Towards More Effective Solution Retrieval in IT Support Services Using Systems Log

Rongda Zhu[1], Yu Deng[2], Soumitra (Ronnie) Sarkar[2(✉)], Kaoutar El Maghraoui[2], Harigovind V. Ramasamy[2], and Alan Bivens[2]

[1] Department of Computer Science, University of Illinois at Urbana Champaign, Urbana, IL 61801, USA
rzhu4@illinois.edu

[2] IBM T.J. Watson Research Center, Yorktown Heights, NY 10598, USA
{dengy,sarkar,kelmaghr,hvramasa,jbivens}@us.ibm.com

Abstract. Technical support agents working in the IT support services field resolve IT problems. They are often faced with the daunting task of identifying the correct solution document through a search system from large corpora of IT support documents. Based on the observation that system logs may contain critical information for identifying the root cause of IT problems, we explore the idea of automatic query expansion by using system logs as a bridge to link queries with the most relevant documents. Given the original query from a user such as a technical support agent, an intermediate query is first formed by adding key terms extracted from system logs using domain-specific rules. Based on topic models, further key terms are selected from corpora of IT support documents, which are combined with the intermediate query to form the final query. Our experimental results show that expanding queries using system logs together with topic models yields better performance in retrieving relevant IT support documents than using topic models only.

Keywords: Log-aided query expansion · Topic model · Retrieval · IT support services

1 Introduction

In the IT support services field, technical support agents and system administrators shoulder the responsibility of troubleshooting and assisting customers with resolving IT problems. The initial information provided to an agent by a customer facing IT problems is often incomplete and may not even be particularly useful in resolving the problem other than to inform the agent that some problem has occurred. Faced with the pressure of resolving customer-reported problems as quickly as possible, technical support agents use a variety of methods to improve the problem determination process. In this work, we consider two methods: (1) the use of system logs to help understand the state of the system on which a problem has been reported, and (2) the use of search systems to retrieve the correct solution from large corpora of IT support documents, where

Q.Z. Sheng et al. (Eds.): ICSOC 2016, LNCS 9936, pp. 730–744, 2016.
DOI: 10.1007/978-3-319-46295-0_52

insights gathered from system logs are used to enhance end user search queries to improve the quality of search results.

There are many popular, publicly available search frameworks such as ElasticSearch [9], Sphinx [26], Lucene [19], Solr [25], Xapian [29], and Indri [13]. Over time, users of search systems become adept at effective query formulation. However, in search systems dealing with specialized domains, users often need to combine search skills with domain-specific knowledge for effective query formulation. For example, IT support services is a specialized domain with huge corpora of both public and proprietary information. While domain-specific knowledge can be acquired over time, it represents a barrier that must be lowered for two reasons: (1) high attrition rates for support agents and (2) the pressure to resolve IT problems in the quickest possible manner.

Various forms of user context have long been used in the field of information retrieval for improving information search efficacy. For example, user location, search history, and implicit user behavior have been used for search type-ahead and improved search engine ranking and accuracy. However, in specialized domains, we believe there is an opportunity to go even further. In particular, we consider *log-aided query expansion* for the domain of IT support services, i.e., the use of relevant *system (or application) logs* as a bridge to link queries with the most relevant IT support documents. We further enhance the expanded intermediate query with key terms selected from corpora of IT support documents through the use of topic models [4,5]. We present early experimental evidence demonstrating the promise of the approach in lowering the barrier for effective query formulation and raising the precision of search systems for IT support. We view this work as a first step towards IT remediation systems that can automatically leverage search to diagnose and resolve IT problems.

The rest of the paper is organized as follows: Sect. 2 presents an example which motivates the need for enhancing the search process for IT support. Section 3 describes the approach designed for log-aided query expansion. Experimental evaluation is discussed in Sect. 4. Section 5 discusses related work and compares them with our approach. The paper concludes with a summary and future work in Sect. 6.

2 Motivating Example

2.1 Example System Log Files

System logs are used to record events that occur at various layers of a computing system: *firmware*, *hypervisor*, *operating system*, *middleware*, and *applications*. These error log files are extremely valuable tools for diagnosing and managing systems. We present two real sample logs from IBM POWER systems [28] deployed at client sites, representing information that is used by IBM technical support representatives to diagnose system failures. The customer specific data in the logs have been altered for privacy reasons.

