BJUT at TREC 2016 OpenSearch Track: Search Ranking Based on Clickthrough Data

Cheng Li[1], **Zhen Yang**[1*], **David Lillis**[2,3]
1. College of Computer Science, Faculty of Information, Beijing University of Technology, China
2. Beijing-Dublin International College, Beijing University of Technology, China
3. School of Computer Science, University College Dublin, Ireland
yangzhen@bjut.edu.cn

Abstract

In this paper we describe our efforts for the TREC OpenSearch task. Our goal for this year is to evaluate the effectiveness of: (1) a ranking method using information crawled from an authoritative search engine; (2) search ranking based on clickthrough data taken from user feedback; and (3) a unified modeling method that combines knowledge from the web search engine and the users' clickthrough data. Finally, we conduct extensive experiments to evaluate the proposed framework on the TREC 2016 OpenSearch data set, with promising results.

Introduction

In this year's OpenSearch Track, our main aims are: (1) Building an efficient ranking function that uses the information from a web search engine and the documents. (2) Explore a novel method that can use the clickthrough data from the feedback to improve the performance of the ranking function. (3) Create a unified model that combines both of these. As search engines play an ever more crucial role in information consumption in our daily lives, ranking relevance is always a challenge. The advancing state of the art for search engines presents new relevance challenges, which drives us towards using user feedback (e.g. clicks) to optimize the retrieval performance of search engines.

In addressing the OpenSearch task, we first crawled the document information for each query from Google Scholar, mainly document titles. We used this information to reflect the content for each document. Next, word2vector was used to build matrices to represent the documents after a series of data processing operations, and training was performed on the dataset to generate a classifier for each query. Then, we used the classifiers to make judgments about the scores of documents and uploaded the ranks to the API. In addition, we used the dataset to train the classifier including the clickthough data received as feedback through the from API. Clickthrough data has been shown to be highly useful for generating search engine rankings (Schuth, Balog, and Kelly 2015; Joachims 2002; Radlinski, Kurup, and Joachims 2008; Agichtein, Brill, and Dumais 2006). Finally, we used both of the above classifiers to create a unified modeling based on structural risk minimization, and ranked the candidate documents.

The remainder of the paper is organized as follows: In Section 2, we present our approach for ranking using web information and clickthough data. In Section 3, we report our experimental results. In Section 4, we conclude the paper.

Ranking System Framework

Figure 1 shows our system framework. It consists of four principal parts: (1) Information gathering, (2) Document annotation, (3) Ranking model and Unified model, (4) Results generation.

Information Gathering

The first step of the ranking process is to gather useful information. Candidate data was received from the Living Labs API, including training set and test set, each query and its corresponding doclist (a list of candidate documents) and user feedback for each query. With regard to information from the web, we crawled a doclist for each query from Google Scholar. For each query, we crawled the top documents up to a maximum of 100 documents. Document titles are the primary information received during this process. For some queries, 100 documents were not available, but the more than 70 were retrieved in each case. Moreover, we gathered a text corpus from Wikipedia to train the word2vector model. This corpus was approximately 11 GiB in size.

Document Annotation

The different search engines used in the OpenSearch task make different information available about each document. For instance, CiteSeerX provides only a single field with the full document text. In contrast, SSOAR returns many fields including abstract, author, description, identifier, language etc. The Microsoft Academic Search results include abstract and URL. Document title is common to each, so this was used for the generation of the document matrices.

Before generating the document matrices, we preprocessed the data. Then, we trained the word2vector model with the corpus and used this model to construct document matrices both of the sites and Google Scholar.

