Information Retrieval Boosted by Category for Troubleshooting Search System

Bin Tong Toshihiko Yanase Hiroaki Ozaki Makoto Iwayama
Research & Development Group, Hitachi, Ltd.
1-280, Higashi-koigakubo, Kokubunji-shi, Tokyo 185-8601, Japan
{bin.tong.hh, toshihiko.yanase.gm}@hitachi.com
{hiroaki.ozaki.yu, makoto.iwayama.nw}@hitachi.com

ABSTRACT

Troubleshooting search system aims at extracting relevant information to solve the problem at hand. It is often the case that documents in troubleshooting system includes an abundant amount of domain-specific categories. However, the useful information about the domain-specific categories, such as relationship between words and categories and relationship between categories, is not fully utilized in simple query search and faceted search. In this paper, we propose an information retrieval method boosted by the domain-specific categories. Given a problem query and categories, the troubleshooting search system is able to retrieve the relevant information of interest with respect to the selected categories. The experiment results show our proposal improves the recall.

Categories and Subject Descriptors

H.3.3 [Information Search and Retrieval]: Retrieval Models

Keywords

Troubleshooting, Category, Co-occurrence Graph

1. INTRODUCTION

Troubleshooting [13] is a form of problem solving, and is often applied to repair malfunctioned facilities or equipments. Maintenance log [10, 5, 2] is one of important documents for troubleshooting, which is generated during conversations between customers and engineers in equipment maintenance. The maintenance log often includes the entries for problem titles and documents that relate to the problem description in details and instructions to solving the problem. To ease the management of the huge amount of the maintenance logs, domain-specific categories, such as machine code, trouble code, and countermeasure code, are used to tag for both the problem titles and the documents.

The target of information retrieval for troubleshooting [11] based on the maintenance logs is to help engineers to examine the similar situations of a problem within a certain period of time, which facilitates an appropriate solution. The troubleshooting search system requires the engineers to input a short problem query to search the relevant information from the documents. It may cause the lexical gap problem [8, 9], because it is difficult for the engineers to compose a succinct and precise problem query to represent their information needs.

Moreover, information about the domain-specific categories is not fully utilized in the problem query search. One way to use the category information is to make faceted search, in which the selected categories are used to filter the ranking results. For example, if a machine code is selected, all search results are restricted to the selected machine code.

However, the faceted search might have two problems in the troubleshooting search system. First, the information related to the selected categories can not be retrieved, since the retrieved information is limited to the selected categories. However, it is natural that system engineers tend to check relevant problems to facilitate their decision making. For example, given a selected machine code, the information about another machine code might be informative to solutions if two machine codes belong to the same machine series and have similar problems. Second, the ranking of search results is only dependent on the problem query but not on the selected categories. For example, a trouble code corresponds to a number of specific countermeasure codes. Given a selected trouble code, the information about its frequent countermeasures is expected to place higher in the ranked list of results.

To mitigate the lexical gap problem and the above mentioned retrieval problems, we propose an information retrieval method using a scoring technique. Our proposal is extended from a word co-occurrence graph in the QSB method [9] that aims at solving the lexical gap problem. In our proposal, besides using the word co-occurrence to score words in documents, the word's score is also weighted by a boosting term about the domain-specific categories. More specifically, the boosting term considers the relationship between categories and words and the relationship between categories. They are utilized to alleviate the above two retrieval problems with respect to the categories.

2. RELATED WORK

The information retrieval for troubleshooting is related to question answering [15] and query biased document summarization document summarization [14, 1, 3]. The most of work about question answering has been focusing on factoid

SIGIR Workshop on Graph Search and Beyond '15 Santiago, Chile
Published on CEUR-WS: http://ceur-ws.org/Vol-1393/.

question answering [6, 12, 7]. However, in the troubleshooting system, the answer of the question is a set of relevant sentences or phrases. As to query biased document summarization, there seems to be no work that leverages other auxiliary information, such as categories. In addition, it is worth to mention the work [12] about non-factoid question answering. In this work, Surdeanu et al. proposed a framework for answer ranking by exploiting various linguistic features generated from popular techniques, such as syntactic parsing, Name Entity Recognition (NER), and Semantic Role Labeling (SRL). However, except for regular sentences, a large number of typos and short phrases exist in maintenance logs. In such a case, those techniques might not perform well due to the irregularities in texts and the lack of training data in the troubleshooting domain.