The first system log we show is an excerpt from the *iqyylog* file that is generated from an IBM Hardware Management Console (HMC) [28]. Mid-range and

large IBM POWER servers need a HMC to create and manage logical partitions, dynamically reallocate resources, invoke Capacity on Demand, and facilitate hardware control. High-end servers with Bulk Power Controllers (BPC) require at least one HMC acting as a Dynamic Host Configuration Protocol (DHCP) server. Typically, more than one HMC is recommended for enhanced availability. When errors occur at the hardware level, the Flexible Service Processor (FSP) and/or BPC asynchronously notify the HMC that a platform error log or event log is available. The FSP is a firmware component that provides diagnostics, initialization, configuration, run-time error detection and correction functions. The FSP connects the managed system to the HMC. The HMC then reads the error log data from the FSP and BPC. Significant HMC events, including platform logs and problem analysis results, are recorded in the HMC *iqyylog*. The latter is a binary file and requires a decoder to view it. This log file among others is either submitted to IBM's technical support through an automated system called *Call Home*, or manually uploaded by the customer to one of IBM's FTP servers for further analysis by support representatives. Figure 1 contains sample content from a decoded *iqyylog* file with various events captured from an IBM POWER 7 system. Each entry in the log file shows a recorded platform event log (PEL_EVENT) along with its timestamp. Some of these events show reference codes and other error details that are key for problem determination. The entry also shows information related to the system that generated the event in the following format: TTTTMMM/NNNNNNN, where TTTT is the machine type, MMM is the model number, and NNNNNNN is the serial number. The example shows that problem analysis was triggered at 10:26 (tagged with PA_START). The results of the problem analysis shows the error code: *A7001152*. The error reported is a generic error that implies that the platform firmware detected a timeout condition which caused a reset of the service processor (FSP). For this particular customer problem, the support agent further examined the FSP dumps and determined that no action was needed, since the FSP was busy and slow to respond to the hypervisor, which in turn caused the timeout and hence a reset.

Another sample log file from an IBM POWER/AIX server is the *snap* file. The AIX *snap* command is used to gather a large amount of system configuration data and compress it into a *snap* core file. The file contains information such as the version of AIX the system is running, what hardware it is running on, what error messages were recorded, what processes were running when the system crashed, what is the firmware level, etc. The information gathered is used to identify and resolve system problems. The *snap* file can also be automatically uploaded to IBM's support repository or manually uploaded by a customer. Figure 2 shows an example error log entry captured from a POWER server. The log shows SCSI disk errors reported from hdisk2. Additional analysis of the diagnostics reported can be used to confirm that hisk2 is failing and needs to be replaced.

The examples discussed above illustrate that system log files contain valuable information that is often used by system administrators and technical support professionals to understand the root cause of a problem, and to gather enough data to effectively query existing knowledge sources in search of resolutions.

The key contribution of this paper is the insight that automatic expansion of support agent queries by analyzing system/application logs and extracting important search terms from those logs can lead to significant improvement in precision and recall. The initial agent query typically represents the customer view of the problem and focuses on the *symptom*, e.g., "my machine wont boot." Search results returned using such queries are not usually effective for problem resolution since the queries do not represent the *root cause*. System logs can complement the agent query in an effective manner, since they contain better indicators of the root cause, e.g., "SAS controller firmware update failure."

We propose an Automated Query Expansion (AQE) system which can identify a set of log file terms to complement the query terms submitted by an agent. In the above example, the AQE would automatically identify terms associated with the controller firmware update issue. That would improve the chances for problem resolution, especially if the machine boots from a remote SAS drive over a storage area network (SAN). While support services agents can perform this task manually, it takes years of experience and training to do it well; thus the value proposition of a systems log-aided AQE system that can perform the task automatically regardless of agent skill.

```
[*] = default formatting; [r] = raw (hex dump)

[*][r]  6005 02-07-16 10:28:04:07 [ +1.0] +FSPDump_PWAD
[*][r]  0B46 02-07-16 10:28:02:35 [ +1.0]  PA_DOM_PRM     domain=9179-MHD/052348T; primary=70
[*][r]  E302 02-07-16 10:28:01:42 [ +1.0]  XUPD           E302F817 921138A9          00
[*][r]  6005 02-07-16 10:27:57:34 [ +1.0] +FSPDump_FSPM
[*][r]  E346 02-07-16 10:26:38:43 [ +1.0]  PA_END
[*][r]  0B14 02-07-16 10:26:38:41 [ +1.0]  PA_Results     A7001152  null PN 57
[*][r]  0B11 02-07-16 10:26:38:38 [ +1.0]  SHProbOpen     Problem 57
[*][r]  E346 02-07-16 10:26:38:26 [ +1.0]  End_PA_Queue
[*][r]  6010 02-07-16 10:26:36:60 [ +1.0]  PEL_Event      B1829543 9179-MHD/052348T
[*][r]  6010 02-07-16 10:26:35:24 [ +1.0]  PEL_Event      B1812638 9179-MHD/052348T
[*][r]  6010 02-07-16 10:26:35:21 [ +1.0]  PEL_Event      B1819522 9179-MHD/052348T
[*][r]  6010 02-07-16 10:26:35:15 [ +1.0]  PEL_Event      B1812A01 9179-MHD/052348T
[*][r]  6010 02-07-16 10:26:34:60 [ +1.0]  PEL_Event      B1812A01 9179-MHD/052348T
[*][r]  6010 02-07-16 10:26:34:56 [ +1.0]  PEL_Event      B1812A01 9179-MHD/052348T
[*][r]  6010 02-07-16 10:26:34:52 [ +1.0]  PEL_Event      B1812A01 9179-MHD/052348T
[*][r]  6010 02-07-16 10:26:34:48 [ +1.0]  PEL_Event      B1812A01 9179-MHD/052348T
[*][r]  6010 02-07-16 10:26:34:44 [ +1.0]  PEL_Event      B1812A01 9179-MHD/052348T
[*][r]  6010 02-07-16 10:26:33:97 [ +1.0]  PEL_Event      B1812A01 9179-MHD/052348T
[*][r]  6010 02-07-16 10:26:33:93 [ +1.0]  PEL_Event      B7006978 9179-MHD/052348T
[*][r]  6010 02-07-16 10:26:33:90 [ +1.0]  PEL_Event      B1819537 9179-MHD/052348T
[*][r]  6010 02-07-16 10:26:33:86 [ +1.0]  PEL_Event      B1812A01 9179-MHD/052348T
[*][r]  6010 02-07-16 10:26:33:80 [ +1.0]  PEL_Event      A7001152 9179-MHD/052348T
[*][r]  6010 02-07-16 10:26:33:77 [ +1.0]  PEL_Event      B7006979 9179-MHD/052348T
[*][r]  6010 02-07-16 10:26:33:25 [ +1.0]  PEL_Event      A7001151 9179-MHD/052348T
[*][r]  E346 02-07-16 10:26:33:24 [ +1.0]  PA_START
```

