Probabilistic Model for Contextual Retrieval

Ji-Rong Wen
Microsoft Research Asia
Beijing, China
jrwen@microsoft.com

Ni Lao
Tsinghua University
Beijing, China
noon99@mails.tsinghua.edu.cn

Wei-Ying Ma
Microsoft Research Asia
Beijing, China
wyma@microsoft.com

ABSTRACT
Contextual retrieval is a critical technique for facilitating many important applications such as mobile search, personalized search, PC troubleshooting, etc. Despite of its importance, there is no comprehensive retrieval model to describe the contextual retrieval process. We observed that incompatible context, noisy context and incomplete query are several important issues commonly existing in contextual retrieval applications. However, these issues have not been previously explored and discussed. In this paper, we propose probabilistic models to address these problems. Our study clearly shows that query log is the key to build effective contextual retrieval models. We also conduct a case study in the PC troubleshooting domain to testify the performance of the proposed models and experimental results show that the models can achieve very good retrieval precision.

Categories and Subject Descriptors
H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

General Terms
Algorithms, Human Factors, Performance, Theory

Keywords
Contextual Retrieval, Probabilistic Model, Query Expansion, Query Log

1. INTRODUCTION
Contextual retrieval is one of the major long term challenges in information retrieval. In a recent workshop report [1], contextual retrieval is defined as "combine search technologies and knowledge about query and user context into a single framework in order to provide the most 'appropriate' answer for a user's information needs". In our organization, we are currently exploring applying contextual retrieval methods on the following two applications:

Mobile search – In the last decade, the dramatic increase in the use and availability of mobile devices such as cellular phones and personal digital assistants (PDAs) has made it more realistic to access information anytime and anywhere. Context-aware mobile search is a search paradigm in which applications can discover and take advantage of contextual information (such as user location, time of day, nearby people and devices, user activity, user profile, etc.). When a user submits a query "bus", search service could suggest bus routes with proper location and time schedule. Better retrieval precision is demanded in mobile search due to the limited bandwidth and display screen of mobile devices.

Context-aware PC troubleshooting – Traditionally, in technical support centers, when a customer send in a problem, information retrieval tools are used to find documents containing solution to the problem. Considering that most customers are not PC experts, their problem descriptions usually are inaccurate. Directly using customers' problem descriptions to retrieval relevant documents often cannot get satisfactory results. We are investigating a more advanced retrieval technique for PC troubleshooting by taking advantage of "PC context". Generally, when customer submits a problem to the support center, the current state information in his/her computer could be collected simultaneously. The state information contains very important cues about the root causes of the current problem. We could treat state information as the context of the problem description (or query) and view the PC troubleshooting as a typical contextual retrieval application. In section 4 and 5, we will show how contextual retrieval could greatly improve the retrieval performance of PC troubleshooting.

Seemingly, the above two applications are quite different. But after working on them for some time, we found that they face very similar challenges in essence. More specifically, the following issues are common in them:

- *Incompatible context*: in previous contextual search and personalized search works, query, context and document are made up of the same kind of component – word. Thus it is easy to accommodate the context data into the text retrieval framework. In many works, context is simply combined with query to construct an expanded query [7] [10]. However, for the two applications we show above, the component of context is totally different from that of query and document. For example, in the PC troubleshooting application, context contains low-level state information of computers. It is impossible to directly measure the similarity between context and document. Thus, simply incorporating context into query does not work.
- *Noisy context:* most previous works treat context as noise-free information. This may be true for some kinds of explicit contexts such as user profiles. But for applications where implicit contexts are collected automatically (just as in our cases), a large portion of contextual data may be irrelevant

SIGIR'04, July 25-29, 2004, Sheffield, South Yorkshire, UK.

to the current query. Noisy contextual information may negatively affect the retrieval performance. Another viewpoint to this problem is related to how to distinguish context-sensitive query and context-free query. If noisy context information could be accurately filtered, context-sensitive query could benefit from correct context and context-free query will not be biased by irrelevant context.