Query Snowball (QSB) is a method for multi-document summarization that extracts the relevant sentences from multiple documents with respect to the given query. The basic idea of this method is to fill up the lexical gap between the query and relevant sentences by enriching the information need representation. In order to achieve it, a co-occurrence graph for the words in the queries and the documents is built. The words in the co-occurrence graph consist of three layers, which are Q words, R_1 words, and R_2 words. Q is the set of query terms. R_1 is the set of words that co-occur with a query term in the same sentence. R_2 is the set of words that co-occur with a word from R_1, excluding those that are already in R_1.

3. QUERY SNOWBALL WITH CATEGORY INFORMATION

To extract relevant information with respect to the selected categories, we extend the co-occurrence graph in QSB by integrating two types of relations, including the relationship between words and categories and the relationship between categories. The reason to extend QSB is that the co-occurrence graph is flexible to integrate the two relations.

The relationship between words and categories represents the distribution on words with respect to categories. If probabilities of words with respect to a given category are high, the information about these words is more likely to be retrieved given that category. The fundamental idea is that the distributions on co-occurrence probabilities of words with respect to different categories might be different. It is assumed that the words of higher probabilities with respect to a category are treated more important in that category. For example, a word appears more often in documents of a category than other categories. The word is therefore more important for that category than the others. Similarly, the relationship between categories represents the occurrences of categories. The information about categories, whose occurrence frequencies with respect to a specific category are high, is more likely to be retrieved. For example, a given category appears more often with specific categories than the others. The information about those specific categories with respect to the given category is treated more important than other categories.

3.1 Co-occurrence Graph Extension

In the first step, we build two co-occurrence graphs for the words and category values in the problem queries and the documents, respectively. If a category value is associated with a document, the words in the document have edges with the category value. In other words, this category value is associated with all the words in the sentences of documents. Similarly, if a category value is associated with a problem query, the words in the problem query have edges with the category value. In other words, this category value is associated with all the words in the problem query.

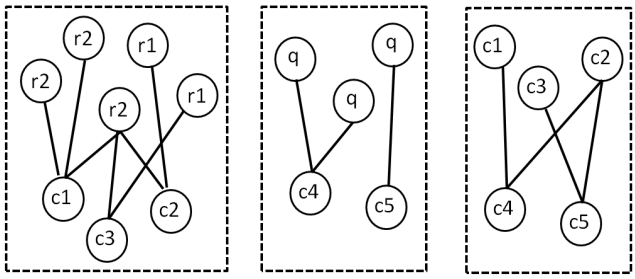

Figure 1: The co-occurrence graph among the words and the categories

In the second step, we build the co-occurrence graph for the category values in both problem queries and the documents. Suppose that a problem query corresponds to a document. In this graph, the category values associated with the problem query have edges with the category values associated with the document. As illustrated in Figure 1, the left side represents the co-occurrence graph for the R_1 and R_2 words and the category values associated with the documents; the middle part represents the co-occurrence graph for the Q words and the category values associated with the queries; the right side represents the co-occurrence graph for the category values in the queries and the category values in the documents. We define C_M as the set of the category values in the query set that are selected by the end-user, and C_N as the set of the category values in the query set that are not selected by the end-user. Similarly, we define C_J as the set of the category values in the document set that are selected by the end-user, and C_K as the set of the category values in the document set that are not selected by the end-user. We also define $C_{MN} = C_M \cup C_N$ as the set of the category values for the queries and $C_{JK} = C_J \cup C_K$ as the set of the category values for the documents.

3.2 Score Boosted by Category Information

In order to integrate two new relationships about categories into the QSB method, we invent a score with respect to a query, C_M and C_J, which is boosted by category information. The new score, which we call cqsb, can be formulated as follows:

$$cqsb(w) = qsb(w)\exp(\lambda \cdot s_{ctg}(w)) \quad (1)$$

where $qsb(w)$ is the score for a word w in the QSB method. $s_{ctg}(w)$ is the boosting term for the word w, which includes two new relationships about categories. More specifically, the probabilities between words and categories and the probabilities between categories, which are calculated through the co-occurrence graph, are used to boost the score $qsb(w)$. λ is a weight for the term $s_{ctg}(w)$. It can be seen from Eq. (1) that when $s_{ctg}(w)$ is larger than 0, $exp(\cdot)$ will be larger than 1. When multiplied with $qsb(w)$, it will give the word w a higher degree of importance. If the value of λ is set to

be 0, $s_{ctg}(w)$ does not take any effect. Note that $s_{ctg}(w)$ is always larger than or equal to zero. The score of a sentence is a summation of the scores for any combinations of two words, which is simply calculated by multiplying the *cqsb* scores of the two words.

