Finding more relevant documents to optimize the QoS

Jingyan Xu
Shanghai Jiao Tong University
joy_xjy@sjtu.edu.cn

Abstract

This is the class project of EE448. In this article, an algorithm to get the new searching results of the original queries with the expansion terms is illustrated, which makes pseudo-relevance feedback. The words from all the documents pass through several filters, divided into useful or useless terms by support vector machine(SVM) according to the features extracted. After adjusting the weights, the expansion terms are added.
Key words: query expansion, SVM, pseudo-relevance feedback, TF-IDF.

1 Introduction

The original queries input by the users is always ambiguous and simple while the important terms are in absence, which always leads to unwanted documents owning high priority in results. To improve the user experience, a query expansion algorithm is generally used in recommending systems. Pseudo correlation feedback (PRF), also known as blind correlation feedback, provides an automatic local analysis method. Various features are extracted from the potential terms to be added to the query. The performance of each term is the influence it makes in the evaluation principles, which will decide whether it will be put into the train set or not and whether it is a positive term or a negative one. The classifying method in this algorithm is SVM and the label of each data depends on the difference of mean average precision(mAP) in the original query and the updated query with an expansion term. Several filters are used after the classifier to achieve the final terms to add, which are adjusted in proper weights into the query to get the new ranking document list.

1.1 The flow of the complete system

Pseudo correlation feedback (PRF), also known as blind correlation feedback, provides an automatic local analysis method. It can automatically execute the manual part of relevance feedback, enabling users to obtain improved retrieval performance without expanding interaction. PRF conducts a regular search to find the initial set of most relevant documents, then assume that the top k ranked documents are relevant, and finally make relevance feedback as before under this assumption.
The specific process of PRF is as follows,

- Module 1: Get the initial ranking result according to the original query.
- Module 2: Remark Top K documents according to the bm25 algorithm as related documents, finding expansion terms form those docs.
- Module 3: Add decent terms into the query in proper weights.
- Module 4:Get the brand-new ranking result.

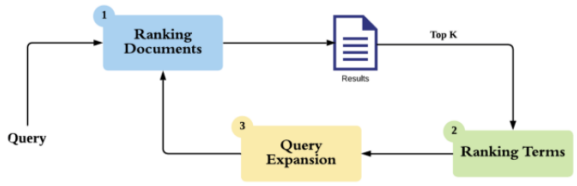

Figure 1: Figure1:The flow of PRF

1.2 The challenges in this algorithm

The dataset is not large enough, including only 200 queries, and each of them has 300 documents with relevant labels works as training. Some of the training data even have to be selected to do the validation, while there are 50 queries to be tested. Therefore, it is very likely to over-fitting in the model while the data are not enough to make a decent classification.

The parameters can be tuned in this algorithm is not that much, but each tuning is time-killing and the effect is not obvious enough to find out the difference directly. For instance, the value of k in finding the Top K documents in module 2 in figure 1 is very important.

The remaining of the paper is organized as follows: Section 2 reviews some related work and the state-of-the-art approaches to query expansion. Section 3 introduces the

dataset and the equipment of this task. In section 4, the specific procedure of this query expansion algorithm is elaborated. Section 5 presents the results of the algorithm and the methods used in improving the result. Section 6 concludes this paper and suggests some future work.

2 Related Work

Query expansion. A variety of methods employ local features[1][2][3] and are well adapted to the Bag-of-Words model[4]. Others are generic and applicable to any global image representation [5][6]. In both cases, ranking is performed on the image level. The text contents can be processed in a similar way.

Pseudo-relevance feedback. PRF has been widely used in image retrieval(IR), which has been applied in different retrieval models: vector space model[7], probabilistic model[8], etc.. In this paper, the PRF principle has also been implemented within the language modeling framework.

The query model. It describes the users' information need is estimated with Maximum Likelihood Estimation(MLE). approaches here: relevance model and mixture model. Considering the relevance labels and Bayesian rule, the simplified model can be estimated as:

$$P(w|\theta_R) \propto \sum_{D \in F} P(w|D)P(Q|D) \tag{1}$$

The probability of a term w in the relevance model is determined by its probability in the feedback documents (i.e. P(w—D)) as well as the correspondence of the latter to the query(i.e. P(Q—D)). The relevance model is used to enhance the original query model by the following interpolation:

$$P(w|\theta_q) = 0.5P(x|\theta_0) + 0.5P(x|\theta_R) \tag{2}$$

The mixed model also attempts to build a language model for the query subject from the feedback document [9], but the method is different from the correlation model. The query subject model to extract is considered as the most distinctive part of the entire document collection.