- *Incomplete query*: in contextual retrieval applications, users usually do not provide complete descriptions about their information needs. In the mobile search application, it is inconvenient for users to accurately describe their query purposes in a small device on the go. For PC troubleshooting, common users only know the symptoms of problems. They are not PC experts and thus can only provide inaccurate description of the problems. Therefore, in our view, the aim of contextual retrieval is not only to constrain queries but also to expand the queries to more complete and precise descriptions of the information needs.

These issues extensively exist in many contextual retrieval scenarios. However, despite that there are a lot of works about personalized search and contextual search, to our best knowledge, there is no a uniform framework to clearly address the issues. However, we think such a kind of models is very crucial to explore core problems of contextual retrieval in a systematic way.

In this paper, we describe how to use past query sessions in query logs to build probabilistic models to tackle with these issues. Query logs record past query sessions across a time span, and we could obtain a contextual search history about what documents are taken as good answers to what queries in certain contexts. Through the logs, probabilistic correlations among query terms, context elements and document terms could be built. Correlations between context and document could be used to solve the incompatible context problem, correlations between query and context could be used to filter out noisy contextual information, and correlations among query, context and document could be utilized to generate more accurate and complete query and thus solve the incomplete query problem.

Based on the correlations, we propose four probabilistic models to generate expansion terms for contextual retrieval. We also conduct a case study to use the models to troubleshooting PC problems. Our experimental results show that the proposed models can effectively achieve good retrieval performance. This case study actually motivates the main ideas in the paper and we hope that the experiences learnt from it could be extended to other applications.

The rest of the paper is organized as follows. Section 2 gives a general description of our methodology and discuss the correlations among query, context and document. Section 3 describes four probabilistic models for generating expansion terms for contextual retrieval. Section 4 introduces our case study of using the models in the PC troubleshooting application. We report our experimental results in Section 5. We discuss related work in Section 6 and conclude the paper and discuss future work in Section 7.

2. METHODOLOGY

A good model for contextual retrieval should correctly interpret user's information need, which is partially represented by both the query and corresponding context respectively. Also, the information need should be translated to the "language of document" to facilitate effective document retrieval. However, as we already pointed out, the three inherent issues in contextual retrieval applications make it difficult to find such a good model. First, the incomplete query and noisy context issues obstruct the accurate capture and interpretation of user's information need. Second, the incompatible context issue not only affects the understanding of information need but make it hard to construct the connection between context and document.

Obviously it is impossible to build a good model solely based on the document set. In this study, we investigate how to utilize query logs to meet the challenges.

Query log records past query sessions. A query session in contextual retrieval typically records a query-context-document sequence.

Query session := <query, context> [clicked document]*

Each session contains one query, its corresponding context and a set of documents which the user clicked on or labeled (which we will call clicked documents). The central idea of our method is that if a set of documents is often selected for the similar queries in similar contexts, the terms in these documents are strongly related to the terms of the queries and elements in the contexts. Also, if similar queries and similar contexts frequently co-occur in the logs, the terms in the queries are tightly correlated to elements in the contexts. Thus probabilistic correlations among query terms, contextual elements and document terms can be established based on the query logs, as illustrated in Figure 1.

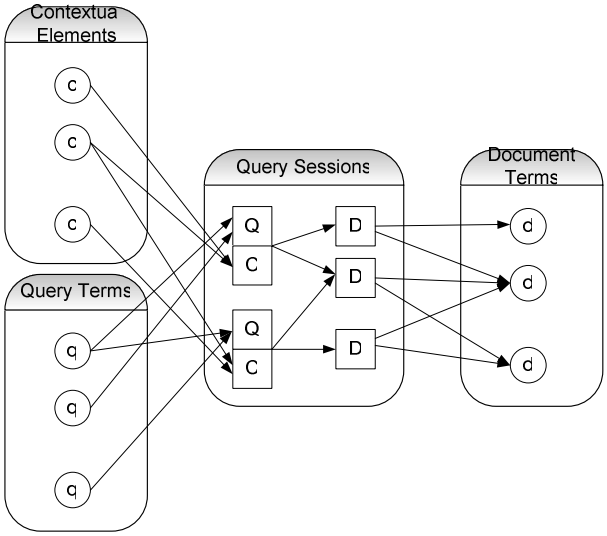

Figure 1. Query sessions build the correlations among query terms, contextual elements and document terms

Mutual information is a measure of the statistical dependency between two random variables based on Shannon's entropy and it is defined as the following:

$$I(a,b) = \ln\left(\frac{P(a,b)}{p(a)p(b)}\right)$$

Taking query logs as a bridge, we could use mutual information to determine the degrees of correlations among query terms, contextual elements and document terms (Figure 2).