3.3 Score for R_1 Words

The relevant score of a word r_1 ($r_1 \in R_1$) with respect to the category values can be formulated as follows:

$$s_{ctg}(r_1) = s_{wc}(r_1, Q_{C_M}^{r_1}) + s_{cc}(r_1, Q_{C_M}^{r_1}) \quad (2)$$

where $s_{wc}(r_1, Q_{C_M}^{r_1})$ measures the relationship between the words and the categories. $s_{cc}(r_1, Q_{C_M}^{r_1})$ measures the relationship between the categories. $Q_{C_M}^{r_1}$ is a set of top k query terms that co-occur most frequently with the word $r1$. In addition, the words in $Q_{C_M}^{r_1}$ follow a constraint that they should have edges with both the word r_1 and the category values in C_M.

The term $s_{wc}(r_1, Q_{C_M}^{r_1})$ in Eq. (2) can be calculated as:

$$s_{wc}(r_1, Q_{C_M}^{r_1}) = \sum_{q \in Q_{C_M}^{r_1}} \sum_{i=1}^{|C_{MN}^{q}|} \theta \frac{freq(c_i, q)}{freq(c_i)} \quad (3)$$

where C_{MN}^{q} is the set of category values for the queries that also have edges with q. Let θ be β if $c_i \in C_M$ and $c_i \in C_{MN}^{q}$, and be $1-\beta$ if $c_i \notin C_M$ otherwise. $freq(c_i, q)$ is the frequency of sentences that include both c_i and q, which can be also represented by the distribution on c_i with respect to q. It can be seen that Eq. (3) measures the closeness degree between the word r_1 and the category values in C_M through the words in $Q_{C_M}^{r_1}$, since the word r_1 does not directly have edges with the category values in C_M in the co-occurrence graph.

The term $s_{cc}(r_1, Q_{C_M}^{r_1})$ in Eq. (2) can be calculated as:

$$s_{cc}(r_1, Q_{C_M}^{r_1}) = \sum_{q \in Q_{C_M}^{r_1}} \sum_{\theta \in \Gamma} \sum_{c_i, c_j} \theta \frac{freq(c_i, c_j)}{freq(c_i)} \quad (4)$$

where $\Gamma = \{\beta, 1-\beta\}$. $c_i \in C_J^{r_1}$ and $c_j \in C_M^{q}$ when $\theta = \beta$. Note that $C_J^{r_1}$ is a set of categories in C_J that also have edges with the word r_1, and C_M^{q} is a set of categories in C_M that also have edges with the word q ($q \in Q_{C_M}^{r_1}$). Let $c_i \in C_{JK}^{-r_1}$ and $c_j \in C_{MN}^{-q}$ when $\theta = 1-\beta$. Note that $C_{JK}^{-r_1} = C_{JK}^{r_1} - C_J^{r_1}$ and $C_{MN}^{-q} = C_{MN}^{q} - C_M^{q}$. $C_{JK}^{r_1}$ is a set of categories in C_{JK} that also have edges with the word r_1, and C_{JK}^{q} is a set of categories in C_{MN} that also have edges with the word q ($q \in Q_{C_M}^{r_1}$). It can be seen that Eq. (4) measures the closeness degree of the category values in $C_J^{r_1}$ and C_M^{q}.

