Information Retrieval and Extraction Term Project1

一、Team member and Division work：

湯忠憲	資料科學碩一	R06946003	BM25 model
陳奎伯	資工碩一	P06922001	LDA+Language model
劉宏國	資工碩一	R06922006	VSM+W2V model

二、Model discussion：

A. Model – BM25 model

1. Introduction：

在這次 project 的 paper 中我們可以看到它使用 BM25 作為 base line model，並計算 MAP 和 nDCG。BM25 屬於 bag-of-words 模型，bag-of-words 模型只考慮 document 中詞頻，不考慮句子結構或者語法關係之類，把 document 當做裝 words 的袋子，具體袋子裡面可以是雜亂無章的[1]。此外，BM25 可以提供較好的 term weighting，同時考量到 inverse document frequency, term frequency 和 document length normalization 的計算。而 BM25 是一種 probabilistic model 的變型，演化的過程中還經過 BM1、BM11 和 BM15 的調適，最後推導出 BM25 的公式。

2. Methodology：

BM25 的 Ranking function:

$$sim_{BM25}(d_j, q) \sim \sum_{k_i[q,d_j]} \mathcal{B}_{i,j} \times \log\left(\frac{N - n_i + 0.5}{n_i + 0.5}\right)$$

其中，

$$\mathcal{B}_{i,j} = \frac{(K_1 + 1) f_{i,j}}{K_1 \left[(1 - b) + b\frac{len(d_j)}{avg\_doclen}\right] + f_{i,j}}$$

K_1和 b 是常數，$f_{i,j}$是 term k_i出現在 document d_j的頻率。

$avg\ doclen$是 corpus 中的文件平均長度。n_i是包含 term k_i的文件個數。

3. Evaluation：

依照 paper 中 table 2 中的評估指標，我們使用 MAP (Mean Average Precision) 和 nDCG@10 (Normalized discounted cumulative gain)。

MAP：The mean of the average precision scores for each query.[2]

---

[1] http://blog.csdn.net/heiyeshuwu/article/details/55669316

[2] project1 PPT

nDCG：The nDCG values for all queries can be averaged to obtain a measure of the average performance of a ranking algorithm.

4. Discussion：

1. Performance: 使用 BM25 Entity retrieval 的 MAP 大約為 0.1764 (Paper MAP: 0.1893)，nDCG@10 為 0.2087 (Paper nDCG: 0.2558)。Performance 的差異可能是因為資料集的大小差異，使得兩個機率模型得到不一樣的結果，也可能是在做資料清理時的處理方式不同導致的。**若將 query 都小寫化，並且把 When, Where, What 等疑問詞都捨去**，可以得到更好的表現 **(MAP: 0.2946, nDCG: 0.3352)**。
2. 不同K_1和 b 的表現：從下表的實驗結果我們可以發現其實 fine tune 參數並不能得到很顯著的效果提升，但彼此之間仍然有些微差異。可以發現較小的K_1值得到的 MAP 也較高。

K_1	b	MAP	nDCG@10
2	0.75	0.2875	0.3278
1.5	0.75	0.2923	0.3341
1.5	0.5	0.2895	0.3351
1.2	0.75	0.2937	0.3349
1.0	0.75	0.2946	0.3352

B. Model – LDA + Language Model

1. Introduction：

LDA (Latent Dirichlet allocation)，是一種主題模型，它可以將文檔視為多個主題所成的集合，並分別按照機率分布的形式給出。同時它是一種無監督學習算法，在訓練時不需要手工標註的訓練集，需要的僅僅是文檔集以及指定整體主題的數量即可。此外 LDA 的另一個優點則是，對於每一個主題均可找出一些詞語來描述它。我們可以利用 LDA 優化 Language Model，也就是除了考慮原始文檔的詞頻機率分布外，再參考主題層次的關聯度，以利提供相關性更準確的 Information Retrieval 服務。

2. Methodology：

排序 Query 與所有文檔相關性的評估分為三個部分：

(1) 考慮每個 term 出現於 document 的次數所占 document 文長的比例。

(2) 考慮每個 term 出現於 collection 的次數總和所占 collection 總文長的比例。

(3) 考慮每個 term 在 LDA 主題模型中與 document 的相關機率總和。

上述第(1)、(2)項是傳統 Language Model 的詞頻統計。
加上第(3)項的做法是：決定整體主題數量後，透過 LDA 演算法對每篇文檔隨機產生各主題的機率分佈，再從主題對應的詞語機率分佈中隨機選擇一個詞，最後我們可以得到兩個機率分佈矩陣 document-topic matrix 以及 topic-term matrix，將它們相乘求得的矩陣 M 便是 document-term 的相關機率，其中 $M_{i,j}$ 的意義是 $document_i$ 和 $term_j$ 在所有 topics 中相應的機率總和。