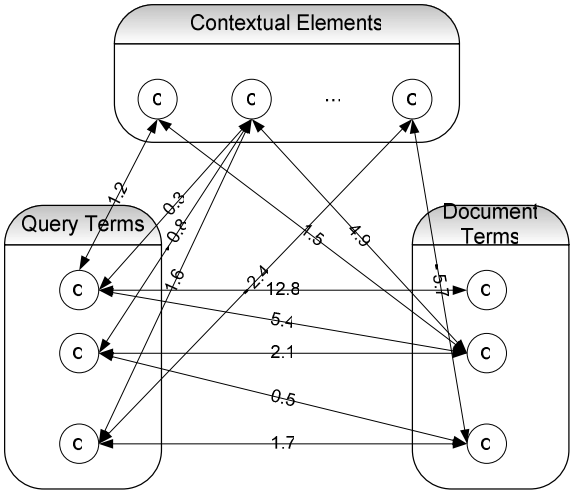

Figure 2. Mutual information among query terms, contextual elements and document terms

One important assumption behind this strategy is that query, context and document are tightly correlated in a query session. This assumption apparently doesn't hold for all query sessions. User may submit a query irrelevant to his/her context, user may search for something different from his/her habits, and user may also wrongly click a document or shift his/her intention when browsing the retrieval results. However, in the long run with a large amount of log data, the query sessions allow us to find significant correlations from a statistical point of view. Similar observations have been made in [6] [16].

[6] also pointed out that there is a big gap between term usages of queries and documents and a probabilistic model built through log mining could effectively bridge the gap. In this study, we further extend the previous utilizations of query logs to tackle the contextual retrieval problems. After building the correlations among query terms, contextual elements and document terms, models could be constructed to estimate "good" document terms based on a query and context. By using these document terms to expand the original queries, it is expected to retrieval good documents matching the information need represented by both query and context. With this strategy, we could easily address the mentioned three issues of contextual retrieval. With the correlations between contextual elements and document terms, incompatible context could be translated to proper document terms and thus retrieve documents related to the context. Correlations between query terms and contextual elements could be applied to filter contextual elements not related to the query. For the incomplete query issue, correlations among query terms, contextual elements and document terms could be used together to generate good expansion terms to make the query more complete and effective for retrieval.

3. MODELS FOR CONTEXTUAL RETRIEVAL

In this section, based on mutual information among query, context and document, we propose several probabilistic models to generate expansion terms for facilitating contextual retrieval.

3.1 Model 1: Context Only Model

The first model is a simple and intuitive one: document terms tightly correlated to context are chosen as expansion terms to modify the original query.

$$M_1(d \lessdot Q, C >) = I(d, C) = \sum_{c_i \in C} I(d, c_i)$$

where *I(d, C)* is the mutual information between context and document term. In terms of the manner of combining query and context, this model is similar to traditional solution to contextual retrieval. However, in our case, expansion terms do not directly come from the context but from documents correlated to the context. In this way, the incompatible context problem could be overcome.

3.2 Model 2: Query-Context Independent Model

The main shortcoming of Model 1 is that expansion terms are purely generated from context and thus are not correlated to query. As a context element may appear in a lot of query sessions containing diverse queries, irrelevant terms related to the context but not to the current query could be inappropriately chosen as expansion terms. Our second model uses query and context together to control the expansion process.

$$\begin{aligned} M_2(d \lessdot Q, C >) &= I(d, < Q, C >) \\ &= I(d, C) + I(d, Q) \\ &= \sum_{c_i \in C} I(d, c_i) + \sum_{q_j \in Q} I(d, q_j) \end{aligned}$$

where *I(d, Q)* is the mutual information between query and document term.