3 Conditions of the experiment

The dataset to pick the decent expansion term from includes 200 queries and each query has 300 related docs, a document marks the relevance between each query with its 300 docs in 0 or 1.

The size of the documents, queries and relevant mark is 193.3MB, 6kB and 834kB respectively.

The environment of the experiment is 2.3 GHz Quad-Core Intel Core i5, 16 GB 2133 MHz.

4 Adding expansion terms

In this paper, adding expansion terms is the main task. The course of a word from original documents to the expanded query is illustrated as Figure2.

Top K documents are chosen first to work as the first filter

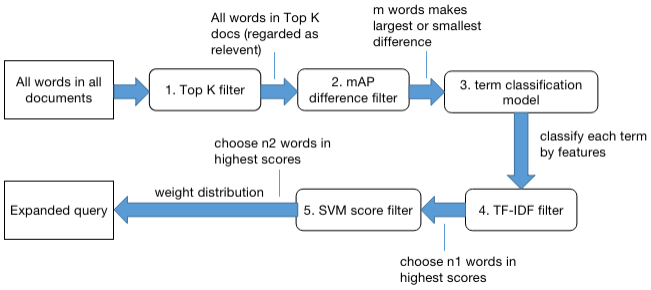

Figure 2: Figure2:The flow to add expansion term

to get the potential terms. Secondly, each word is added into the original query one by one to get the new ranking. The difference of mAP determines whether the term will be added into the training set. The terms cause large differences are chosen as positive or negative terms. After that, the labeled terms will be trained by support vector machine(SVM) with their features. The terms in the test set are processed in a similar way. Another filter for the terms in the test set is the score of TF-IDF, namely the product of text frequency and inverse document frequency. The scores with higher TF-IDF scores will be chosen to make the final filtering. Finally, the score evaluated by SVM and the adjusted weights are considered to get better performances.

4.1 Top K filter

It is impossible to extract features from each word in all the documents. Therefore, some filters are applied. Top **20** documents in the bm25 score are regarded as relevant documents and all the words whose frequency is equal to or more than the baseline, **3** times, are chosen for the later procedures.

4.2 mAP difference filter

The words from filter 1 are added to the original query in the same weight to gain the new bm25 ranking of the 300 documents, and the labels of relevance do help in calculating the mean average precision(mAP) of each term. For each query, the terms make larger differences are labeled 1 and those make larger negative differences are labeled -1 for training.

4.3 term classification model

There are 30 positive terms and 30 negative terms for each query, and there are 200 queries in the training set, which is balanced. This is typical supervised learning, and the validation set is 30% of the total training set. The classifier is input with 10 features extracted from each term and the

output is the label.

Among the ten features, there are five kinds of features obtained from the set of feedback documents and all the documents, so there are actually five features altogether, illustrated as follows[7]:

Feature 1: term distribution

Term distribution means the ratio of the text frequency of the certain term.

$$f_1(e) = log\frac{\sum_{D\in F} TF(e,D)}{\sum_t \sum_{D\in F} TF(t,D)} \tag{3}$$

F is the set of Top K documents in f_1, f_3, f_5, f_7, f_9 and t is all the documents in $f_2, f_4, f_6, f_8, f_{10}$, D is document. The logarithm amplifies the result.

Feature 2: Co-occurrence with single query term

$$f_3(e) = log\frac{1}{n}\sum_{i=1}^{n}\frac{\sum_{D\in F} C(w_i,e|D)}{\sum_t \sum_{D\in F} TF(t,D)} \tag{4}$$

$C(t_i, e|D)$ is the frequency of co-occurrence of each query term w_i with the expansion term e in document D. The co-occurrence is labeled 1 when the distance between w_i and e is equal to or less than the window, 12 words.

Feature 3: Co-occurrence with pairs query terms

$$f_5(e) = log\frac{1}{|\Omega|}\sum_{(w_i,w_j)\in\Omega}\frac{\sum_{D\in F} C(t_i,t_j,e|D)}{\sum_t \sum_{D\in F} TF(t,D)} \tag{5}$$

Similar to Feature 2, the co-occurrence of each pair in the original query term with the expansion term is the numerator with a larger window, 16 words.