Fig. 1. A sample ***iqyylog*** events log

```
LABEL:      SC_DISK_ERR4
IDENTIFIER: DCB47997

Date/Time:       Wed Jan  10 09:57:34 2016
Sequence Number: 2593
Machine Id:      11B27B74411
Node Id:         W4TSSLOG
Class:           H
Type:            TEMP
WPAR:            Global
Resource Name:   hdisk0
Resource Class:  disk
Resource Type:   vdisk
Location:        U9117.MMA.0626C64-V3-C2-T1-L8100000000000000


Description
DISK OPERATION ERROR

Probable Causes
MEDIA
DASD DEVICE

User Causes
MEDIA DEFECTIVE

    Recommended Actions
    FOR REMOVABLE MEDIA, CHANGE MEDIA AND RETRY
    PERFORM PROBLEM DETERMINATION PROCEDURES
```

Fig. 2. A sample system error log file

2.2 Example System Log-Aided Query

We present a real-world example that illustrates the value added by log-aided query expansion. We consider a typical query used by support agents as part of their standard problem determination procedure. Error codes such as System Reference Codes (SRC) are often used to construct queries while troubleshooting hardware problems. An SRC code is a sequence of eight characters that identifies the name of the system component that detects the error and the underlying condition. The first 4 characters indicate the error type, while the last 4 characters provide additional information such as the underlying error condition.

Figure 3 shows search results returned when an IT support agent chose the SRC code *"10009028"* as the query term. What the agent did not know at the time of issuing the query was that this error code was being reported by a POWER7 system. The results obtained show documents that pertain to various versions of IBM POWER systems, with the relevant result appearing third from the top. Figure 4 shows the results from an expanded query formed based on the system's log data. The exact platform version (namely, POWER7) is extracted from the log data and the following expanded query is formed: *"10009028 POWER7"*. The top search result in this case is the correct document which describes how to resolve the error. The relevant document appearing ranked first versus third has the potential to cut down problem resolution time from hours to minutes, which in turn may significantly impact customer satisfaction.

Number	Document Title	Snippet	Confidence
1	10009028 (POWER6)	...-01.ibm.com/support/knowledgecenter/POWER6/area7/**10009028**.htm http://www-01.ibm.com/support/knowledgecenter/POWER6/area7/**10009028**.htm knowledgecenter POWER6 **10009028 10009028 10009028 10009028** Explanation SPCN Licensed Internal Code is not...	100%
2	10009028 (POWER8)	...-01.ibm.com/support/knowledgecenter/POWER8/p8eai/**10009028**.htm http://www-01.ibm.com/support/knowledgecenter/POWER8/p8eai/**10009028**.htm knowledgecenter **POWER8 10009028 10009028 10009028**...	97%
3	10009028 (POWER7)	...-01.ibm.com/support/knowledgecenter/POWER7/p7eail/**10009028**.htm http://www-01.ibm.com/support/knowledgecenter/POWER7/p7eail/**10009028**.htm knowledgecenter POWER7 **10009028 10009028** Subscribe to this information IBM PowerLinux information **10009028**...	96%
4	10009028 (POWER5 POWER6)	...-01.ibm.com/support/knowledgecenter/POWER5/area7/**10009028**.htm http://www-01.ibm.com/support/knowledgecenter/POWER5/area7/**10009028**.htm knowledgecenter POWER5 POWER6 **10009028 10009028 10009028 10009028** Explanation SPCN Licensed Internal Code is not...	95%
6	(1000) Reference codes (POWER5 POWER6)	...100091DD 10009023 100091DE 10009024 10009025 10007640 10007641 **10009028** 10009029 1000902D 10009031 10009032 10009033 10009034 10009035...10009022 10009023 10009023 10009024 10009024 10009025 10009025 **10009028 10009028** 10009029 10009029 1000902D 1000902D 10009031 10009031 10009032...	50%