The advantage of Model 2 over Model 1 is that it will assign higher weights to document terms tightly correlated to both the query and context.

In this formula, since it is difficult to directly calculate joint probability *P(d, <Q, C>)* due to the data sparseness problem, we introduce three independence assumptions: independency among query terms, independency among context elements, and independency between query and context. While the former two assumptions are reasonable and extensively used in pervious probabilistic models, the third one could actually bring negative effects to contextual retrieval. For example, a document term occasionally has very high mutual information values with some contextual elements could be selected as expansion term even when it is not correlated to the current query, and vice versa.

3.3 Model 3: Query-Context Dependent Model

To lower the negative effect of the query-context independency assumption, we introduce a third model to accommodate query-context dependency relationships. This model is defined as:

$$M_3(d \lessdot Q, C>) = I(d, <Q, C>) \propto \sum_{c_i \in C} I(d, c_i) + \sum_{q_j \in Q} I(d, q_j) + \alpha \sum_{c_i \in C, q_j \in Q} I(d, <q_j, c_i>)$$

$\sum_{c_i \in C, q_j \in Q} I(d, <q_j, c_i>)$ is the mutual information between document term and query-context pair. It is this factor that introduces query-context dependency into the model. The formula means that a document term tightly correlated to both query and context will get an extra plus of its weight. Parameter α is used to adjust the weight of this query-context dependency factor.

3.4 Model 4: Context Filtering Model

A common problem in Model 1, 2 and 3 is that the noisy context problem is not addressed at all. Obviously, noisy context could easily lead to irrelevant expansion terms. So our fourth model utilizes mutual information between query and context to eliminate noisy elements in the context.

$$M_4(d \lessdot Q, C>) = I(d, <Q, C>) \propto I(d, <Q, C'>) \propto \sum_{c_i \in C'} I(d, c_i) + \sum_{q_j \in Q} I(d, q_j) + \alpha \sum_{c_i \in C', q_j \in Q} I(d, <q_j, c_i>)$$

where

$$C' = \{c \mid c \in C, I(c, Q) \geq \varepsilon\}$$

I(c, Q) is the mutual information between query and a contextual element. It is used to detect and remove contextual elements not tightly correlated to the current query. ε is the threshold for the filtering process. The intuition here is that if a contextual element is seldom observed to be related to a query in the past, this contextual element is considered as an irrelevant contextual information.

Our query expansion method is based on the probabilistic models described above. When a new query with context is submitted, a list of correlated document terms are selected and ranked based on the conditional probability obtained by the models. Then the top-ranked terms can be selected as expansion terms.

4. A CASE STUDY – TROUBLESHOOTING PC PROBLEMS

In this section, we introduce one interesting case study of applying the proposed models to troubleshoot PC problems. Actually our research on contextual retrieval is mainly motivated by this application.

4.1 Problem Description

The typical scenario of troubleshooting a PC problem is as the follows. The customer reports his/her problem to the engineer in a technique support center by sending a problem description and a set of related machine states. According to the symptom description and PC states, the engineer tries to find solution from a knowledge base containing a large amount of troubleshooting articles. When a solution is found, the engineer guides the customer to solve his/her problem step by step. Clearly, the retrieval accuracy is critical here for the engineer to help the customer to solve problem promptly. Unfortunately, the problem description sent by the customer is usually very vague and only surficial symptom about the problem is provided. Since similar symptom may be caused by many different root causes, a simple text retrieval tool is limited for helping engineer to quickly identifying the solution. In the real environment, for those unclear problem descriptions, engineers will ask the users questions to collect more information or manually analyze the state information and then modify the problem descriptions. In fact, contextual retrieval models could do this automatically.

A unique property of our problem setting is that a set of PC states is sent together with customer's problem description. These states contain important information to understand the problem. We take the problem description as the query and the registry states as the accompanying context of the query. Thus we view the PC troubleshooting as a typical contextual retrieval application.