Feature 4: Weighted term proximity

$$f_7(e) = log\frac{\sum_{i=1}^{n} C(x_i,e)dist(w_i,e|F}{\sum_{i=1}^{n} C(w_i,e)} \tag{6}$$

The numerator of Feature4 is the product of the co-occurrence of each word in the query with the expansion term and their distance, the minimum number of words between them. The window of $C(w_i, e)$ is 12 words, just like Feature2.

Feature 5: Document frequency for query terms and the expansion term together

$$f_9(e) = log[\sum_{D\in F} I((\Lambda_{t\in q} t \in D)\Lambda e \in D) + 0.5] \tag{7}$$

I(x) is the indicator function whose value is 1 when the Boolean expression x is true, and 0 otherwise. The constant 0.5 here acts as a smoothing factor to avoid zero value.

4.4 TF-IDF filter

The words in the test are chosen from the top 20 documents. Due to the lack of the labels of relevance, the second filter is substituted with the TF-IDF score.

TF-IDF (term frequency is inversely proportional to document frequency) is a statistical measure used to evaluate the relevance of words to documents in a document collection. This can be done by multiplying by two measures: how many times a word appears in the document, and how often the word appears in the reverse document in a group of documents. The higher the score is, the more effective the term is.

The terms with higher scores in TF-IDF are filtered, extracting ten features for further processing.

4.5 SVM score filter

The terms filtered by the TF-IDF score are classified by the trained SVC. To get the ranking of the terms, the output is changed into probability. The terms with higher scores are chosen to add to the original query.

4.6 weight adjustment

The original queries are regarded as the most important. The expanded terms are divided into some stages, the ones with higher scores have higher weights while the lower get lower weights.

$$w_i = \begin{cases} \frac{total}{stage} + 1, i <= len(query) \\ \frac{total}{stage}, i > len(query), total/i > total/stage \\ \frac{total}{i}, otherwise \end{cases} \tag{8}$$

w_i is the weight of the i^{th} term in the new query

5 Methods and results

5.1 Term processing

The choice of terms to be added into the training set decides the performance of the classifier, and the quality of the testing set is also interfered with. Therefore, this procedure could never be paid more attention. The obvious margin between the useful terms and the useless ones does great help in QE. Each modification to extract new features from the new words takes much effort, needs two days to do the computation.

The value of k top documents in the bm25 score, together with the bar of the lowest text frequency makes sense in both training set and testing set. The scale of terms s filtered by TF-IDF also counts in the testing set. The feature of the words makes decent classification when k is 20, the text frequency bar is 3 and the limitation s is 280 words.

The number of labeled data to be added is important as well. To make the training set balanced and representative, from each query, 30 terms makes the largest difference in mAP either positive or negative are selected to feature extraction.

5.2 Feature processing

Some parameters in feature extracting influence the effect of the classifier, such as the window of the co-occurrence. For a single word, the window size can be 12 and for pair words, 16 is a proper size.

The values vary among different features due to the various measurements. Therefore, before training the model, the values of each feature need a normalization. After subtracted the mean value, the values in the same column are divided by the variation to get 0 as the mean value.

5.3 Model training

To avoid over-fitting or under-fitting, the training set is separated into a training set and validation set at 7:3. The output of the model is the probability to give a ranking for each term. The radial-based kernel function (RBF) is used.

$$K(x_i, x_j) = exo[-||x_i - x_j||^2/2\sigma^2] \tag{9}$$

It has relatively fewer hyperparameters and has shown to be effective. Other parameters in SVM are set as follows:

C = 26 and gamma=1.02 to give a larger margin and lower difference between train precision and test precision.

Undoubtedly, the terms' choice interferes the classifier's performance most. The test accuracy is around 70% with the parameters above.

The tuning of the parameters above depends on the physical meaning of C and gamma, they are tried in many intervals until the precision are close and high enough.

5.4 Adding expansion terms

The different weights of each term make sense for more relevant documents to get a higher ranking. When the top 30 terms in SVM score are chosen and the weights are divided into 6 stages, the NDCG of the re-ranked documents improves a lot comparing with the original one. Formula (8) makes most words have only 1 in weight, and give a few words with high weights, ensuring the most important status of the original query to show the respect of the user's needs. The uneven distribution is more realistic, ensuring the terms whose score is not high enough to have their own effects.