3.4 Score for R_2 Words

Similarly, the relevant score of a word r_2 ($r_1 \in R_2$) with respect to the category values can be formulated as follows:

$$s_{ctg}(r_2) = s_{wc}(r_2, Q_{C_M}^{r_2}) + s_{cc}(r_2, Q_{C_M}^{r_2}) \quad (5)$$

where $s_{wc}(r_2, Q_{C_M}^{r_2})$ measures the closeness degree between the word r_1 and the category values which have edges with q words. $s_{cc}(r_2, Q_{C_M}^{r_2})$ measures the closeness degree between the category values in the query set and the category values in the document set, which are respect to the word r_2. $Q_{C_M}^{r_2}$ represents a set of query terms q ($q \in Q$) that have close relationship with the word r_2. Since the word r_2 does not have edges with the Q words in the co-occurrence graph of the word-word relation, the measurement of the relation could be done through the R_1 words by using the frequency, such as $freq(r_1, r_2)$ and $freq(r_1, q)$. An intuitive example of the measurement is a multiplication of $freq(r_1, r_2)$ and $freq(r_1, q)$ for the word r_2 and the word q. The word q ($q \in Q_{C_M}^{r_2}$) also holds two constraints that the word q is able to reach the word r_2 in the co-occurrence graph through a specific word r_1 and the word q should have edges with the category values in C_M.

The term $s_{wc}(r_2, Q_{C_M}^{r_2})$ in Eq. (5) is calculated as:

$$s_{wc}(r_2, Q_{C_M}^{r_2}) = \sum_{r_1 \in R_{r_2}^{1}} \frac{freq(r_1, r_2)}{sum_{R_{r_2}^{1}}} s_{wc}(r_1, Q_{C_M}^{r_2}) \quad (6)$$

where $R_{r_2}^{1}$ represents a set of R_1 words which have the top k highest frequencies with the word r_2, and $sum_{R_{r_2}^{1}} = \sum_{r_1 \in R_{r_2}^{1}} freq(r_1, r_2)$. The term $s_{wc}(r_1, Q_{C_M}^{r_2})$ can be calculated by Eq. (3).

The term $s_{cc}(r_2, Q_{C_M}^{r_2})$ in Eq. (5) can be calculated through Eq. (4) by substituting $Q_{C_M}^{r_1}$ with $Q_{C_M}^{r_2}$.

4. EXPERIMENTS

In this experiment, we use the maintenance reports from a leading construction machinery company in Japan. We collect a part of the maintenance reports of 4 dominated troubles from total 19 troubles. Note that equipment code is the category for the problem queries. Phenomenon code and the countermeasure code are the categories for the documents. In each data set, one query consists of a problem query, model code, phenomenon code, and countermeasure code. We also manually label the important sentences from the documents. For each query, we search the sentences in the documents, and evaluate the performance by comparing if the sentences in the top rank are matched with the labelled sentences. Note that precision, recall and F-score are used as criteria. Among the three metrics, recall is the most important criterion. The reason is that, in troubleshooting system, system engineers prefer to examine all similar cases until they feel confident to solve the problem at hand. In this experiment, the training data and the test data are the same, since system engineers find out similar cases in the past from the troubleshooting system. Note that building and updating the co-occurrence graph can be done periodically in an unsupervised way.

For the comparisons, we select four baseline methods, which are *cqsb*, *qsb*, *lexsim*, and *lexsim+qsb*. We name our proposal by *lexsim+cqsb*. *lexsim* represents the lexical text similarity between the problem query and the sentence in the comments, which can be simply calculated by the cosine similarity between two vectors with bag-of-words features. One example of the bag-of-words features is the counts of frequent words in documents. Note that stop words are removed when counting the frequencies of words. *lexsim+qsb* aggregates *lexsim* and *qsb* to obtain the final ranking list, which belongs to the rank aggregation problem [4]. As a preliminary step, a simple rank aggregation method is used. Due to the different scales of *lexsim* and *qsb* scores, we simply sum up the orders of two ranking lists that use *lexsim* and *qsb*, respectively. In other words, the smaller the summation of two orders is, the higher rank the sentence gets.

Similarly, the orders of *lexsim* score and *cqsb* score is aggregated in $lexsim + cqsb$. In this experiment, we set two weights. One is for the weight between the *qsb* score or *cqsb* score and the *lexsim* score. The other is λ, which is for the category relation term in the *cqsb* score. The tuning space of the two weights is $[0, 0.01, 0.1, 1, 10]$. We use Macro Recall, Mean Average Precision (MAP), and F_3 score as recall, precision, and F-score, respectively.