Windows Registry is the main configuration state store on PCs, and there are typically more than 200,000 registry states on a machine and the total number of registry states is more that one million [15]. The incorrect values of these registry states may cause various problems. But it's extremely difficult for human to figure out the root cause from such a large number of states. Since the size of the contextual data is very huge, it would cause serious data sparseness problem and make the contextual retrieval task very difficult, if not impossible. Fortunately, we can quickly remove most of the contextual data based on the following two observations:

First, based on the log data, we found that only about 5,000 registry states are ever reported to cause problems in the past. Among these registry states, a few states are reported to cause problems many times, but most states only have small numbers of occurrences. We found the number of occurrences of the states approximately follows a Zipf distribution [18]. Based on this observation, we can simply remove the states not found to cause any problem in the past.

Second, a tool called Strider is invented to dramatically filter out irrelevant states [15]. The basic idea of Strider is that only states changed recently and used by the current failed operation are likely to be the root causes of a problem. The work flow of Strider is as follows. When a user finds an operation (such as sending email) has failed, and he remembers that he could do this operation in the past and has a snapshot of the good state information at that time (snapshots are automatically maintained in Windows), he can compare that good state with the current bad state to produce a diffing set. At the second step, he repeats the failed operation and traces the registry states related to the operation. The trace set and the diffing set are intersected to produce a candidate set. Hopefully, the root cause is contained in the candidate set. In our statistics, Strider can decrease the number of candidate states from 200,000 to less than 40 for 90% queries.

Based on the above two observations, contextual data with relatively small size for each query can be obtained. This could be taken as a query-independent contextual noise filtering step. Although the observations are come from the troubleshooting data, we guess that the contextual data in other domains may have the similar properties and methods for contextual noise filtering could be explored in a similar spirit.

4.2 Data

The document set is made up of 142,448 troubleshooting articles written manually by experienced engineers. Although these articles are represented in a XML structure, in our experiments, we treat each article as a free text document.

We collected 31,933 query sessions from past query logs. A session contains the problem description, related registry states and the solution articles recommend by engineer. Therefore, each case is a complete query-context-document sequence.

Based on the document set and sessions, we calculate mutual information *I(d, q), I(d, c), I(q, c), I(d, <q, c>)* respectively using maximum likelihood estimates.

$$I(d,q) = \ln \frac{f(d,q) \times M}{f(d) f(q)}$$

$$I(d,c) = \ln \frac{f(d,c) \times M}{f(d) f(c)}$$

$$I(q,c) = \ln \frac{f(q,c) \times N}{f(q) f(c)}$$

$$I(d,<q,c>) = \ln \frac{f(d,<q,c>) \times M}{f(d) f(<q,c>)}$$

where *f(q), f(c), f(d, q), f(d, c), f(q, c), f(d, <q, c>)* are the numbers of query sessions in which the terms or contextual elements or pairs appear. *f(d)* is the document frequency of term *d*. *M* is the number of documents and *N* is the number of query sessions.

In traditional statistical learning settings, smoothing techniques are used to solve the problem of data sparseness in the training process. Since the number of query sessions (31,933) is relatively small, the training data is still quite sparse given the large number of parameters involved. In our case, we use simple ways for smoothing. Since mutual information tends to give high value for rare events, we remove terms or contextual elements with occurrence frequencies smaller than 10. For any *f(a, b)=0* in the logs, we define *I(a, b)=0.*

We collected 29 queries of real-world failures reported by our colleagues and users of Web support forums. These cases are not picked up from the query logs and thus they are independent from the data we used to train the models. For each case, the problem symptom description and registry states are recorded and noisy states are filtered using methods in Section 4.1. The 29 queries are listed in Table 1.

5. EXPERIMENTAL RESULTS

This section provides empirical evidence about the effectiveness of our contextual retrieval models. All experiments are conducted on the PC troubleshooting data.

