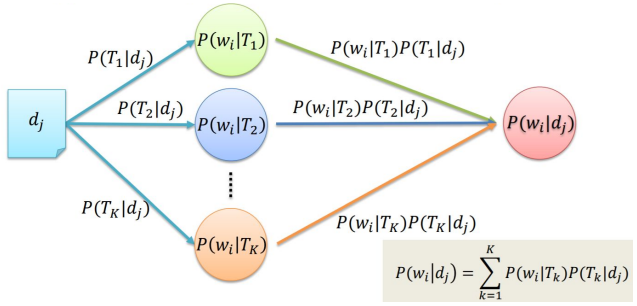
Information Retrieval and Applications

HW04-PLSA model with EM algorithm

M10915201陳牧凡

此次作業運用到的MODEL是PLSA model，跟前兩個作業最大的差別在於這個model有靠慮到語言中會有同義詞或是同詞異義的問題，例如同樣"java"一詞在講人文地理時可能是一座島嶼、或是一種咖啡豆，但在電腦資訊主題裡面的話就很有可能是指程式語言的java。為了解決這個問題我們會自訂義一個變數"主題(Topic)"，透過機率去計算這篇文章為該主題的機率、以及這些字詞出現在該主題的機率，交互去比對之後算出他們的文章相似度，如下圖。



那麼訓練其topic的方式是採用EM algorithm來進行計算:

- E-step

$$P(T_k|w_i,d_j) = \frac{P(w_i|T_k)P(T_k|d_j)}{\sum_{k=1}^{K} P(w_i|T_k)P(T_k|d_j)}$$

- M-step

$$P(w_i|T_k) = \frac{\sum_{d_j \in \mathbf{D}} c(w_i,d_j)P(T_k|w_i,d_j)}{\sum_{i'=1}^{|V|}\sum_{d_j \in \mathbf{D}} c(w_{i'},d_j)P(T_k|w_{i'},d_j)}$$

$$P(T_k|d_j) = \frac{\sum_{i=1}^{|V|} c(w_i,d_j)P(T_k|w_i,d_j)}{\sum_{i'=1}^{|V|} c(w_{i'},d_j)} = \frac{\sum_{i=1}^{|V|} c(w_i,d_j)P(T_k|w_i,d_j)}{|d_j|}$$

透過不斷EM training迭代的方式來得到一組較佳的P(T|d)和P(w|T)，進而計算文章相似度。

實作部分由於考慮到這次dataset較大，因此在算DOCTF的時候，採用sparse marix來儲存，而考慮到EM model的訓練時間，這部分會用@numba.jitannotation來事先編譯好這段程式碼，進而加快計算時間；具體時間差別大概會差到100倍左右。

PLSA由分成三個submodule組成，分別是unigram, em, bg model，如下式子:

$$P'(w_i|d_j) = \alpha \cdot P(w_i|d_j) + \beta \cdot \sum_{k=1}^{K} P(w_i|T_k)P(T_k|d_j) + (1-\alpha-\beta) \cdot P(w_i|BG)$$

由裡面的alpha, beta來決定各自部分的權重。以下是嘗試了幾組a, b參數的MAP分數結果:

topic=128 iter=100

a\b	0.3	0.4	0.5	0.6	0.7
0.3	X	X	X	X	0.558
0.4	X	0.556	X	0.572	
0.5	X	X	0.581		
0.6	0.570	0.586			
0.7	0.590				

(X代表沒有做，a+b>1的也省略不做)

可以看出一個趨勢是隨著unigram權重上升以及EM的權重下降，分數越來越高，因此我認為PLSA在unigram為主，EM為輔的情況會得到較好的結果，而BG則似乎沒有太大的作用。

基本上只要做出unigram和bg model就可以過baseline的分數了(大概0.48~0.50)，於是乎中間的EM model就成為了能不能高分的關鍵了，而影響EM model的因素有兩項: topic數量、training次數，以下是比較這兩項參數的MAP結果:

a=0.7 b=0.3

topic\epoch	50	100	200	500
48	0.567	X	X	0.560
64	0.569	0.581	0.570	X
128	0.571	0.590	X	0.591

由於EM training中初始值是random設定的，因此無法保證每次實驗都是以一樣的狀態下進行比較，不過大致上可以看出一個趨勢就是topic和epoch數高的話普遍較高分，但是高過一定程度之後就會飽和(或是overfitting)。

另外，從整體計算中所需要使用的參數還可以從中去做一些改變，第一個是在使用DOCTF進行M step時，可以改成用TFIDF，但是似乎起到反效果，MAP只有0.576(topic=128, iter=100, a=0.7, b=0.3, with TFIDF)；第二個是在個文章計算TF時事先濾掉一些stop words，我使用nltk的stopwords進行過濾之後，比原始的dictionary少了約100個字，以這樣的狀況得到了改善的效果，MAP上升為**0.595 (topic=128, iter=100, a=0.7, b=0.3, with TF, filtered stopwords)**，這即是我最終的public分數。