

Group 13 Final Project Report

洪明邑 D09948005、陳若曦 B06902079、黃禹翔 R09946004

1. Background and motivation

在這一次的Final Project中，我們的目標為根據給定的Query，在全部的文件中挑選出與之相關的文件。在實際過程中，我們先將 Query 和 Document 都進行 Data Preprocessing 清除雜訊後，嘗試一些不同的Information Retrieval 演算法，以檢索和各個Query關聯程度最高的相關文件。

2. Dataset and Preprocessing

Dataset

資料集總共有100,000筆病例文件，而每一筆病例文件中都有摘要(Abstract)和詳細內容(Body)兩個部分。而在30筆Query裡頭，則可以分為15筆已經有相關的病例文件記錄的Training Query和15筆沒有的Testing Query，我們的目標便是針對這些Testing Query，各自找出與其描述最相關的病例文件，在每個Query中也都包含了電子健康記錄中的入院病歷(Note)，更為精簡的描述(Description)，以及對於描述的摘要(Summary)這三個區塊。

Preprocessing

為了更好的進行Information Retrieval的工作，在實際將資料放入IR System搜尋以前，我們對資料進行Preprocessing，以除去一些會令搜尋結果變差的干擾，以下則是一些我們用來Preprocessing的方式。

- **Noise Removal**: 移除資料中的HTML tag、網址、括號及標點符號等。
- **Text Normalization**: 全部字母統一以小寫的方式表示、移除資料中的數字，並將資料裡頭所有的英文縮寫復原為完整的形式，例如把常見的“don't”復原為“do not”，最後再使用NLTK套件中的功能拆解出現的每一個單詞。
- **Final Preprocess**: 在這個部分，我們嘗試了幾種不同的方法，包含移除停用字(Stop Word Removal), 詞幹提取(Stemming, 抽取字詞的詞幹)，以及詞形還原(Lemmatization, 將字詞還原成其一般形式)等，不過在實際測試時，我們發現移除停用字以及長度太短(小於3)的字有最好的效果。

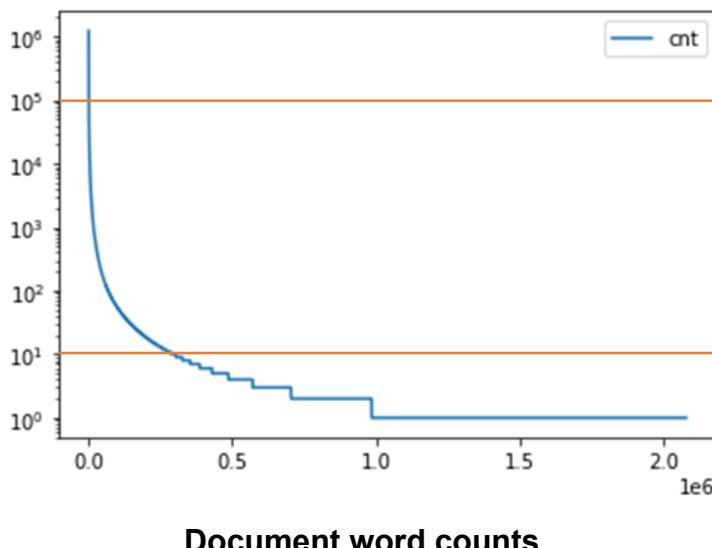
- **Medical NER**: 除了上述的方法之外，我們也嘗試 Allen Institute for AI 利用 SciBERT 訓練出的 Name Entity Recognition 模型 [1]來協助前處理，希望能針對醫療相關名詞進行檢索。由於深度模型的長度限制與運算資源的不足，我們只使用此模型擷取 Query 的 Description 和文件的 Abstract 中關於醫療相關的名詞。但由於相較於前述做法，Medical NER 對後續的模型助益並不大，因此在後續的實驗中並未一貫採用。
- 最後我們統計所有字詞在資料中的出現數量，並移除那些在資料中過於常見的字詞或是太過罕見的字詞。另外，因為資料的數目龐大，為了加速資料處理，我們實作了 Multiprocessing。我們將全部的文件以及 Query 都做了上述的 Preprocessing。在實際的嘗試過程中，我們發現 Query 裡頭 Note 欄位的資料相對雜亂，因此 Query 最後只使用了 Description 和 Summary 兩個欄位。