Two labels are applied to each document. One is based on its position in the Google Scholar results, and the other is

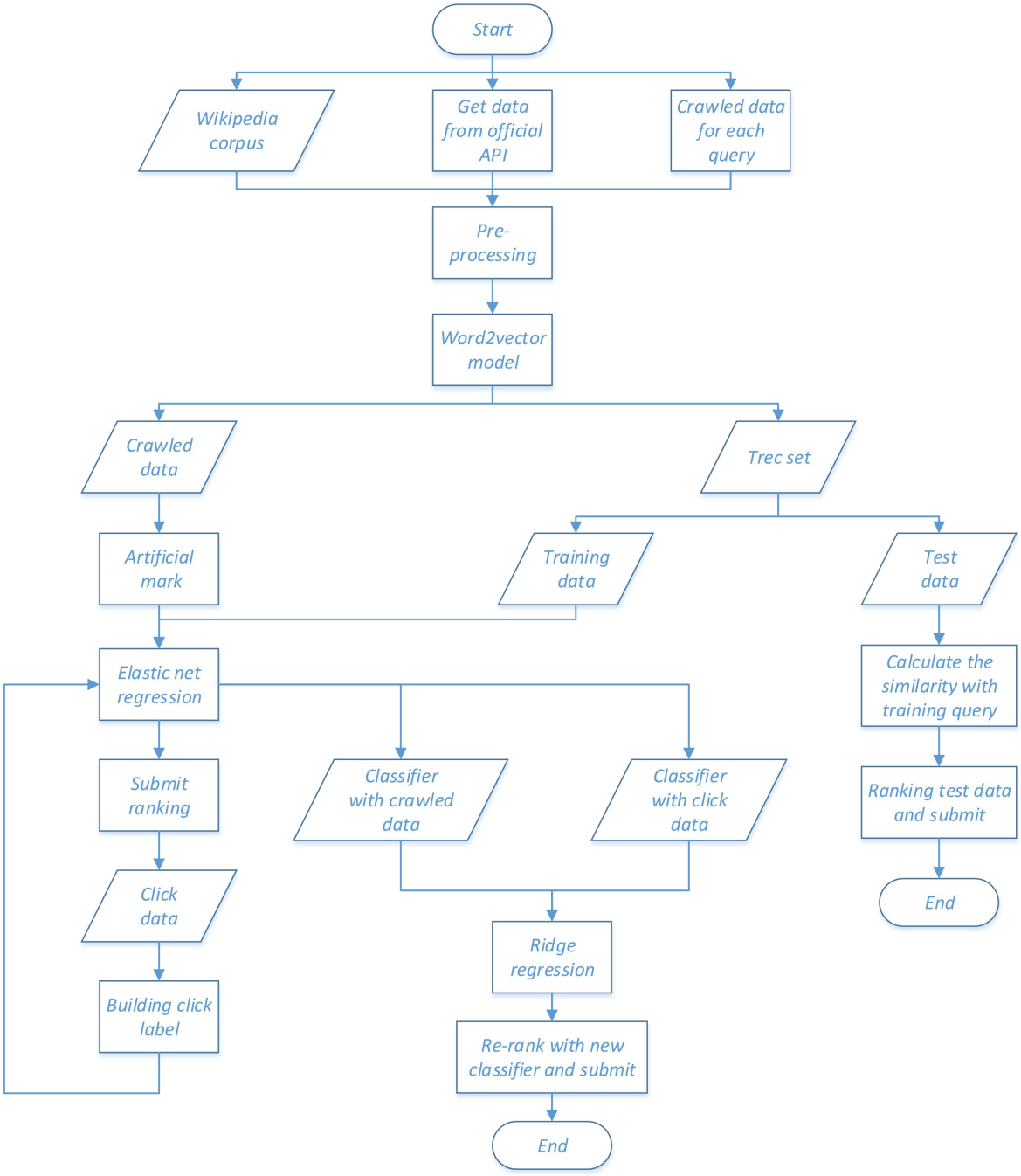

Figure 1: System Framework.

based on the clickthrough data. For the first label, we use five scores to represent different levels of the documents. The higher-ranked documents should have higher scores than those ranked later by Google Scholar. Each document is given a score as follows (rank ranges are inclusive):