We use BM2500 in Okapi [14] as the ranking function for the backend retrieval system. It is of the form

$$\sum_{T \in Q} w^{(1)} \frac{(k_1 + 1) tf}{(K + tf)} \frac{(k_3 + 1) qtf}{(k_3 + qtf)}$$

where Q is a query containing terms T, tf is the frequency of occurrence of the term within a specific document, qtf is the frequency of the term within the topic from which Q was derived, and $w^{(1)}$ is the Robertson/Spark Jones weight of T in Q. K is calculated by

$$k_1((1 - b) + b \times dl / avdl)$$

where *dl* and *avdl* denote the document length and the average document length measured in some suitable unit, such as word or a sequence of words. In our experiments, we set $k_1 = 1.5$, $k_3 = 5$, $b = 0.5$, and $avdl = 200$. To be fair, we fix the values of parameters for all experiments.

Table 1. Queries used in the experiments

NO	QUERY	CONTEXT*
1	Cannot open Outlook attachment	29
2	Blank Windows activation page	27
3	IE always launch in offline mode	26
4	Spell checker does not work in Outlook Express	13
5	Title bar text changed after visiting some websites	78
6	Using Word as the Outlook email editor, cannot turn off Word 'Document Map' option	26
7	Cannot open address book in Outlook	39
8	Unable to play file with media player	41
9	Cannot open .exe programs	24
10	Outlook cannot remember password	38
11	Windows Messenger stops running	26
12	'File not found' message potentially due to a virus or misconfiguration in antivirus program	14
13	Not launch browser when click Internet in Start menu	12
14	Office 2000 patch error	35
15	Error 1606 when installing and uninstalling	18
16	Excel cannot startup	17
17	Can not change IE home page settings	13
18	Office shortcut bar missed	13
19	MSN Explorer dial-up unable to sign in using analog phone line	21
20	Duplex Printer becomes simplex printer	16
21	IE can only save image file as bmp file	32
22	Desktop Tab missing from Display Properties	24
23	CD ROM error	15
24	Windows Media player product feedback causes AV	14
25	Windows Update access denied	29
26	'Search' prompts when double clicking a folder	24
27	Start System Restore, blank window appears	19
28	Cannot create a new dial up connection	26
29	Cannot connect to internet	51

*We only list the number of registry states and do not list the exact states due to space limitation.

For comparison purpose, retrieval method using only query texts is taken as the baseline. Then we use Model 1, 2, 3, 4 to choose 10 expansion terms to modify the original queries to conduct retrieval. *qtf* of original query terms are set to 5 and *qtf* of expansion terms are set to 1 uniformly. For Model 3 and 4, α is set to 0.5. For Model 4, ε is set to 1.0 to filter noisy context.

Table 2. Ranking results for 29 queries by different models

Model	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12	Q13	Q14	Q15
Baseline	132	7	5	16	3	29	153	150	4697	51	5	30	11	32	4
Model 1	20	1	10	8	5	8	191	2	2039	4	8	2	12	5	1
Model 2	20	1	18	3	6	7	200	2	1308	3	5	2	22	5	1
Model 3	5	1	12	6	6	8	55	2	372	3	2	2	16	7	1
Model 4	1	1	6	1	3	6	1	4	1	3	6	2	3	2	1

Model	Q16	Q17	Q18	Q19	Q20	Q21	Q22	Q23	Q24	Q25	Q26	Q27	Q28	Q29
Baseline	11	9	31	9	5	17	3	1024	7	4	22	7	951	1410
Model 1	13	1	4	5	7	21	3	15	8	12	1	3	7	8
Model 2	2	1	3	4	6	18	3	5	21	32	1	1	7	3
Model 3	3	1	1	4	3	13	3	4	12	36	1	1	5	4
Model 4	1	1	1	4	6	18	3	9	21	26	1	1	1	1

Table 3. Proportions of queries whose answers are ranked at top 1, 5, 10, 20, 50 and 100

Model	Top 1	Top 5	Top 10	Top 20	Top 50	Top 100
Baseline	0%	24%	41%	55%	72%	76%
Model 1	14%	45%	69%	90%	93%	93%
Model 2	17%	59%	72%	83%	93%	93%
Model 3	21%	62%	76%	90%	93%	97%
Model 4	45%	72%	90%	93%	100%	100%

In our experiments, Porter stemming and a stopword list containing 426 words are used. We only use single word and do not consider phrase information.

