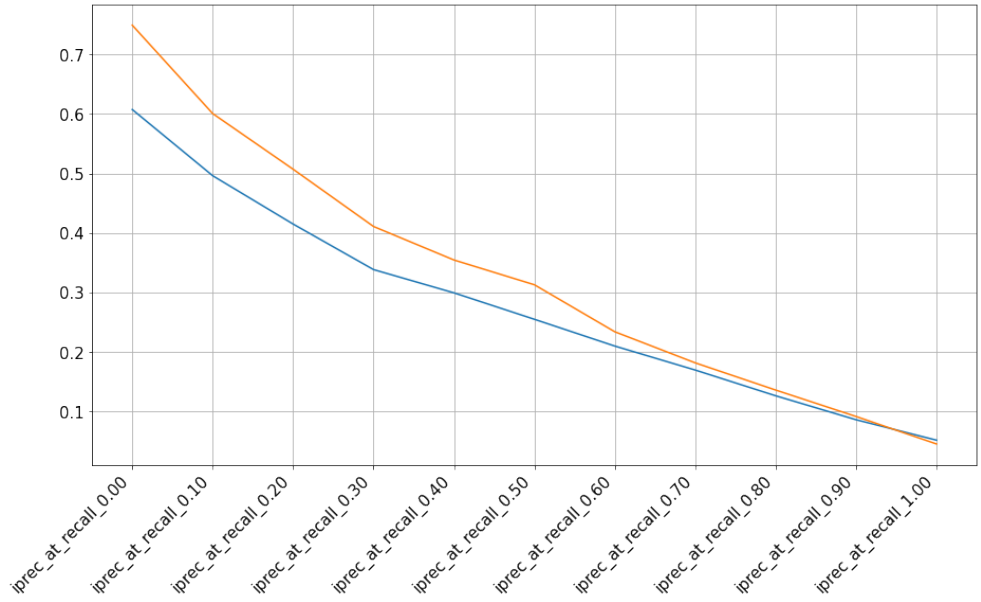
我們單看已字根化的 index，如同上節箱型圖所示，無論是整體或是對排序靠前的資料而言，qld 的表現都比較好，但我不認為這因此代表 Laplace smoothing 的平滑方式表現較為優秀，原模型的準確率變化如下：



接著我嘗試改變 Jelinek-Mercer smoothing 的 λ 值，對 λ 由 0.1 至 0.9，其 MAP 變化如下圖：



因此我認為在 λ 為 0.1 時應該會有較好的表現，但變化應該不會太大，也許在這組資料中 Laplace smoothing 的確較為適合，λ 為 0.1 時與 qld 比較如下：



3.1.1 字根化 (stemming)

stemming 的優點在於，可以將時態，詞性不同，但可能想表達相關意含的詞視為相同，這樣應該能更接近使用者所表達的意思。而在這組資料中，stemming 也的確發揮了很大的作用，幾乎所有經過詞幹提取建立的模型，表現都較優秀。

4 結論

在這個專案中，獲得最多經驗值的是使用 anserini 這類的套件，在經過專案一自己寫向量空間模型之後比較兩者，不得不承認語言速度的重要性以及有效率的方法能加快許多流程，例如學到使用 shell script 將 Jelinek-Mercer 中 lambda 值跑過一次，其實是非常快速的。

而模型的部份，bm25 在這次表現應該最優秀是無用置疑的，我相信還有一些模型及參數是我沒有嘗試過的，但時間關係我僅做了以上的比較。若對原始資料感興趣可以造訪我的 github repo，我將 trec eval 以及 retrivel result 放至於此：

`https://github.com/nathan-tw/anserini_wt2g`