Fig. 3. Search results from the support agent's original query

Number	Document Title	Snippet	Confidence
1	10009028 (POWER7)	http://www-01.ibm.com/support/knowledgecenter/**POWER7**/p7eai/**10009028**.htm knowledgecenter **POWER7 10009028 10009028** Subscribe to this information **POWER7** information **10009028** Explanation SPCN Licensed **Internal Code** is not valid. Response The Licensed **Internal Code** in the primary node is not valid. The code•••	100%
2	(1000) Reference Codes (POWER7)	http://www-01.ibm.com/support/knowledgecenter/**POWER7**/p7eai/1000_info.htm knowledgecenter **POWER7** 10003125 10007602 10009109 10007603 10001500 10001501 10001502...100091DD 10009023 100091DE 10009024 10009025 10007640 10007641 **10009028** 10009029 1000902D 10009031 10009032 10009033 10009034 10009035...1000) Reference codes Subscribe to this information **POWER7** information (1000) Reference codes 100000A0 100000AA 100000AC...	97%
3	10009028 (POWER6)	...-01.ibm.com/support/knowledgecenter/POWER6/area7/**10009028**.htm http://www-01.ibm.com/support/knowledgecenter/POWER6/area7/**10009028**.htm knowledgecenter POWER6 **10009028 10009028 10009028 10009028** Explanation SPCN Licensed Internal Code is not...	67%
4	10009028 (POWER8)	...-01.ibm.com/support/knowledgecenter/POWER8/p8eai/**10009028**.htm http://www-01.ibm.com/support/knowledgecenter/POWER8/p8eai/**10009028**.htm knowledgecenter **POWER8 10009028 10009028 10009028**...	66%

Fig. 4. Improved search results from the log-aided expanded query

3 Log Aided Search

Our approach is based on the observation that effective handling of an IT problem requires understanding the symptoms and causes of the problem and then identifying the relevant solution(s). In the IT support services context, information about symptom(s), cause(s) and solution(s) are usually obtained from diverse sources. Search queries submitted by end users experiencing IT problems (or by technical support agents on behalf of such users) often focus on the symptoms. System logs typically contain messages that are useful in understanding the underlying causes or the broader context of problems. Relevant solutions may be documented or described in corpora of knowledge sources.

The problem we are addressing is: how to guide the search engine in retrieving the most relevant IT support documents containing the solution(s) to the problem (as identified by its symptoms), taking into account the underlying cause(s) or broader context of the problem. Our solution consists of two functionally independent parts: (1) analysis of system logs and (2) topic modeling on the corpora of IT support documents. Implementation of the solution combines the two parts to help formulate more effective queries to the search system. Parsers are used to extract key information from log files, and topic modeling is used to discover hidden "themes" in the corpus. We expect each theme (topic) to be about three aspects of a single type of error, namely: symptom, cause and solution. After clustering corpus terms together into one topic, we expect to find terms related to solutions in the documents, using the symptom terms from the query and the causal terms from the system logs.

The system consists of two different modules, one offline and the other online. Two steps are performed in the offline module. First, documentation of the different types of logs for a domain of interest are analyzed, and parsers are built for each type of log files. The parsers take log files in raw text form as input and output critical information about the error messages in the log. This information can be about root causes, component names, or even possible solutions. Second, a topic model is built on the corpus. Our corpus consists of all the documents indexed in the search system. These documents represent different knowledge sources and cover most of the problems encountered by customers. Therefore, we expect to find words relevant to solutions in the topic model.

The online module is executed when a new query is submitted. It consists of three steps. When an agent submits a query, a case number is also provided to the system. The first step consists of fetching the log files for the case using the case number, and parsing the files to extract key information from each one. The log files may contain information about system profiles and events. We have implemented parsers to extract information from these log files. The extracted information is used to expand the original query, resulting in an intermediate query. The second step is to select terms from the corpus using a topic model-based generative process, forming an expanded query. In the third step, the terms in the expanded query are re-weighted to form the final query.

The five steps (two offline and three online steps) are outlined in an algorithm as follows. In this section, we use the domain of mid-range storage systems[1] to illustrate our solution.