6 Conclusion

The model performs well in the public test set, while the result in the private test set are not that satisfying. Though the variance of ranking does not change too much, the difference with baseline shows the unrobust of the algorithm, and the possible reasons and their solutions are listed as follows:

1. The scale of data for training is not enough.

There are only 200 queries for training before validation, while the test data is as much as 50 queries. The training model cannot include enough situations, which may explain a little. In future work, more augmented training data can be applied.

2. The evaluation methods can expand diversity.

The test set in this experiment only involves the classifier's accuracy in distinguishing the term's usefulness. However, for the final ranking, direct evaluation is in lack. Despite the fact that various methods to do the expansion are applied, the lack of more elaborate relevance mark also makes it harder to do the estimation of the training effects.

In future work, when the entire experiment is designed without various limitations, more well-described data are in need to gain objective evaluations, reflecting the quality of the expansions better.

3. The model for classification is not so robust to fit different test sets.

The large difference between the score in two different test sets shows that the model may excel in certain terms, whose features may be classified accurately. However, I can not make sure which of the queries are expanded well. It is unsuitable to expand the same length for each query. In fact, when extracting features from the test data, the values become much sparser than the training data, which makes me suspect that some features are not so representative. For example, the co-occurrence of a word pairs in the original query with the expansion term, it is very likely that there exists no co-occurrence in any of the documents at all. Therefore, the property of the uncertainty of feature engineering decides the unstable results.

In future work, the learning algorithm may be changed into unsupervised learning for generalization, and deep learning may be a better idea for gaining features in higher-dimension which can not be extracted from the above with less effort. The course of designing the features and extracting them is not that easy but not help a lot.

4. The expanded terms should be found from a larger word bag.

As the task is text mining, the properties of language should be taken into consideration. In the online shopping websites, it is likely to see similar products together with the inquired products being recommended which are not mentioned in the original query words. Similarly, in searching for documents, the association between synonyms may also

be applied.

In future work, the synonyms of the original queries can be added which are not likely to appear in the documents, and the keywords of each document and all the documents can be summarized to give labels for the documents as the information just in the documents is not sufficient to make great expansions.

Acknowledgments

Thanks to Weinan Zhang, the teacher of this course, who elaborated a lot of important algorithms in both machine learning and deep learning in class. Thanks to the two TAs, Jiawei Hou and Xinyi Dai, who gave me a lot of help in the coursework.

References

[1] A. Iscen, G. Tolias, Y. Avrithis, T. Furon, and O. Chum, "Efficient diffusion on region manifolds: Recovering small objects with compact cnn representations," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 926–935, 2017.

[2] O. Chum, A. Mikulík, M. Perdoch, and J. Matas, "Total recall ii: Query expansion revisited," in *CVPR 2011*, pp. 889–896, 2011.

[3] O. Chum, J. Philbin, J. Sivic, M. Isard, and A. Zisserman, "Total recall: Automatic query expansion with a generative feature model for object retrieval," in *2007 IEEE 11th International Conference on Computer Vision*, pp. 1–8, 2007.

[4] Sivic and Zisserman, "Video google: a text retrieval approach to object matching in videos," in *Proceedings Ninth IEEE International Conference on Computer Vision*, pp. 1470–1477 vol.2, 2003.

[5] H. Jegou, H. Harzallah, and C. Schmid, "A contextual dissimilarity measure for accurate and efficient image search," in *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2007.

[6] D. Qin, S. Gammeter, L. Bossard, T. Quack, and L. van Gool, "Hello neighbor: Accurate object retrieval with k-reciprocal nearest neighbors," in *CVPR 2011*, pp. 777–784, 2011.

[7] G. Cao, J.-Y. Nie, J. Gao, and S. Robertson, "Selecting good expansion terms for pseudo-relevance feedback," in *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '08, (New York, NY, USA), p. 243–250, Association for Computing Machinery, 2008.

[8] S. E. Robertson and K. S. Jones, "Relevance weighting of search terms," *Journal of the American Society for Information Science*, vol. 27, no. 3, pp. 129–146, 1976.

[9] C. Zhai and J. Lafferty, "Model-based feedback in the kl-divergence retrieval model," *CIKM*, pp. 403–410, 2001a.