我們把 (1)、(2)、(3) 對應的評分結果稱為 p1、p2、p3，排序公式設計如下：

$$\mathbf{Scores} = \boldsymbol{\lambda 1} * (\boldsymbol{\lambda 2} * \mathbf{p1} \ + \ (\mathbf{1} - \boldsymbol{\lambda 2}) * \mathbf{p2}) \ + \ (\mathbf{1} - \boldsymbol{\lambda 1}) * \mathbf{p3}$$

λ1 是考慮 (1)+(2)、(3) 項相對重要性的 Smoothing Parameter
λ2 是考慮 (1)、(2) 項相對重要性的 Smoothing Parameter
對於 Query 中的各個 term，透過公式求得 Score 加總越高的文檔排序越往前。

3. Evaluation：

(1) nDCG (Normalized Discounted Cumulative Gain)
(2) MAP (Mean Average Precision)
本模型效能數據如下表 (並與這次 project paper 的 SDM、BM25F-CA 做比較)

	LDA+LM (本模型)	SDM	BM25F-CA
nDCG@5	0.6608		
nDCG@10	0.6391	0.4185	0.4387
MAP	0.5808	0.3259	0.4185

```
kpchen@Kadabra:~/IR/trec_eval$ ./trec_eval -m ndcg_cut qrels-v2.txt result.txt
ndcg_cut_5              all     0.6608
ndcg_cut_10             all     0.6391
ndcg_cut_15             all     0.5641
ndcg_cut_20             all     0.5357
ndcg_cut_30             all     0.5353
ndcg_cut_100            all     0.6893
ndcg_cut_200            all     0.7519
ndcg_cut_500            all     0.7519
ndcg_cut_1000           all     0.7621
kpchen@Kadabra:~/IR/trec_eval$ ./trec_eval -m map qrels-v2.txt result.txt
map                     all     0.5808
```

4. Discussion：

(1) 在排序公式中 λ1、λ2 的調整過程中發現，若對於不同 Query categories 採用不同的策略得到的整體效能較好：因為 SemSearch_ES 較適合採用單純 LM 做預測，其餘較適合採用 LDA；另外，ListSearch 傾向較多考慮該文檔中的 Query 詞頻來做相關性預測。以下為本模型使用的參數：

Query categories	λ1	λ2
SemSearch_ES	0.99	0.01
INEX-LD	0.01	0.01
QALD2	0.01	0.01
ListSearch	0.01	0.99

附上本模型與 (λ1, λ2) 調整為 (0.5, 0.5) 的模型效能比較：

	LDA+LM (本模型)	0.5*LDA+0.5*LM
nDCG@10	0.6391	0.6357
MAP	0.5808	0.4255

(2) 在 LDA 部分預先決定的「整體主題數量」參數調整過程中，發現設定為 200 會有最佳效能，由此可知這個值非常接近 DBdoc 中所有文檔隱含的主題總數。

C. Model – VSM+W2V model

1. Introduction：

由最基本的 VSM model 及 NMF model 實作開始，並且再個別結合 w2v，比較出 MAP 及 nDCG score 較佳的模型，其中 w2v 方面也有測試是否要用 per-train data(Glove)或是是否考慮 tf-idf 權重等因素來改善準確率。

2. Methodology：

VSM：documents 及 query 的個別 vectory 會先運算 tf-idf 權重，最後將 doc vector 和 query vector 透過 cosine similarity 得到和 query 相似程度較高之 documents。

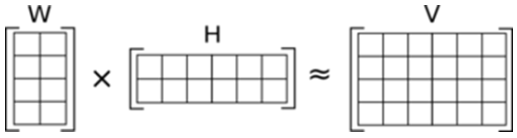
NMF：基本方法即是將一個非負矩陣分解成另外兩個非負矩陣(陣內的元素(element)都是大於或等於零的實數)，前矩陣近似為後兩個矩陣的乘積。若以數學描述，即為任一尺寸為$N \times M$的非負矩陣，透過 NMF 分析，可求得兩個非負矩陣$\mathbf{W}$和$\mathbf{H}$。(Ranking function：cosine similarity)

$$V \cong \mathbf{WH}$$

(1) 矩陣$\mathbf{V}$通常稱為資料矩陣(data matrix)，Ex：Doc – Term 矩陣 m × n tf-idf 值；m：documents、n：terms。