- 5: ranked between position 1 and position 5
- 4: ranked between position 6 and position 10
- 3: ranked between position 11 and position 20
- 2: ranked between position 21 and position 50
- 1: ranked at position 51 or higher

The other label is based on clickthrough data. It can be understood as the probability of the document is clicked, and is defined by following formula:

$$P(d) = 0.5 + \frac{d_t - d_f}{d_t + d_f + 0.5} \times \frac{0.5}{1 + e^{-k(d_t + d_f - s - 0.25)}} \quad (1)$$

where $P(d)$ is the probability for document d, d_t is the number of times the document was clicked, d_f is the number of times the document was not clicked. The latter part of this formula is a logistic function to dampen the effect of clicks for small sample sizes (so that a document that has appeared once but has not been clicked will not get a probability of zero). K affects how quickly the function begins to rise and s is the shift factor. Values of $K = 0.33$ and $s = 10$ were used. A document that has never been clicked begins with a probability of 0.5, and this probability will rise or fall depending on the user clickthrough data.

Ranking Model and Unified Model

The ranking model consisted of the web information ranking model and the clickthrough data ranking model.

Existing search engines exhibit excellent performance in retrieving information, therefore the starting point for the model is to train the classifier using Google Scholar. We adopted Least Squares method into the framework, which can learn a linear model to fit the training data (Hu et al. 2013). The square loss is:

$$L(H, I) = ||GH - I||^2 \tag{2}$$

where G is the content matrix of Google document data, H is the linear classifier, and I is the label matrix we defined. Square loss is a widely used method for text analytics.

However, simple square loss alone is insufficient for training a stable and robust classifier. For the documents, it is observed that not all the words in titles are relevant to the query but instead we can use some key words to represent the documents. A mature method is sparse learning, which has been used in various fields to obtain an effective model. To ensure sparsity of the model, we used the L1-norm penalization based on square loss. To overcome overfitting, the most popular method is regularization. One of the most widely used methods is ridge regression, which introduces an L2-norm penalization. At the moment, the problem can be solved by the elastic net, which does automatic variable selection and continuous shrinkage, and selects groups of correlated variables.

This model is suitable for the web information ranking model, and can be used as clickthrough data ranking model by replacing Google document matrix G with the site document matrix C, and replacing the score labels with the second label outlined above.

A unified model was then created. As with the ranking model, we used square loss to train the model, which can be written as:

$$\min L(H_1, H_2) = ||GH_1 - GH_2||^2 + ||CH_1 - CH_2||^2 \tag{3}$$

where H_1 and H_2 are the classifiers obtained from the ranking model. L2-norm penalization was then used to control the robustness of the learned model (Beck and Teboulle 2009).

Results Generation

For the training dataset, we used the classifier obtained from ridge regression to rank. As for the test dataset, first we calculated the cosine similarity between the test queries and training queries, which can be defined as following formula:

$$sim(q_i, q_j) = \frac{q_i \cdot q_j}{||q_i|| \cdot ||q_j||} \tag{4}$$

where q_i and q_j are two vectors representing a test query and a training query and $||q||$ is the length of the vector q. And then, after normalizing the cosine similarity vector, we could get the classifiers of test queries, as follows:

$$H_i = \sum_{j=1}^{n} \alpha_{ij} H_j \tag{5}$$

where α_{ij} is the parameter of the cosine similarity with normalization, H_i is the test query classifier and H_j is the training query classifier.

If the two queries are semantically similar to each other, they should have a close representative in the word2vector model within the large scale corpus. Considering this aspect, we used the equation (5) to express the classifiers, and ranked the test dataset with these.

Experimental Results

This section, we introduce the evaluation methods and our results.

