GENERATIVE QUERY EXPANSION

CS410 : QUERY EXPANSION THROUGH DOCUMENT VECTORS AND WORD EMBEDDING

BYUNG IL CHOI
DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN
(EMAIL: BICHOI2@ILLINOIS.EDU, CHOI@BYUNG.ORG)

ABSTRACT. One way of looking at querying is to find similar documents to a query, rather than finding an exact answer to a question in general. By looking at queries as micro-documents, we can assume that key words or topics of a query can be generated from significant vocabularies of the language model of documents that the user is looking for. Matching a query vector to a document vector can be done by expanding query vector by the document vector which we want them to be searched. The dimensions of both vectors, however, are different. This difference can be filled by training a document vector that corresponds to the query vector. A supervised training would work best in this case, moreover this method can be extended to a semi-supervised method.

KEYWORDS

Query Expansion, Text Retrieval, Generative, BM25

INTRODUCTION

The act of querying is important to fetch relevant documents. But, most queries don't contain enough information to do the task under the assumption of bag of words for searching exact matching terms for ranking; which is the most naive searching without any kind of reinforcement. Because, a questions or query is not qualitatively consistent[1], and the information or the text that the query is looking for may not contain words in the query[2].

A query needs an expansion for a better text retrieval, but how? By starting simple training and evaluation as supervised learning, a search system can train good relevant document vectors. Moreover, the system can expand its query vector with a trained document vector in its searching process or in an evaluation process that comes after the document vector training. This is similar to the word distribution repsentation of a topic. With assumption that knowing which words discribes a query can help the document searching task, it's predicatable that the accumulation of good relevant document vector training will increase the average precision, which is what the generative query expansion[3] will do.

This relevantly simple supervised learning process could be easily applied to the query expansion by simply adding their vectors. And, this possibly can be the analogical model of web search engine. The difference is that web search engine gets feedback from its user, but this process assumes that there is a good document that represents the relevant document for a query, which should be a label in an ordinary supervised training process. Similar to most learning tasks, the growth of improvement will slow down as training goes or as there are more accumulation of trained documents.

[1] non-consistent will mean that the query may choose vague or less relevant terms for searching

[2] one good example is from cranfield data set which will be explained further later.

[3] temporary term which this paper implmenets for the query expansions

RELATED WORK

This expansion method is derived from Rocchio Feedback, Topic Mining, and typical supervised learning process. Rocchio Feedback moves the query to the centroid of relevant documents as we can see from the below figure,

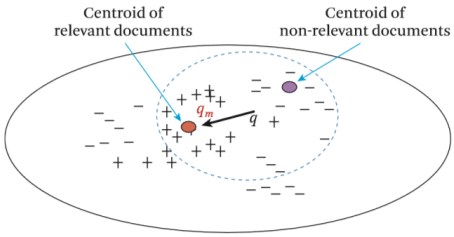


Figure 1. Rocchio Feedback's vector space and the query movement[4]. where the formula[5] is defined as:

$$\overrightarrow{q}_m = \alpha \cdot \overrightarrow{q} + \frac{\beta}{|D_r|} \cdot \sum_{\overrightarrow{d}_j \in D_r} \overrightarrow{d}_j - \frac{\gamma}{|D_n|} \cdot \sum_{\overrightarrow{d}_j \in D_n} \overrightarrow{d}_j$$

where $\overrightarrow{q}$ is the origina query vector, $\overrightarrow{q}_m$ is the expanded vector, D_r is the relevent documents, and D_n is the set of non-relevant feedback documents (Zhai and Massung 136).

Instead of getting a centroid, generative query expansion uses generative model that is from the language models of relevant documents, which are from the labeled training set or from feedback as follows:

$$\overrightarrow{d} = \underset{w_1 \cdots w_n}{argmax} \prod_{w_1 \cdots w_n \in D_r} P(w_i) \qquad \text{<eq 1>}$$

where $\overrightarrow{d}$ is the trained document vector that will be used for the query expansion, D_r is the relevant document vector treated as one document, and n is the parameter for the dimensionality of a document vector $\overrightarrow{d}$.

For checking another possiblity of expanding a query vector, word embedding was also considered. For testing, word vectors were trained in a corpus level using GloVe[6], and expanded the query by most similar words. The results, however, were barely noticeable. MAP[7] was improved about 1% for testing on cranfield dataset, and in worst case, MAP was decreased[8]. Thereby, additional query expansion from word embedding is excluded for this project.