Algorithm 1. System Log Aided AQE Algorithm

Parameter: Number of terms selected from topic model N, number of topics in topic model K, weight for expanded terms from system logs λ

Offline Steps:

- Log Analytics: select set of terms $\{w_1, w_2, \ldots, w_L\}$ from system logs
- Topic Modeling: get probability $p(w|t_i)$ for term w, where t_i is the i^{th} topic

Online Steps:

- Form the intermediate query: expand the original query with terms from the logs
- Corpus term expansion: using the probability of a term given the intermediate query to select the terms from the corpus
- Term weighting and generating the final query

Output: Final query

The process is also illustrated in Fig. 5.

3.1 Offline Step 1: Log Analytics

In order to effectively analyze system logs, we have leveraged the expertise of our agents and incorporated their knowledge in the form of rules for extracting information from log files. Key pieces of information extracted include error messages, machine types, and names of components in abnormal states.

The agents documented 17 unique representative error types that appear in mid-range storage logs. These error types cover failures in five different components: *Controller*, *Enclosure*, *Drive*, *Logical Drive*, and *Arrays*. These components consist of many subcomponents which can fail. We have implemented a parser and analyzer to find evidence related to different types of subcomponent failures that appear in the log files. The common errors that the parser can identify in the log files include, but are not limited to, the following: *Controller Failure*, *Controller Reboot*, *Path Redundancy Loss*, *Impending Drive Failure*, *Cache Disabled*, *Insufficient Cache Backup Capacity*, *Bypassed Drive*, *Batteries Near Expiration*, *Batteries Not Available*, *ESM Failure*, *Power Fan Failure*, and *Individual Drive Degraded Path*.

The above error types can be very informative with regard to root cause analysis of a failure whose symptoms are observed by a customer. If this information can be incorporated into a query, the new (expanded) query will be more

[1] A mid-range storage system's performance and cost lies between expensive, high-end enterprise-class and cheap, low-performance storage systems.

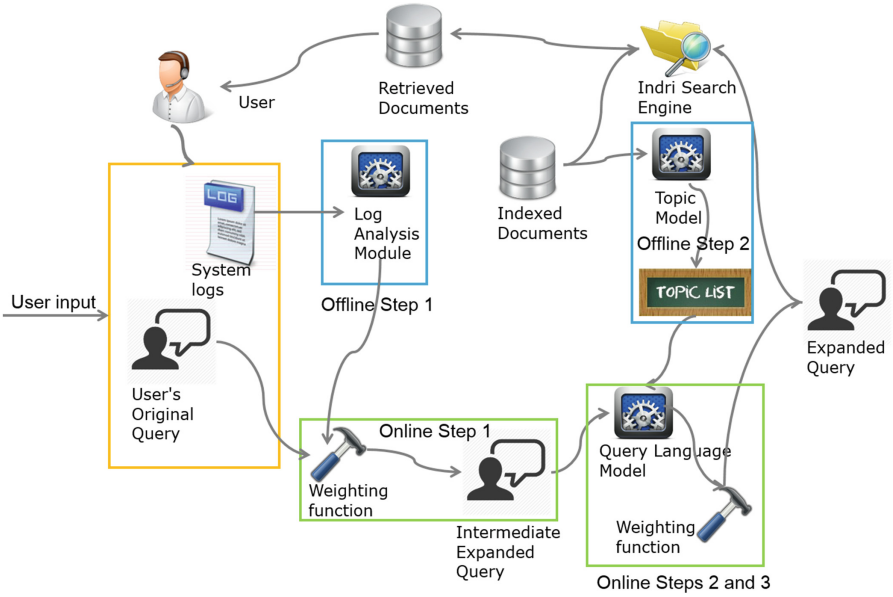

Fig. 5. Solution overview

powerful than the original in its ability to retrieve the most relevant documents which contain the solution to the problem that the customer is experiencing. For example, as we have shown in Sect. 2.2, adding the term "POWER7" from the log to the query "10009028" helps improve the search results.

3.2 Offline Step 2: Topic Modeling

In the first offline module, we incorporate the log terms which are mostly about errors or their root causes. In the second offline module, we further include select terms from the corpus of documents which describe solutions.

The goal of the second offline step is to help find information relevant to the case from the corpus to add to the query, so that the rank of relevant documents can be improved. As discussed above, topic modeling is expected to uncover solutions pertaining to specific types of errors given symptoms and root causes.

Specifically, we run the topic modeling algorithm Latent Dirichlet Allocation (LDA) [4] on all of the knowledge source documents in the corpus. The output is a set of topics, where each topic t is a probability distribution over all the terms in the vocabulary V. Using the topic model, we can get the probability $p(w|t)$, where $w \in V$.

3.3 Online Step 1: Intermediate Query

The online steps are performed each time a new query is processed. Inputs to this step include the text of the original query and system log files for the case.