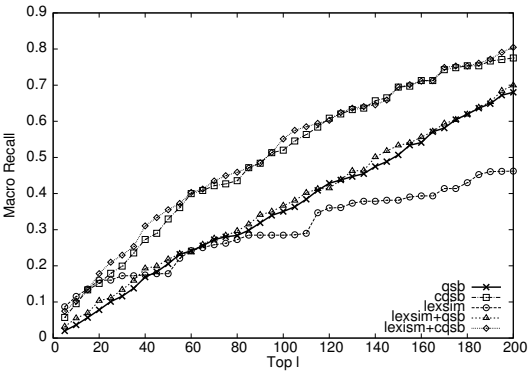

Figure 2: Recall for trouble code 03

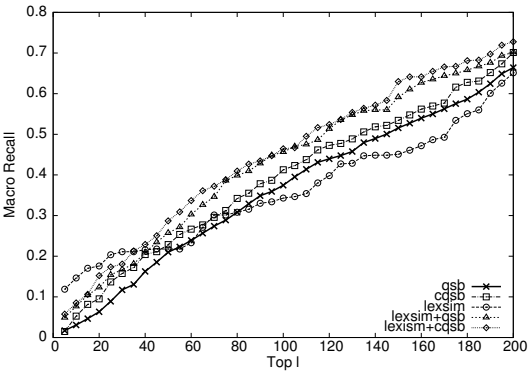

Figure 3: Recall for trouble code 05

As recall is the important criterion in troubleshooting search system, we calculate Macro Recall for each data set by setting the top l ($l \in \{5, 10, \ldots, 195, 200\}$) ranking sentences. Figure 2, Figure 3, Figure 4, and Figure 5 show the Macro Recalls of all the methods, when the number of the top l sentences changes from 5 to 200. It can be seen that the performances of $lexsim + cqsb$ are better than other baseline methods. It is also noticed from Figure 5 that the performance differences between *qsb* and *cqsb* are not obvious when the number of top sentences is increasing. The reason might be that the probabilities of words in informative sentences with respect to categories and the co-occurrence probabilities of categories are evenly distributed. Therefore, the second term on the right side of Eq. (1) does not differ over words to a large extent.

We also investigate MAP and F_3 score. For simplicity, we show their results in cases in which a best result and a worst result of Macro Recall for $lexsim + cqsb$ are achieved. The results are illustrated in Table 1 and Table 2, which are from the trouble code 03 data set and the trouble code 05 data set, respectively. Note that β and the number of the top l sentences are set to be 1 and 100, respectively. It is shown that $lexsim + cqsb$ outperforms $lexsim + qsb$, *cqsb*,

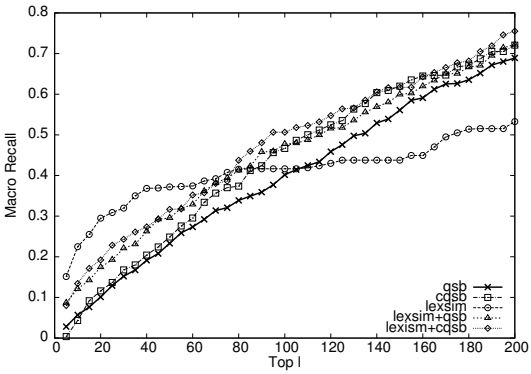

Figure 4: Recall for trouble code 14

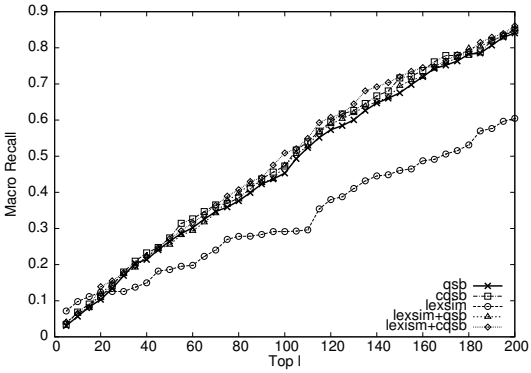

Figure 5: Recall for trouble code 18

Table 1: F_3 scores in trouble code 03 data set (best case)

Methods	Macro Recall	MAP	F_3 score
lexsim+cqsb	.5513	.0690	.3244
lexsim+qsb	.3665	.0407	.2036
lexsim	.2849	.0957	.2379
cqsb	.5200	.0609	.2964
qsb	.3503	.0312	.1731

Table 2: F_3 scores in trouble code 05 data set (worst case)

