Enhancing passage retrieval in log files by query expansion based on explicit and pseudo relevance feedback

Hassan Saneifar, Stéphane Bonniol, Pascal Poncelet, Mathieu Roche

Enhancing Passage Retrieval in Log Files by Query Expansion Based on Explicit and Pseudo Relevance Feedback

Hassan Saneifar
LIRMM - University Montpellier 2 - CNRS. Montpellier, France
Satin Technologies, Montpellier, France
saneifar@lirmm.fr
Stéphane Bonniol
Satin Technologies, Montpellier, France
stephane.bonniol@satin-ip.com
Pascal Poncelet
LIRMM - University Montpellier 2 - CNRS. Montpellier, France
Pascal.Poncelet@lirmm.fr
Mathieu Roche
LIRMM - University Montpellier 2 - CNRS. Montpellier, France
Mathieu.Roche@lirmm.fr

Abstract

Passage retrieval is usually defined as the task of searching for passages which may contain the answer for a given query. While these approaches are very efficient when dealing with texts, applied to log files (i.e. semi-structured data containing both numerical and symbolic information) they usually provide irrelevant or useless results. Nevertheless one appealing way for improving the results could be to consider query expansions that aim at adding automatically or semi-automatically additional information in the query to improve the reliability and accuracy of the returned results. In this paper, we present a new approach for enhancing the relevancy of queries during a passage retrieval in log files. It is based on two relevance feedback steps. In the first one, we determine the explicit relevance feedback by identifying the *context of the requested information* within a learning process. The second step is a new kind of pseudo relevance feedback. Based on a novel term weighting measure it aims at assigning a weight to terms according to their relatedness to queries. This measure, called TRQ (Term Relatedness to Query), is used to identify the most relevant expansion terms.
The main advantage of our approach is that is can be applied both on log files and documents from general domains. Experiments conducted

on *real data* from logs and documents show that our query expansion protocol enables retrieval of relevant passages.

1 Introduction

Passage retrieval, representing an important phase in question answering and information locating methods, is the task of searching for passages which may contain the answer for a given question. As an accurate and reliable definition, a passage is a fixed-length sequence of words which can begin and end anywhere in a document Ofoghi et al. (2006). Passage retrieval has been extensively investigated since late 1980 and early 1990's O'Connor (1975), Wade and Allan (2005).

Text retrieval methods are typically designed to identify whole documents that are relevant to a query. In these approaches a query is evaluated by using the words and phrases in each document (i.e., the index terms) to compute the similarity between a document and a query Kaszkiel and Zobel (1997). This means that by means of text retrieval methods, we are not able to locate the seeking information in documents, but to find only the relevant documents (i.e., documents containing the seeking information). As an alternative, we have passage retrieval which makes it possible to locate the requested information within a document. In this context, each document is seen as a set of passages, where a passage is a contiguous block of text. Thus, in order to retrieve passages containing the sought information, the similarity of each passage to a query is calculated. In fact, passage retrieval can be considered as an intermediate between document retrieval and information extraction. Therefore, the main objective is to locate sought information within documents and thus reduce the search space wherein we will look to extract the exact information.

The main differences between different passage retrieval systems are the way that they select the passages, that is to say, what they consider as a passage? how to discover its boundaries? and how to evaluate its relevancy? Tellex et al. (2003a). In most passage retrieval methods, we distinguish two main phases: (1) *passage segmentation*, (2) *passage ranking (scoring)*. Passage segmentation is the task of determining text segments in documents which are considered as candidate passages. In passage segmentation, the issue is how to define the passage boundaries? and how to recognize them in a corpus? While passage ranking assesses the relevancy of passages according to a given query. By means of scoring and measuring functions, one evaluates how much a passage is correlated to a query. Based on passage segmentation time, we have *index-time* passaging methods versus *search-time* passaging ones.

In index-time passaging, documents are split into text chunks (segments) before indexing them Tiedemann and Mur (2008). This means that the cutting documents to several segments (i.e., passages) is carried out before analysing queries. Thus, passage types and their size are independent from queries. In these methods, the passage ranking is performed later by evaluating the relevance of each candidate passage according to a given query. Contrary to the

index-time passaging, in search-time methods, one determines passages along with searching for requested information Llopis et al. (2002). In these approaches, the passage segmentation is dynamically carried out along with passage ranking based on query properties. This means that the boundaries of a passage are dynamically determined ensuring that the best relevance score is obtained for the passage. The index-time methods allow a quicker calculation; nevertheless, the second methods allow different segmentation models in accordance with query properties Llopis et al. (2002).

In this work, we investigate a study on *passage retrieval* and *boosting its performance* within a *question answering* system on a specific domain Ligozat et al. (2012). Here we deal with a particular type of complex textual data, i.e., log files generated by Integrated Circuit (IC) design tools. Since IC design tools runs a long time in batch mode in Industry, the generated log files are often the user's sole feedback. Users constantly need to check progress by listing these logs. These log files are digital reports on configurations, conditions, and states of systems. In this domain, to ensure the design quality, there are some quality check rules which should be verified. These quality check rules are usually formulated in the form of *natural language questions* (eg., "*Capture the total fixed cell STD*" or "*Captures the maximum Resistance value*"). Verification of these rules is principally performed by analysing the generated log files. In the case of large designs that the design tools may generate megabytes or gigabytes of log files each day, the problem is to wade through all of this data *to locate the critical information* that we need to verify the quality check rules.