The log files are fed to the log parser, the output of which identifies (log) terms to be added to the original query for forming an intermediate query. The terms identified from the log files are selected based on rules defined by experienced support agents. The intermediate query contains information from the original query as well as the log files. For example, for a case of mid-range storage systems, the key terms "path redundancy lost 1726" may be selected from the log files, where "1726" is the machine type and "path redundancy lost" is the error. The newly added terms from the logs are weighted, the procedure for which is described later in Sect. 3.5.

3.4 Online Step 2: Expanding Query with Corpus Terms

The goal of this step is to expand the intermediate query further by selecting terms from the corpus. Assume that the intermediate query is **q**. The approach is similar to that of query language models, i.e. we rank each of the corpus terms according to its probability of being generated by the intermediate query. Specifically, we use the following two-step generative process:

- Use the query q to generate a topic t
- Use the topic t to generate a word w

In this way, the probability of a word $w \in V$, given a specific query q can be computed as:

$$p(w|q) = \sum_i p(w|t_i)p(t_i|q)$$

where each t_i is a topic produced by running LDA on the corpus.

In this formula, the probability $p(w|t_i)$ can be computed directly from the output of LDA. Probability $p(t_i|q)$ is the topic mixture inferred by treating the query as a new short document, which can also be acquired directly by LDA inference using the corpus topic model. Therefore, all the probability terms in the right hand side are available from the topic model over the corpus. In this way, the words in vocabulary V can be ranked based on the above probability. Empirically, we choose to add only the top five words. For example, for the mid-range storage case mentioned in Sect. 3.3, the top terms "drive module array host" may be returned from the topic model.

3.5 Online Step 3: Term Weighting and Final Query

The procedure used to weight the terms is also very important. Since the terms from the original query are still the ones best characterizing the user's information need, they are each given a weight of 1. The terms added to form the intermediate query are from the system logs, which should also be directly and equally relevant to the information needed. Therefore, they are each assigned identical weights λ. The third part of the expanded query consists of terms from

the documents, and they are ranked by the probability after the query is seen. Intuitively, they are not so directly relevant to the query, so we just use the ranking probability as the weight. Therefore, we compute the weight of the word w, μ_w as the following:

$$\mu_w = \begin{cases} 1 & \text{if } w \in q_o \\ \lambda & \text{if } w \text{ is from the log} \\ p(w|q) & \text{if } w \text{ is from the corpus} \end{cases}$$

Here q_o is the original query and q is the intermediate query.

4 Experiments

This section describes the verification of our approach using real world data. The data set used is a set of real query sessions by technical support agents using an IBM search system built on top of Indri [13]. A complete session consists of a piece of query text, a case number, a ranked list of returned documents, and the agent user's vote for the query. A vote, submitted by the user, is one of the returned documents that the user believes to be the most relevant to their case. These votes can be used as ground truth for the evaluation. The log files used to form the intermediate query are retrieved by the case number in the session from IBM's Enhanced Customer Data Repository (**ECuRep**[2]). Our first test data set consists of 18 such complete query sessions. The second test data set has 50 incomplete sessions where the log files are missing, but the query text, ranked list of results and votes are available. Note that these two data sets are completely separate from each other. We have used MALLET LDA package [20] in our implementation.

In the experiments, we measure the average rank of voted documents, and the percentage of the query sessions where the voted documents are ranked in the top five (GAIN@5) and the top ten (GAIN@10). For the first test set which has 18 complete sessions, we compare the above metrics using the original query, the expanded query with only the terms from the topic model, the expanded query with only log terms, and the final query expanded using both log terms and the terms from the topic model. For the second set which has 50 incomplete sessions, we only compare the performance of the original query with the expanded query using terms from the topic model.

In Table 1, we show the metrics of the first test set. The expanded queries, using either log terms or terms from the topic model or both, have better performance than the original queries for all three metrics. It also shows that using system logs in retrieving solution documents is critical in IT support services. In addition, it shows that expanding queries with both system logs and topic model can further boost the performance.

Table 2 shows the results for the second test set. The comparison here is between the original queries and the queries expanded with only terms from the

[2] http://www.ecurep.ibm.com.

Table 1. Retrieval performance on complete sessions

Metrics	Original query	AQE with topic model	AQE with system log	AQE with system log and topic model
Average rank	8.44	6.88	5.06	4.88
Gain@5	38.9 %	50.0 %	50.0 %	55.6 %
Gain@10	66.7 %	66.7 %	72.2 %	72.2 %

Table 2. Retrieval performance on incomplete sessions

Metrics	Original query	AQE with topic model
Average rank	8.26	6.36
Gain@5	48.0 %	56.0 %
Gain@10	72.0 %	82.0 %

topic model. In practice, the incomplete sessions represent situations where the log files are not sent by customers, or the log files are in an internal representation which is not easily parsable. As shown in the table, expanding queries using terms in the topic model has helped improve the performance consistently across all three metrics.