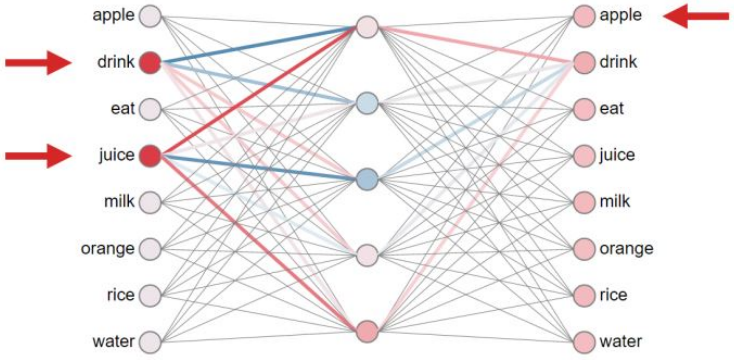
(2) $\mathbf{W}$通常稱為基底矩陣(Basis matrix)，尺寸為 m × f，其中 f 為基底個數。由式 V≅ $\mathbf{WH}$可看出，資料矩陣$\mathbf{V}$的每一個行向量可近似為基底矩陣$\mathbf{W}$的所有行向量之線性組合。

(3) **H** 稱為編碼矩陣(Encoding matrix)，尺寸為 f × n，其每一行代表了上述之線性組合的係數，詳細來說，當選擇基底(座標軸)為矩陣$\mathbf{W}$的行向量時，樣本矩陣$\mathbf{V}$的任何第j個行向量(即第j筆資料)所對應的編碼(座標系數)為矩陣$\mathbf{H}$的第j行向量。



w2v：w2v 第一種是透過 Continuous Bag Of Words（CBOW），利用上下文的詞來當作神經網路的輸入，最後預測這個目標的詞是什麼。第二種則是 Skip-Gram 演算法，剛好跟第一種演算法相反，他所輸入的是當前的詞來預測這一段的文章上下文的詞。如果這兩個向量的上下文被判定為相似的，那 word2vec 會調整向量裡的數值來讓彼此在向量空間裡的距離拉近，而反之則會把他拉遠。(Ranking function：cosine similarity)。

$$p(w_t \mid w_{t-m}, \cdots w_{t-1}, w_{t+1}, \cdots, w_{t+m})$$

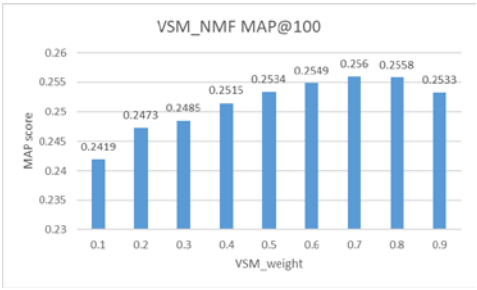


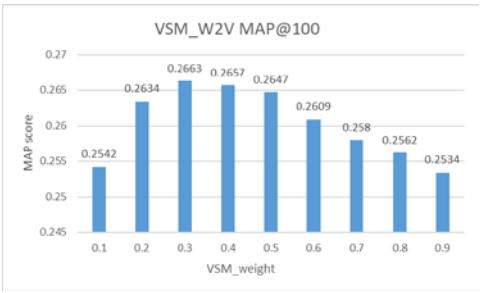
https://www.etusolution.com/index.php/tw/news/blog/97-blog/technical-point-of-view/632-word2vec、ADLxMLDS Lecture 4: Word Embeddings