3. Methodology

在此專題當中，我們嘗試使用多種傳統IR模型如：TF-IDF、BM25、PL2等，同時使用query expansion協助檢索，並考量不同模型 ensemble 的結果。同時，我們也嘗試各項機器學習模型對初步檢索的結果重新排序，並探討不同參數的設定下的效能，如：隨機森林、Multi-Layer Perceptron、SentenceBERT。

TF-IDF、BM25、PL2

首先單純使用預設參數的TF_IDF、BM25、PL2 做IR，我們使用 PyTerrier [2] 作為我們實作的平台。在嘗試萃取的過程中發現，引入使用太過頻繁的文字和太稀有的詞彙容易造成模型效能的低落，因此我們設定不同的詞彙頻率上下界，try and error 找出合適的區段，最後頻率10-100000為我們嘗試後最合適的區段。



經過以上的處理，我們挑選出其中表現最好的模型PL2上傳到Kaggle，可以在 Public leaderboard 上取得0.09左右的成績。

Model Performance on Training Data Set

name	map	P_50	recall_50
BR(TF_IDF)	0.17380	0.22800	0.23972
BR(BM25)	0.18201	0.23467	0.24951
BR(PL2)	0.20402	0.26000	0.27257

TF-IDF、BM25、PL2 with Query Expansion

在IR發展的歷程中, query expansion 可以說是相當重要的技術。Wiki 上面的定義: Query expansion (QE) is the process of reformulating a given query to improve retrieval performance in information retrieval operations, particularly in the context of query understanding. [3] 我們可以藉由IR系統找到的相關重要文件的資訊再次加入query之中做到query expansion, 以提升最終IR系統的效能。我們將各種模型加上Query Expansion的流程, 實驗結果發現可以顯著的提升檢索效能的, 其中還是PL2有最好的表現, 在Public leaderboard上取得 0.176 左右的成績。

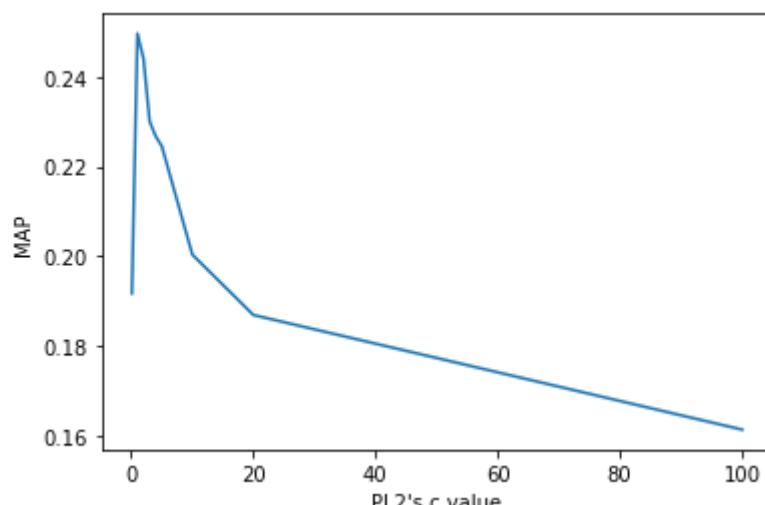
**Model Performance on Training Data Set
(with Query Expansion)**

name	map	P_50	recall_50
BR(TF_IDF)	0.21247	0.26400	0.29769
BR(BM25)	0.21133	0.26000	0.29415
BR(PL2)	0.24958	0.29333	0.30769

TF-IDF、BM25、PL2 with Query Expansion and Fine-tune Parameters

除了加入了Query Expansion顯著提升效能, 也利用training dataset進行模型參數的優化, fine tune PL2及BM25模型的重要參數。