5 Related Work

Query expansion is the general process of reformulating a seed query to improve retrieval performance. A query is the primary statement of a user's information need. However, users don't always provide the most optimal queries due to limitations in their scope of knowledge, limited time for query formulation, intrinsic ambiguity of their information need, or the difference between the terms used by users and content providers. Therefore, an enhanced query can improve the performance of the retrieval system.

There have been many studies on the problem of automatic query expansion (AQE), which aims at expanding query terms automatically to better meet the user's information need. One of the most straightforward approaches is to perform linguistic analysis such as stemming [3,12,15], finding synonyms on thesauri such as WordNet [21] and applying other forms of semantic association [18,27]. However, such methods depend largely on the quality of the thesauri and can only utilize the semantic ties between the terms.

Other methods using corpus and query specific techniques utilize statistical features such as co-occurrence information, mutual information, or more complicated measurements of the corpus or query context [17,23,30]. A popular approach is the Relevance-based Model proposed by Lavrenko and Croft [17] along with its variants [6,10,16]. Lavrenko and Croft [17] present a method of estimating the probability of observing a word in the documents relevant to a

query (relevance model) when no training data is available. The idea is to learn the parameters of a relevance model based on the fact that a query is a random sample from the model. The authors also investigate whether topics discovered in the corpus can be used for query expansion. The Relevance-based Model is followed by calculating a topic model based relevance model:

$$p(w|q) = \sum_i p(w|t_i)p(t_i|q)$$

Lavrenko and Croft show that $p(w|q)$ is a very good approximate measurement of the relevance of terms without training data. We borrow this idea in our work, with the key difference being that in our approach, an intermediate query is used in the above equation, which is an expansion of the original user query using terms from system logs. That expansion proves to be crucial for performance improvement in the IT support domain. Other approaches include utilizing information from various sources, such as tagging recommendations based on Wikipedia pages [22].

Another important category of methods is search log analysis, where the idea is to mine associations between different query sessions by a user. Different sessions can be associated chronologically, i.e. two adjacent queries will be associated, or semantically, i.e. two queries sharing terms or semantic overlap will be associated. After queries are associated, both the associated queries themselves [11,14,31] and the results retrieved based on the queries [1,2] can be used to enhance the original query. Previous work has also explored the use of clicked results of associated queries to extract search terms (e.g., [8,24]).

Though there has been extensive research on AQE, to the best of our knowledge, no previous work specifically targets the AQE problem in IT support services and utilizes system logs. A relatively close work is [7] on the use of LDA for web service clustering. However, there is no use of system logs as an important source of information for problem determination, which is a key focus of our work.

6 Conclusion

We have presented a novel systems log-aided method for the domain of IT support services for automatic query expansion, which can identify a set of terms from log files to complement the query terms submitted by an agent addressing a customer problem. Normally, this task is performed manually by support services agents, and they have to acquire years of experience and training to do it effectively. Our experimental results indicate that the systems log-aided method can improve retrieval performance significantly, and yield even better results when combined with topic modeling on the corpus.

In the future, we plan to explore techniques to perform key term selection from log files that are more flexible and domain independent. In addition, we plan to experiment with other topic modeling techniques and compare their performance with LDA.