Evaluation Measures

During the OpenSearch runs, CiteSeerX and SSOAR both use interleaved comparisons on their live websites. The specific type of interleaving used is Team Draft Interleaving (TDI), whereby the rankings produced by participating teams are interleaved with the current production ranking of the site. Users are shown this interleaved ranking, but are unaware of the origin of the results. The ranker (participant or production) that contributes more documents that users click is preferred.

OpenSearch used five metrics for evaluation, as follows (Schuth, Balog, and Kelly 2015):

- Impressions: the total number of times when rankings (for any of the test queries) from the given team have been displayed to users.
- Wins: a win occurs when the ranking of the participant has more clicks on results assigned to it by Team Draft Interleaving than clicks on results assigned to the ranking of the site.
- Losses: A loss is the opposite to a win.
- Ties: a tie occurs when the ranking of the participant obtains the same number of clicks as the ranking of the site.
- Outcome: Outcome is defined as:

$$wins = \frac{wins}{wins + losses} \tag{6}$$

An outcome value below 0.5 means that the ranking of the participant performed worse than the ranking of the site (i.e., in overall, it has more losses than wins).

Table 1: Training Data Performances.

type	Outcome	Wins	Losses	Ties	Impressions
train	0.4468	21	26	5	52

Table 2: Test Data Performances.

Round	Outcome	Wins	Losses	Ties	Impressions
1	0.3333	3	6	1	10
2	0.6	6	4	1	11
3	0.5432	44	37	15	96

Results Analysis

Two sets of results were obtained: one based on the training data and the other based on the test data.

Table 1 shows our results from CiteSeerX for the training data. The evaluation method is Team Draft Interleaving (TDI), it statistically test whether the number of wins for the better retrieval function is indeed significantly larger by using a test against outcome ≤ 0.5. The outcome of the training data is 0.4468, which means the ranking function of the site performs better than ours, although not to a substantial degree.

Table 2 shows our from CiteSeerX on the test data. There were three rounds for the test period. In the first round, the site had the better performance than our system, in the second round, our system performed better, and in the third round, the outcome was 0.5432, our system performed better too. Around the all rounds, our ranking function is effective and not bad.

Conclusion

In this paper, we study the problem of optimizing the ranking function with clickthrough data. We used the word2vector model to represent the queries and the documents effectively instead of traditional vector space or other model. A unified framework is proposed, incorporating web information and clickthrough data. In the optimization process, the popular regularization methods of elastic net regression and ridge regression are adopted. Experiments over long periods were conducted to evaluate the proposed framework on a real world academic search engine and the experimental results demonstrate the effectiveness of our proposed framework.

In the future work, we will consider more features to better represent documents and conduct further extensive experiments to improve the robustness and stability of our method.

Acknowledgments

This research was supported by the National Natural Science Foundation of China (No. 61671030), the Excellent Talents Foundation of Beijing, the Importation and Development of High-Caliber Talents Project of Beijing Municipal Institutions (No.CIT&TCD201404052), and the Guangxi Colleges and Universities Key Laboratory of Cloud Computing and Complex Systems (No15205).

References

Agichtein, E.; Brill, E.; and Dumais, S. 2006. Improving web search ranking by incorporating user behavior information. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, 19–26. ACM.

Beck, A., and Teboulle, M. 2009. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences* 2(1):183–202.

Hu, X.; Tang, J.; Zhang, Y.; and Liu, H. 2013. Social spammer detection in microblogging. In *nternational Joint Conference on Artificial Intelligence*, volume 13, 2633–2639. Citeseer.

Joachims, T. 2002. Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, 133–142. ACM.

Radlinski, F.; Kurup, M.; and Joachims, T. 2008. How does clickthrough data reflect retrieval quality? In *Proceedings of the 17th ACM conference on Information and knowledge management*, 43–52. ACM.

Schuth, A.; Balog, K.; and Kelly, L. 2015. Overview of the living labs for information retrieval evaluation (ll4ir) clef lab 2015. In *International Conference of the Cross-Language Evaluation Forum for European Languages*, 484–496. Springer.