For evaluatio purpose, documents are ranked with BM25/Okapi(Zhai 108)[9] scoring function which is defined as follows:

$$f(q,d) = \sum_{w \in q \cap d} c(w,q) \frac{(k+1)c(w,d)}{c(w,d) + k(1 - b + b\frac{|d|}{avdl})} \cdot log\frac{M+1}{df(w)} \quad b \in [0,1],\ k \in [0, +\infty)$$

BM25/okapi is the state of the art, thus was adapated for the valuation; which is good for both short corpus and long corpus since there is a penalization depending on the length of a document.

PROBLEM DEFINITION

Is MAP of 0.2~0.3 good enough for the searching task? This project's goal is to increase MAP of the text retrieval task; not by locally contextual prediction(such as word embedding), but by learning through the data given for the task. Employing a simple train-and-evaluation model where training is done with supervision will be measured; meaning that a labeled dataset will be used for both training and evaluation.

How will the dimensionality of the document vector would affect the query expansion? Will small dimension do the job? How will extension of their dimensions affect the task of searching? This dimentionality will be parameterized and effects on MAP will be measured as a result.

---

[4] ChengXiang Zhai and Sean Massung, Rocchi Feedback figure from "Text Data Management and Analysis"

[5] ChengXiang Zhai and Sean Massung, Rocchio Feedback formula from "Text Data Management and Analysis"

[6] Jeffrey Pennington, Richard Socher, Christopher D. Manning, GloVe: Global Vectors for Word Representation

[7] Mean Average Precision

[8] for the detail go to the following section

[9] ChengXiang Zhai and Sean Massung, BM25/Okapi formula from "Text Data Management and Analysis"

Cranfield dataset and APNews dataset will be used for both training and evaluation. The documents, queries, and query relevance are all included in the dataset. All of these files will be used as input to the training and the output will be set of document vectors that can be used for the generative query expansion.

METHODS

How query is expanded sorely depends on the data and the language model that the data has for generative model. The query relevance data is used for the alternative to the human labeled data. But, instead of using lots of them, only partial relevances are selected and evaluated with the rest. Unlike the k-crossfold evaluation, which trains k-1 folds and evaluates on the 1-fold, following method trains from n relevant document per query and evaluates on $r-n$ documents where r is the number of relevent documents.

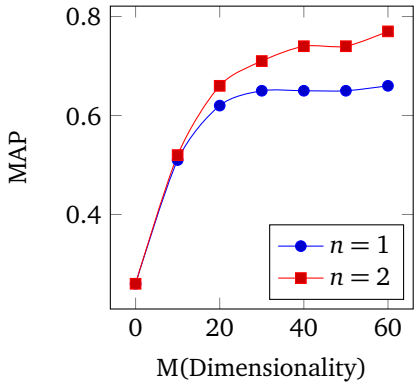
In the evaluation step, MetaPy[10] is used for ranking with the help of BM25/Okapi algorithm. The idea of training is using the language model of target training data by using $\overrightarrow{d} = \underset{w_1 \cdots w_n}{argmax} \prod_{w_1 \cdots w_n \in D_r} P(w_i)$[11] with removal of stop words for D_r where $D_r = \underset{i \leq M}{\cup} D_{r_i}$, M ="number of relevant documents used for the training". The dimension for the document vector is parameterized as n, which will be used to compare the MAP in the evaluation. So, the expected outputs are simple MAP values. This simple two dimensional output will show how precision will grow and will give good evidence of the parameter's effect.

The first attempt of a query expansion depended on the word embedding using GloVe[12]. The word vectors for the vocabularies in the corpus was trained, and found most top-5 similar words for the words in the queries, and used those similar words to expand the queries, which was used in the evaluation. This has a pontetial to work on the big corpus. However, on a small corpus like cranfield, the query expansion with word embedding didn't work well; improvement was less than 1%; in lots of cases, the average precision went down. Hence, the evaluation related to this type of expansion is going to be omitted in the next section.

EVALUATION

To see the status quo, the result without expansion, lets set n=0. Then, there would be no query expansion. Then, the MAP for cranfield set is about 0.269, and 0.286 for APNews dataset. To evaluate, it's crucial to see how MAP improves with different parameters.

For the purpose of improvement, it's important to improve the queries with near-0 average precisions. Let's call these queries hard-queries. Cranfield dataset has about 32 hard-queries. With generative query expansion, we can get much better results for hard queries, which is shown below:



As n become non-zero, MAP starts to increase dramatically. Even with one document training per query, which is equivalent to the case when $n = 1$, MAP is far better than the base case of $n = 0$, which is without the training.