3. Evaluation：

同 BM25。

4. Discussion：

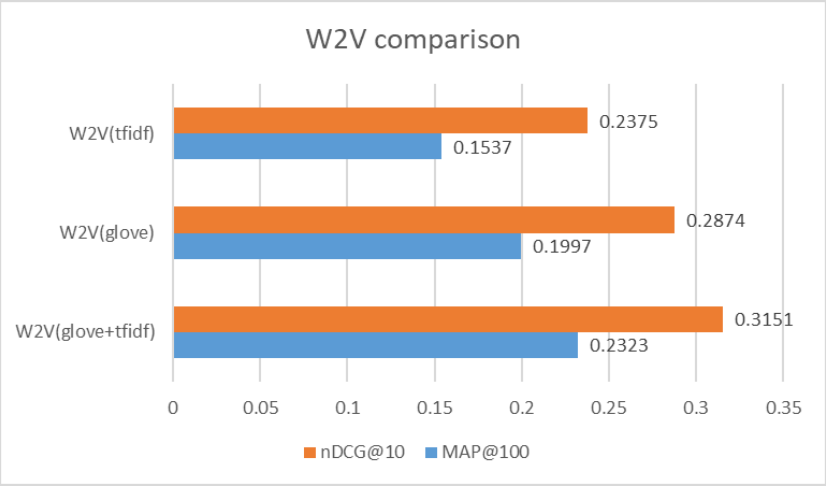




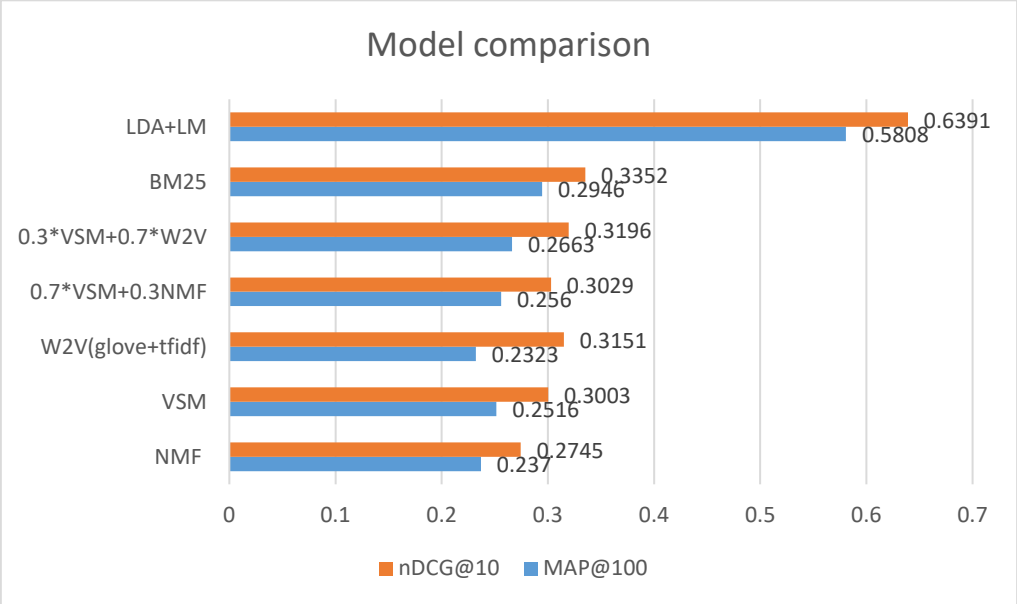
透過線性組合，將 VSM、NMF 及 w2v 算出來的分數個別乘上一個權重，得到最後的分數再去排序找相似的 documents。

其中以 VSM weight=0.7，NMF weight=0.3，VSM 結合 NMF 的效果較佳。而 VSM weight=0.3，w2v weight=0.7，VSM 結合 w2v 較佳。

以 w2v 來說，w2v 利用 Glove 的 pre-train data 和透過 tf-idf 計算權重的準確率較佳(下圖所示)，因此再結合 VSM 的方式是以上述的 w2v 進行結合。由 Conclusion 的圖所示，VSM 和 w2v 結合後的準確率有比個別的準確率提升一點。



三、Conclusion：



Paper[3]中的 methods comparison 主要分為傳統方法以及需要用到 DBpedia-entity v1 的 supervised methods (bottom block)。我們主要嘗試了不需要額外 training data 的方法，包含 BM25 和 Language model，並且另外嘗試了 Vector Space Model 及其變型。從上圖我們可以比較出各個模型之間的準確度。Probabilistic based model 如 Language Model 和 BM25 有較高的分數，然後是 Vector space model。經驗結果我們可以發現如老師 PPT 裡面提到的，VSM 雖然提供了一個比較新穎的想法，但實際效能並沒有特別的突出。

[3] DBpedia-Entity v2: A Test Collection for Entity Search (http://hasibi.com/files/sigir2017-dbpedia_entity.pdf)