References

1. Beeferman, D., Berger, A.: Agglomerative clustering of a search engine query log. In: Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2000, pp. 407–416. ACM, New York, NY, USA (2000). http://doi.acm.org/10.1145/347090.347176
2. Billerbeck, B., Scholer, F., Williams, H.E., Zobel, J.: Query expansion using associated queries. In: Proceedings of the Twelfth International Conference on Information and Knowledge Management, CIKM 2003, pp. 2–9. ACM, New York, NY, USA (2003). http://doi.acm.org/10.1145/956863.956866
3. Bilotti, M.W., Katz, B., Lin, J.: What works better for question answering: stemming or morphological query expansion? In: Proceedings of the Information Retrieval for Question Answering (IR4QA) Workshop at SIGIR 2004 (2004)
4. Blei, D., Ng, A., Jordan, M.: Latent dirichlet allocation. J. Mach. Learn. Res. **3**, 993–1022 (2003). http://dl.acm.org/citation.cfm?id=944919.944937
5. Blei, D.M.: Probabilistic topic models. Commun. ACM **55**(4), 77–84 (2012). http://doi.acm.org/10.1145/2133806.2133826
6. Cartright, M.A., Allan, J., Lavrenko, V., McGregor, A.: Fast query expansion using approximations of relevance models. In: Proceedings of the 19th ACM International Conference on Information and Knowledge Management, CIKM 2010, pp. 1573–1576. ACM, New York, NY, USA (2010). http://doi.acm.org/10.1145/1871437.1871675
7. Chen, L., Wang, Y., Yu, Q., Zheng, Z., Wu, J.: WT-LDA: user tagging augmented LDA for web service clustering. In: Basu, S., Pautasso, C., Zhang, L., Fu, X. (eds.) ICSOC 2013. LNCS, vol. 8274, pp. 162–176. Springer, Heidelberg (2013)
8. Cui, H., Wen, J.R., Nie, J.Y., Ma, W.Y.: Query expansion by mining user logs. IEEE Trans. Knowl. Data Eng. **15**(4), 829–839 (2003). http://dx.doi.org/10.1109/TKDE.2003.1209002
9. ElasticSearch: https://www.elastic.co
10. Halpin, H., Lavrenko, V., St, C.: Relevance feedback between hypertext search and semantic search. In: Proceedings of the Semantic Search Workshop at the World Wide Web Conference (2009)
11. Huang, C.K., Chien, L.F., Oyang, Y.J.: Relevant term suggestion in interactive web search based on contextual information in query session logs. J. Am. Soc. Inf. Sci. Technol. **54**(7), 638–649 (2003). http://dx.doi.org/10.1002/asi.10256
12. Hull, D.A.: Stemming algorithms: a case study for detailed evaluation. J. Am. Soc. Inf. Sci. **47**(1), 70–84 (1996). http://dx.doi.org/10.1002/(SICI)1097--4571(199601)47:1⟨70:AID-ASI7⟩3.3.CO;2-Q
13. Indri: http://www.lemurproject.org/
14. Jones, R., Rey, B., Madani, O., Greiner, W.: Generating query substitutions. In: Proceedings of the 15th International Conference on World Wide Web, WWW 2006, pp. 387–396. ACM, New York, NY, USA (2006). http://doi.acm.org/10.1145/1135777.1135835
15. Krovetz, R.: Viewing morphology as an inference process. In: Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 1993, pp. 191–202. ACM, New York, NY, USA (1993). http://doi.acm.org/10.1145/160688.160718
16. Lavrenko, V., Choquette, M., Croft, W.B.: Cross-lingual relevance models. In: Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2002, pp. 175–182. ACM, New York, NY, USA (2002). http://doi.acm.org/10.1145/564376.564408

17. Lavrenko, V., Croft, W.B.: Relevance based language models. In: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2001, pp. 120–127. ACM, New York, NY, USA (2001). http://doi.acm.org/10.1145/383952.383972
18. Liu, Y., Li, C., Zhang, P., Xiong, Z.: A query expansion algorithm based on phrases semantic similarity. In: 2008 International Symposiums on Information Processing (ISIP), pp. 31–35, May 2008
19. Lucene: https://lucene.apache.org/
20. McCallum, A.K.: Mallet: a machine learning for language toolkit (2002). http://mallet.cs.umass.edu
21. Navigli, R., Velardi, P.: An analysis of ontology-based query expansion strategies. In: Workshop on Adaptive Text Extraction and Mining, Cavtat Dubrovnik, Croatia, 23 September 2003
22. Oliveira, V., Gomes, G., Belém, F., Brandão, W., Almeida, J., Ziviani, N., Gonçalves, M.: Automatic query expansion based on tag recommendation. In: Proceedings of the 21st ACM International Conference on Information and Knowledge Management, CIKM 2012, pp. 1985–1989. ACM, New York, NY, USA (2012). http://doi.acm.org/10.1145/2396761.2398557
23. Park, L.A.F.: Query expansion using a collection dependent probabilistic latent semantic thesaurus. In: Zhou, Z.-H., Li, H., Yang, Q. (eds.) PAKDD 2007. LNCS (LNAI), vol. 4426, pp. 224–235. Springer, Heidelberg (2007)
24. Riezler, S., Vasserman, A., Tsochantaridis, I., Mittal, V., Liu, Y.: Statistical machine translation for query expansion in answer retrieval. In: Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL2007). Prague, Czech Republic (2007). http://www.stefanriezler.com/PAPERS/ACL07.pdf
25. Solr: http://lucene.apache.org/solr/
26. Sphinx: http://sphinxsearch.com/
27. Symonds, M., Zuccon, G., Koopman, B., Bruza, P., Sitbon, L.: Term associations in query expansion: a structural linguistic perspective. In: Proceedings of the 22nd ACM International Conference on Information and Knowledge Management, CIKM 2013, pp. 1189–1192. ACM, New York, NY, USA (2013). http://doi.acm.org/10.1145/2505515.2507852
28. Systems, I.P.: www.ibm.com/systems/power/
29. Xapian: http://xapian.org/
30. Xu, J., Croft, W.B.: Query expansion using local and global document analysis. In: Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 1996, pp. 4–11. ACM, New York, NY, USA (1996). http://doi.acm.org/10.1145/243199.243202
31. Yin, Z., Shokouhi, M., Craswell, N.: Query expansion using external evidence. In: Boughanem, M., Berrut, C., Mothe, J., Soule-Dupuy, C. (eds.) ECIR 2009. LNCS, vol. 5478, pp. 362–374. Springer, Heidelberg (2009)