PL2模型針對參數 c (the term frequency normalisation parameter value) [4] 進行grid search, 我們發現當 $c=1$ (Default setting)的時候map score有最大值。



Fine-tune PL2 parameter c based on the training dataset

BM25模型有三個參數可以調整(c , k_1 , k_3)，我們同樣使用grid-search進行參數的尋找，`{"c" : [0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1], "k1": [0.3, 0.6, 0.9, 1.2, 1.4, 1.6, 2], "k3": [0.5, 2, 4, 6, 8, 10, 12, 14, 20]}`，最後找出最佳的參數組合為`['c=0.5', 'k1=2', 'k3=2']`。BM25模型在調參之後效能有一定幅度的進步，但沒有超越PL2。

**Model Performance on Training Data Set
(Fine-tune Parameters)**

name	map	P_50	recall_50
BR(BM25)	0.23851	0.26800	0.28344
BR(PL2)	0.24958	0.29333	0.30769

Ensemble Models

Ensemble是機器學習中常使用的的技術之一，一般做法為藉由投票或平均合併多個種模型的結果，以達到提升最終效能。在此我們嘗試使用簡單平均將BM25、PL2、DPH三種模型做ensemble，在training data set中有小幅度的提升，在Public leaderboard上拿到0.17的成績。

**Model Performance on Training Data Set
(Ensemble Model)**

name	map	P_50	recall_50
BR(BM25)	0.23998	0.29333	0.29920
BR(PL2)	0.25224	0.29867	0.30598
BR(DPH)	0.24165	0.29467	0.30128
Ensemble	0.25502	0.29733	0.30639

Learning to Rank

在此將先前訓練好的IR模型分數正規化之後輸入機器學習模型進行Learning to Rank的訓練 [5]。

Random Forest

將TF-IDF、BM25、PL2、DPH及搭配 Query Expansion的TF-IDF、BM25、PL2、DPH共八種模型輸出的Score，輸入Random Forest Model，嘗試使用Learning to Rank 的技術提升檢索的結果。在training data set中，經過RF模型之後各項分數皆有顯著的提升，我們更進一步將結果上傳到kaggle，很可惜public leaderboard 的成績卻下降到了0.114，我們推測遇到了嚴重的overfitting 的問題，我們嘗試調整tree數量及深度以減輕overfitting，很可惜沒有在顯著的效果。

**Model Performance on Training Data Set
(RF Learning to Rank Model)**

name	map	P_50	recall_50
BR(TF_IDF)	0.21379	0.26800	0.29217
BR(BM25)	0.23998	0.29333	0.29920
BR(PL2)	0.25224	0.29867	0.30598
BR(DPH)	0.24165	0.29467	0.30128
RF	0.26466	0.34133	0.35764

Neural network(MLP)

我們也將八種模型輸出的 Score, 輸入 Multilayer Perceptron, 嘗試使用 Learning to Rank 的技術提升檢索的結果。我們將training data set分成兩等分作為training及validation set, 經過調整Layer數、Node數、Learning rate等參數, 使用validation date set挑選出合適的參數組合為: learning rate=0.0001, max iteration = 1000, one hidden_layer with 200 nodes, validation fraction = 0.5, activation = 'relu', batch_size = 1, early_stopping with 10 iterations no change。

**Model Performance on Training Data Set
(MLP Learning to Rank Model)**

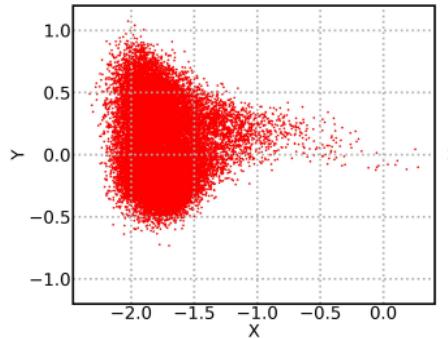
name	map	P_50	recall_50
BR(TF_IDF)	0.21379	0.26800	0.29217
BR(BM25)	0.23998	0.29333	0.29920
BR(PL2)	0.25224	0.29867	0.30598
BR(DPH)	0.24165	0.29467	0.30128
MLP	0.25369	0.30000	0.31309