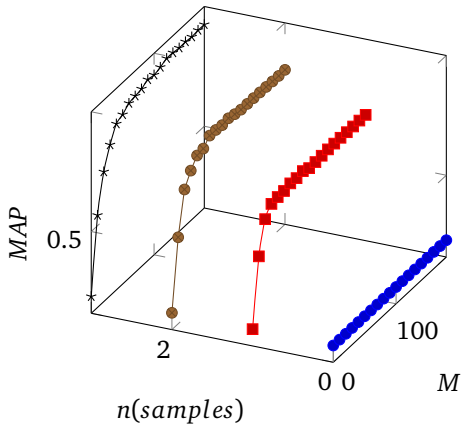
On the other hand, we can see that as M grows MAP also grows, but not monotonically; it converges at a certain dimension(M). This shows that finding proper dimension is also an important task for the expansion.

[10] https://github.com/meta-toolkit/metapy

[11] details at related work

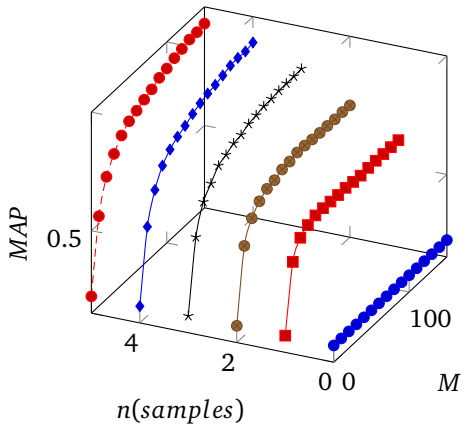
[12] http://nlp.stanford.edu/pubs/glove.pdf

It is more clear to see when we put the evaluation of all hard-queries together as below:



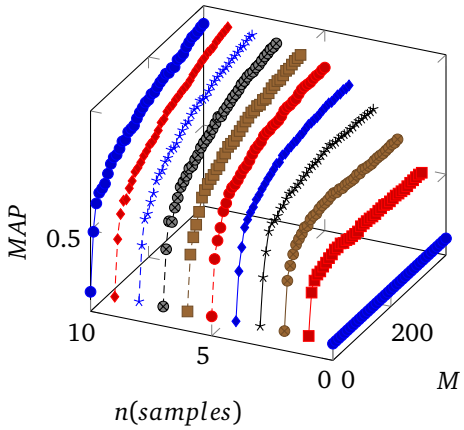
It shows how MAP grows with respect to n and m. Above is the result how hard queries improved over training, which shows good potential to improve the overall query precision. And, when dimentionality is around 50, it starts to converge to its max. This is due to the characteristics of the Cranfield dataset which has reletively short documents compare to APNews.

The overall evaluation for all queries on cranfield dataset is as below:



which shows that as n and m grows, the MAP also grows; higher the better. When n=4, we can see that there is not much of benefit of increasing n. This is because the query relevance and documents are not large enough; cranfield dataset is small. But, if samples are appropriate and large enough it will learn better as we can get from APNews dataset, which has far more query relavance and documents.

Cranfield dataset evaluation was for small dataset. And, APNews dataset is to work on a larger dataset; APNews dataset has average of 100 relevance documents per query. The evaluation result is as follows:



the trend of improvemet over the n and M is similar to the result of cranfield dataset. The result shows that the larger n and M for training gives more benefit. But, what is more interesting is that as M grows larger, where $n > 6$ gives more boost on growth; unlike $n < 5$ where it had more logarithmic growth. This shows that generative query expansion works even better with larger data set with more query relevances.

CONCLUSION

As we can see from the evaluation and its analysis, training from related documents greatly increase the MAP. If the system is trained with appropriate documents or is fed with similar documents as inputs from the user, the generative query expansion will provide a great synergy. This can also be used as a starting cluster of a semi-supervised learning for the query expansion, whenever there is an appropriate feedback or relevant document to the query.

When using generative query expansion, following can be assume to be good pratice. (i) when there are small number of samples, increase dimension of the document vector. (ii) when there are good number of samples for training, it's good to reduce the dimension for the performance. (iii) when there exist a enough computational resource, finding optimized parameters will help both MAP and performance.

FURTHERMORE

For huge corpora, it maybe possible to train appropriate word vectors using word embedding; in a real practical system, the size of its corpora will be hugh; eg. web search engines. The locally trained word vectors may not be useful as we saw from cranfield dataset, but training from large corpus can provide a better chance of finding similar terms, eg. hyponyms, synonyms. It will be interesting to expand the query with word vectors by finding synonyms of most significant[13] words from a trained document vector, and use them as extra dimensions.

ACKNOWLEDGEMENT

MetaPy was heavily utilized for the evaluation of the generative query expansion. GloVe was also utilized for experimental test. Many ideas were conceived from professor ChengXiang Zhai's lectures and his book.

APPENDIX

Single memebered team. Byung Il Choi did all the work with more than 20 hours of work[14].

[13] significant meaning the words have high probability in their language model

[14] This is written as a response to the project guideline of CS410 project.