Methods	Macro Recall	MAP	F_3 score
lexsim+cqsb	.4645	.0557	.2679
lexsim+qsb	.3893	.0383	.2031
lexsim	.3431	.1346	.2971
cqsb	.4128	.0370	.2049
qsb	.3746	.0299	.1739

Table 3: The recalls at different values of β

β	$tr03_{100}$	$tr03_{200}$	$tr05_{100}$	$tr05_{200}$
1	.5200	.7752	.4128	.7013
0.8	.5225	.7865	.4133	.7057
0.6	.5560	.8020	.4276	.6935
0.4	.5456	.7809	.3901	.7045

and *qsb* in both cases. It is also noticed that, in the worst case, even if Macro Recall of $lexsim + cqsb$ is better than the others, its F_3 score is lower than that of *lexsim*. Note that F_1 has the same trend as F_3 in this experiment.

We also check the effect of the parameter β, as it influences the score about the relationship between the categories. Table 3 shows the macro recalls of *cqsb* at different values of β and l ($l \in \{100, 200\}$) in the data sets of trouble code 3 and trouble code 05. It is implied that 0.8 and 0.6 might be good values for *cqsb* to improve the recalls.

5. CONCLUSION

An information retrieval method using the scoring technique boosted by the domain-specific categories is proposed for troubleshooting search system. The knowledge about category information, which includes the relationship between words and categories the relationship between categories, is well integrated into a co-occurrence graph. The experiments on the maintenance logs proved the improvement of recalls, showing the effectiveness of using the category information.

6. REFERENCES

[1] A. Celikyilmaz and D. Hakkani-Tur. A Hybrid Hierarchical Model for Multi-document Summarization. In Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL), pages 815–824, 2010.

[2] A. Chandramouli, G. Subramanian, and D. Bal. Unsupervised Extraction of Part Names from Service Logs. In Proceedings of the World Congress on Engineering and Computer Science (WCECS), pages 826–828, 2013.

[3] H. Daumé, III and D. Marcu. Bayesian Query-focused Summarization. In Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics (ACL), pages 305–312, 2006.

[4] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank Aggregation Methods for the Web. In Proceedings of the 10th International Conference on World Wide Web (WWW), pages 613–622, 2001.

[5] B. Edwards, M. Zatorsky, and R. Nayak. Clustering and Classification of Maintenance Logs using Text Data Mining. In Data Mining and Analytics 2008, Proceedings of the Seventh Australasian Data Mining Conference (AusDM), pages 193–199, 2008.

[6] Z. Ji, F. Xu, B. Wang, and B. He. In Proceedings of the 21st ACM International Conference on Information and Knowledge Management (CIKM), pages 2471–2474, 2012.

[7] J. Ko, L. Si, and E. Nyberg. A Probabilistic Framework for Answer Selection in Question Answering. In Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings of the Conference (NAACL-HLT), pages 524–531, 2007.

[8] J.-T. Lee, S.-B. Kim, Y.-I. Song, and H.-C. Rim. Bridging Lexical Gaps Between Queries and Questions on Large Online Q&A Collections with Compact Translation Models. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 410–418, 2008.

[9] H. Morita, T. Sakai, and M. Okumura. Query Snowball: A Co-occurrence-based Approach to Multi-document Summarization for Question Answering. In The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference (NAACL-HLT), pages 223–229, 2011.

[10] E. Mustafaraj, M. Hoof, and B. Freisleben. Mining Diagnostic Text Reports by Learning to Annotate Knowledge Roles. In Natural Language Processing and Text Mining (NLPT), pages 46–67, 2007.

[11] F. Roulland, S. Castellani, A. N. Kaplan, M. A. Grasso, and J. O'Neill. Real-time Query Suggestion in a Troubleshooting Context, Xerox, US8510306 B2, 4, 2014.

[12] M. Surdeanu, M. Ciaramita, and H. Zaragoza. Learning to Rank Answers to Non-factoid Questions from Web Collections. Computational Linguistics, 37(2):351–383, June 2011.

[13] I. Sutton. Process Risk and Reliability Management: Operational Integrity Management. Elsevier, 2010.

[14] A. Tombros and M. Sanderson. Advantages of Query Biased Summaries in Information Retrieval. In Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR), pages 2–10, 1998.

[15] E. M. Voorhees. Overview of the TREC 2003 Question Answering Track. In The Text Retrieval Conference Proceedings (TREC), pages 54–68, 2003.