One particular characteristic of our experimental setting is that for each of the 29 queries there is only one relevant document to serve as the solution. We therefore manually choose the article exactly contains the solution as the only relevant document for each query. Then we calculate the proportions of queries for which the correct answers are found at top 1, 5, 10, 20, 50 and 100 results to quantitively measure the performance of the models. In practice, correct answer ranked beyond top 10 is nearly useless since engineers have no time to go through too many documents to find solutions.

The experimental results of the baseline and Model 1-4 are illustrated in Table 2 and Table 3.

The baseline method performs badly. It can only rank the answers of 41% queries at top 10 and there is no one query whose answer could be ranked at top 1. It shows that it is hard to achieve good performance without using contextual data in this application since queries here are usually with an incomplete or vague description. For example, Query 9 "Cannot open .exe programs" is a very high-level description of the problem. There are a lot of possible causes for this problem. In the current case, the real root cause of the problem is that the PC is infected by the "SirCam Virus". So the answer document is one containing the solution of getting ride of the virus. However, when only query text is used to conduct the retrieval, such a important information is not captured and the correct answer is ranked at No. 4697. There are also several queries whose answers are ranked at top 5, namely Query 3, 5, 11, 15, 20, 22 and 25. Most of these queries describe problems with one or few possible root causes and thus are with little ambiguity. In these cases, contextual data is less important.

By simply using the contextual data, Model 1 achieves a significant improvement on retrieval performance over the baseline. The contextual data is useful to find good terms about the root cause of the problems. However, since some expansion terms produced from the contextual data could be irrelevant to the current query, the ranking results of some queries become worse.

Model 2 uses query and context together to generate expansion term and its ranking results are better than Model 1. Moreover, when adding the query-context dependency factor into the model, Model 3 improves the performance further.

Finally, the results of Model 4 show that contextual noise filtering plays a very important role of improving retrieval performance. This model could rank the answers of 45% queries at top 1 and 90% queries at top 10. Such a precision could greatly improve the productivity of support engineers. The big performance improvement may partially attribute to that there are a lot of noisy information in context in this application. However, other applications, especially those in which contextual data is produced implicitly, may suffer from the same problem and query-dependent contextual noise filtering could be a crucial technique for them too.

6. RELATED WORK

There has been some work on using query logs to enhance Web searching. [2] exploited "clickthrough data" in clustering URLs and queries using graph-based iterative clustering technique. [16] used a similar method to cluster queries according to user logs in order to find Frequently Asked Questions (FAQs). These FAQs are then used to improve the effectiveness of question answering.

There are also a lot of works on query expansion [5] [6] [12] [17]. The idea in [6] is closest to ours. The authors found that there is a big gap between term usages of queries and documents. They built a probabilistic model to describe the correlations between query terms and documents terms and then used this model to choose high-quality expansion terms. They reported a big performance improvement in a real search engine. However, their model targets

to traditional retrieval problems and do not consider any contextual information.

Many problems in machine translation, information retrieval, text classification can be modeled as one based on the relation between two spaces. The central issue of statistical machine translation is to construct a probabilistic model between the spaces of two languages [4]. In information retrieval, many statistical methods [3] [8] [9] have been proposed for effectively finding the relationship between terms in the space of user queries and those in the space of documents. [11] proposed using texts and the associated summaries to build a stochastic keyword generation model to improve text classification.

[10] discussed some general issues and challenges in Web contextual search. [7] proposed a contextual search method to extract relevant terms from surrounding text when query is picked up from a text body.

To our best knowledge, there is no previous work extensively discusses the correlations among query, context and document in the contextual retrieval setting. Also, the incompatible context, noisy context, incomplete query problems are not addressed by any existing retrieval models.

7. CONCLUSION

In this paper, we propose several probabilistic models for contextual search. These models are designed mainly to solve the important issues in contextual search, such as incompatible context, noisy context, and incomplete query. We conduct a thorough case study in the PC troubleshooting domain to testify the performance of the models and experimental results show that the models can achieve very good retrieval precision.