MLP使training data set的分數顯著的提升, public leaderboard上也獲得0.181的成績, 相較於原本的 PL2 有些許的提升, 這也是我們最終選定的模型。

Siamese BERT-Networks

在上述的方法之外, 我們也嘗試了以孿生架構訓練BERT, 原因有二:

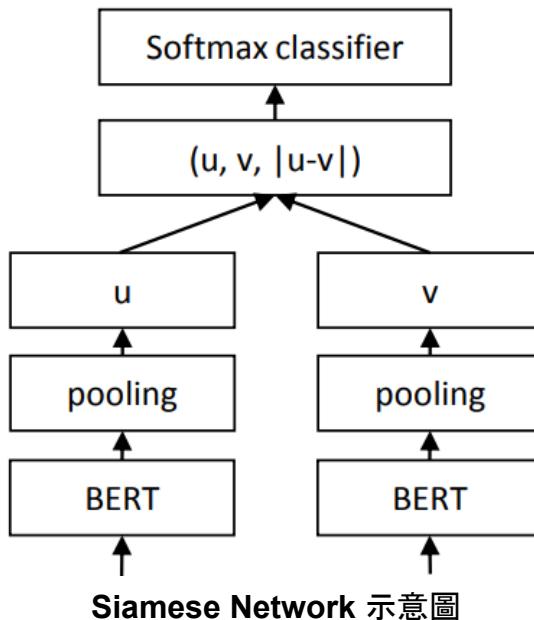
一、根據 Reimers, N., & Gurevych, I. (2019) [6], 使用此種結構有助於避免直接使用一般 BERT 模型取得的文章向量 ([CLS] 的向量或是平均所有 token 得到的向量)時, 語意空間中各向量容易密集佔據一個圓錐的情形 (如下圖所示)。



from Gao, Jun, et al.(2019)

二、由於此次訓練集數量極少且文章極長，比起針對這些少量的樣本以傳統的 finetune 方式訓練 BERT(將 Query 和文章相接在一起放入模型訓練，預測是否應匹配)，也許嘗試針對文章的 embedding 訓練，並使用訓練後模型針對文章取得的向量，利用內積對 BM25 等方法取得的初步文章重新排序會是一個不錯的方法。訓練原理如下：

1. 針對訓練集中配對的 query 與文件，嘗試拉近兩者 embedding 的距離。
 2. 針對訓練集中其他 query 配對到的文件，嘗試拉遠兩者 embedding 的距離。
- 此原理由 python sentence-transformers 套件中的 Multiple Negatives Ranking Loss 實現。損失函數細節見 (Henderson, Matthew, et al.(2017)) 中 4.4 節 [7]。模型訓練完畢後，用以針對各 query 再前述 PL2 搭配 query expansion 模型回傳的前 100 名結果重新排序，然而表現並不理想。訓練集的分數不增反減，因此最後我們並沒有採用這個做法。其中，BERT 預訓練模型我們採用 Allen AI 機構的 SciBERT，SciBERT 預訓練資料集中包含大量醫療相關文本，我們期望透過預訓練的過程來彌補訓練資料的不足。



4. Result and Conclusion

在實作過程中觀察到，如果不使用 Preprocessing、query expansion 等技術處理，直接使用原始資料進行檢索，BM25 檢索的結果可以發現包含以下問題：

- 1) 病人背景類似，但患處及接受診治方式不同。如：皆是年長婦女、一個胃痛一個腰痛。由於"年長婦女"一詞在所有文件中出現頻率不高，BM25 容易給其高分，導致檢索到不相關的結果。
- 2) 病人背景不同，但患處或病因類似。如：小男孩與老人皆右大腿骨折。
- 3) 檢索到類似口吻書寫的文件。如：兩篇文章都用"患者的醫療歷史"來指涉患者過去的醫療紀錄。"歷史"一詞在文件中不常出現，因此在BM25的公式中取得高分。
- 4) 指涉同一種病症或病人，然而用詞習慣不同，導致 BM25 檢索到類似的結果。如：elderly female 和 89-year-old woman 其實是類似的意思，但兩者的 BM25 分數為 0；又如同個病症、手術，query 以醫療縮寫表示，文件以全名表示，都會造成 BM25 難以檢索到正確的答案。
- 5) 文章的長度也使得目前熱門的深度模型較難以完整利用。