Although we only test our proposed models in the specific PC troubleshooting domain, we believe that the basic ideas in the paper could be adopted to deal with similar retrieval problems in other domains. Actually, we are investigating how to apply the models to mobile search, which exhibits the similar context properties such as incompatible and noisy context.

In the paper, we focus on using a query expansion strategy to build contextual retrieval models. In the future, we plan to use language model and translation model methods to build other models for contextual search, in a similar way to the models in [3] [8] [9] [13].

8. REFERENCES

[1] Allan, J. et al, Challenges in Information Retrieval and Language Modeling, Report of a Workshop held at the Center for Intelligent Information Retrieval, University of Massachusetts Amherst, September 2002

[2] Beeferman, D. and Berger, A., Agglomerative clustering of a search engine query log, In Proceedings of ACM SIGKDD 2000, Boston, MA, USA, pp. 407-416, 2000

[3] Berger, A. and Lafferty, J., Information Retrieval as Statistical Translation. In Proceedings of ACM SIGIR 1999, pp. 222-229, 1999

[4] Brown, P., Della Pietra, S., Della Pietra, V. and Mercer, R., The mathematics of statistical machine translation: Parameter estimation," Computational Linguistics, 19(2), pp. 263-311, 1993

[5] Buckley, C., Salton, G., Allan, J., and Singhal, A., Automatic query expansion using SMART, TREC 3. Overview of the Third Text REtrieval Conference (TREC-3), pp. 69-80. NIST, November 1994.

[6] Cui, H., Wen, J.-R., Nie, J.-Y., and Ma, W.-Y., Query Expansion by Mining User Logs, IEEE Transaction on Knowledge and Data Engineering, Vol. 15, No. 4, pp. 829-839, July/August 2003

[7] Finkelstein, L. et al, Placing Search in Context: The Concept Revisited, In Proceedings of the Tenth International World Wide Web Conference (WWW10), Hong Kong, May 2001

[8] Jin, R., Hauptmann, A. G. and Zhai C., Title Language Model for Information Retrieval, In Proceedings of the ACM SIGIR 2002, pp. 42–48, 2002

[9] Lafferty, J. and Zhai, C., Document Language Models, Query Models, and Risk Minimization for Information Retrieval, In Proceedings of the ACM SIGIR 2001, pp. 111–119, 2001

[10] Lawrence, S., Context in Web Search, IEEE Data Engineering Bulletin, Volume 23, Number 3, pp. 25–32, 2000

[11] Li, C., Wen, J.-R. and Li, H., Text Classification Using Stochastic Keyword Generation, Proceedings of the Twentieth International Conference on Machine Learning (ICML 2003), Washington, DC USA, August 2003

[12] Mitra, M., Singhal, A. and Buckley, C., Improving Automatic Query Expansion. In Proceedings of the ACM SIGIR 1998, pp. 206-214, Melbourne, August 1998

[13] Ponte, J. and Croft, W. B., A Language Modeling Approach to Information Retrieval. In Proceedings of the ACM SIGIR 1998, pp. 275–281, 1998

[14] Robertson, S. E., Walker, S. and Sparck Jones, M. et, al., Okapi at TREC-3, In D. K. Harman, editor, In Proceedings of the Second Text Retrieval Conference (TREC-3), NIST Special Publication, 500-225, 1995

[15] Wang, Y.-M., Verbowski, Chad., Dunagan, J., Chen, Y., Wang, H.J., Yuan, C., and Zhang, Z., "STRIDER: A Black-box, State-based Approach to Change and Configuration Management and Support," in Proc. Usenix Large Installation Systems Administration (LISA) Conference, pp. 159-171, October 2003.

[16] Wen, J.-R., Nie, J.-Y. and Zhang, H.-J., Query Clustering Using User Logs, ACM Transactions on Information Systems (ACM TOIS), 20(1), pp. 59-81, 2002

[17] Xu, J. and Croft, W.B., Improving the Effectiveness of Information Retrieval with Local Context Analysis. ACM Transactions on Information Systems (ACM TOIS), 18(1), pp. 79-112, 2000

[18] Zipf, G.K., Human Behavior and Principle of Least Effort: an Introduction to Human Ecology, Addison Wesley, Cambridge, MA, 1949