此外，訓練資料標住的品質也是一大問題。例如訓練集中的 query 2 與文件 3459798 皆是在年長者關節成形手術相關的紀錄，且病人患發部位相同，但卻不被包含在正確答案中。由於這一次任務的應用場景包含大量不易閱讀的文件，因此不排除標註資料本身的召回率就不高的情況。

後期我們再運用 Learning to Rank 及深度學習相關的技術時，發現遇到了 training data set sample size 太小的問題，造成 re-rank 的時候很容易遇到 overfitting 的問題，我們在訓練 MLP 時刻意將 validation 的比例調高以嘗試找到較為穩定的模型參數。

以下為本團隊上傳檢索結果至 Public/Private leaderboard 重要模型之分數

Method	Public Score	Private Score
TF-IDF	0.04486	0.07904
BM25	0.05754	0.09506
Fine-tune BM25 + QE	0.16395	0.21340
Fine-tune PL2 + QE	0.17671	0.22579
Linear Ensemble	0.17011	0.22040
Learn2Rank - MLP	0.18178	0.22457
Learn2Rank - LR	0.17157	0.22036

Learn2Rank - RF	0.11415	0.15692
PL2 + SBERT telescoping	0.13062	0.15392
Medical NER + BM25	0.01292	0.02882

Future work

目前對於長文章的檢索有 doc2query 的作法：給定一篇文章，利用語言模型生成可能的 query 或是這篇文章可能可以回答的問題，再將其新增到既有的文章中來幫助檢索。較著名的模型有 docTTTTTquery (Nogueira et al. (2019)) [8]，利用 T5 模型來達成此一目的。礙於時間限制，未能實作完整此一模型來評估效果好壞。

5. Each member's responsibility

Name	Responsible For
洪明邑 D09948005	Data preprocessing, Modeling, Report
陳若曦 B06902079	Report
黃禹翔 R09946004	Medical NER, Modeling (SBERT, BM25), Report

6. Reference

1. Neumann, Mark, et al. "Scispacy: Fast and robust models for biomedical natural language processing." arXiv preprint arXiv:1902.07669 (2019).
2. Craig Macdonald, Nicola Tonello, Sean MacAvaney, and Iadh Ounis. 2021. PyTerrier: Declarative Experimentation in Python from BM25 to Dense Retrieval. Proceedings of the 30th ACM International Conference on Information & Knowledge Management. Association for Computing Machinery, New York, NY, USA, 4526–4533.

DOI:<https://doi.org/10.1145/3459637.3482013>

<https://pyterrier.readthedocs.io/en/latest/index.html>

3. Wikipedia contributors. (2021, June 27). Query expansion. In *Wikipedia, The Free Encyclopedia*. Retrieved 05:29, January 12, 2022, from https://en.wikipedia.org/w/index.php?title=Query_expansion&oldid=1030725877
4. Gianni Amati and C.J. van Rijsbergen. 2002. Probabilistic models of information retrieval based on measuring the divergence from randomness. ACM Transactions on Information Systems (TOIS) 20, 4 (2002), 357–389 <http://terrier.org/docs/v4.2/javadoc/org/terrier/matching/models/PL2.html>
5. <https://pyterrier.readthedocs.io/en/latest/ltr.html>
6. Reimers, Nils, and Iryna Gurevych. "Sentence-bert: Sentence embeddings using siamese bert-networks." arXiv preprint arXiv:1908.10084 (2019).
7. Henderson, Matthew, et al. "Efficient natural language response suggestion for smart reply." arXiv preprint arXiv:1705.00652 (2017).
8. Nogueira, Rodrigo, Jimmy Lin, and A. I. Epistemic. "From doc2query to docTTTTTquery." Online preprint (2019).