In order to locate information in log files, we are mainly interested in passage retrieval. We have previously proposed an approach to split log files into text chunks (segments) based on the their logical structure Saneifar et al. (2011). This means that we first perform an *index-time passage segmentation*. The justification of this choice is based on performance gain and the fact that the structure of log files provides significant information for a relevant query independently of the segmentation. Now the challenge is to find log file segments that may contain the requested information by means of passage scoring. But the particularities of such textual data (*i.e. log files*) significantly impact the accuracy and performance of passage retrieval in this context.

Due to the fact that log files are multi-source and multi-vocabulary data, the main challenge is the existing gap between vocabulary of queries and those of log files. We call this problem mismatch vocabularies. This issue is also noted in some other work. For example, the authors of Buscaldi et al. (2010) note that the answer to a question may be unrelated to the terms used in the question itself, making classical term-based search methods useless Buscaldi et al. (2010). Because the user's formulation of the question is only one of the many possible ways to state the seeking information, there is often a discrepancy between the terminology used by the user and the terminology used in the document collection to describe the same concept Van der Plas and Tiedemann (2008). This issue is highlighted in the case of log files which are by default multi-vocabulary data. Also, we have to deal with other challenges in passage retrieval

from log files. We can briefly note the lack of data redundancy and thus lack of answer (information) repetitions, lack of paraphrasing or surface patterns in log files, and the lack of semantic resources. We discuss and develop all these issues as well as mismatch vocabularies problem in Section 2.

Taking all these difficulties into account, we finally choose Query Expansion in order to improve the performance of passage retrieval in log files and overcome this domain problems notably mismatch vocabularies. Query expansion (or query enrichment [1]) attempts to improve retrieval performance by reformulating and adding new correlated terms to queries. In general the idea is to add more terms to an initial query in order to disambiguate the query and solve the possible term mismatch problem between the query and the relevant document Keikha et al. (2011). Here we present a query expansion approach using two novel relevance feedback levels.

The relevance feedback process, introduced in the mid-1960s is a controlled, automatic process for query reformulation, that can prove unusually effective Salton and Buckley (1997). Relevance feedback is a powerful technique whereby a user can instruct an information retrieval system to find additional relevant documents by providing relevance information on certain documents or query terms Selberg (1997). The basic idea behind relevance feedback is to take the results initially returned from a given query and to use information about whether or not those results are relevant to reformulate a new query.

Relevance feedback refers to an interactive process that helps to improve the retrieval performance. This means that when a user submits a query, an information retrieval system would return an initial set of result documents and then ask the user to judge whether some documents are relevant or not; after that, the system would reformulate the query based on the user's judgements, and return a set of new results. There are principally three types of relevance feedback: explicit, pseudo (blind), and implicit.

When there are no real relevance judgements available, alternatively, pseudo relevance feedback may be performed, which simply assumes that a small number of top-ranked documents in the initial retrieval results are relevant and then applies relevance feedback. Besides, somewhere in between relevance feedback and pseudo relevance feedback is implicit feedback, in which a user's actions in interacting with a system are used to infer the user's information need Lv and Zhai (2009). All these methods will be detailed in Section 3.

Our query expansion based on relevance feedback involves two levels. In the first one, we implement an explicit relevance feedback system. The feedback is obtained from a training corpus within a *supervised learning approach*. We propose a new method for learning the *context* of questions (queries), based on the "*lexical world*" notion. Then, the contexts of questions are used as relevant documents wherein we look for expansion terms.

The second phase consists in a novel kind of pseudo relevance feedback Lee

[1] We use query expansion and query enrichment interchangeably in this paper.

and Croft (2013). Contrary to most pseudo relevance feedback methods considering the initial top-ranked documents as relevant, our method is based on a *new term weighting function*, called TRQ[2], which gives a score to terms of corpus according to their *relatedness* to the *query*. Indeed, we present the TRQ measure as an original term weighting function which aims at giving a high score to terms of the corpus which have a significant probability of existing in the relevant passages[3]. Terms having the highest TRQ scores are selected as expansion terms.

We also evaluate the application of our approach in general domains. We thus use the documents used in TREC[4] evaluation campaigns. We study the difference between the application of our approach in specific and general domains. We show that our approach gives *satisfactory* results on *real data* from the industrial field as well as general domains.

In Section 2, we present the main characteristics of log files which rise some challenges in passage retrieval. Existing work about passage retrieval systems and the application of relevance feedback in the query expansion are presented in Section 3. Section 4 presents some notions used in our query expansion approach and also the first level of query enrichment. In Section 5, we develop our query expansion approach by presenting our novel term weighting function. Section 6 is devoted to developing the application of our approach in open domains. Finally, the experiments on real data are presented in Section 7.

2 Difficulties in Passage Retrieval in Log Files

The particular characteristics of logs, described below, rise some challenges in passage retrieval and information extraction in log files. Here, by presenting these challenges and difficulties, we explain how they led us to query expansion as a solution.

First, we focus on problems arising from the multi-source aspect of log files. In the design of Integrated Circuits, different design tools can be used at the same time, while each tool generates its own log files. Therefore, although the logs of the same design level contain the *same* information, their *structure* and *vocabulary* can vary significantly *depending* on the *design tool used*. On the other words, each design tool has its own vocabulary to report the same information. This implies that queries which are expressed using a vocabulary could not necessarily correspond to the vocabulary of all tools, or the query terms do not necessarily exist in the corresponding answers. We explain this issue, called here mismatch vocabularies, by the help of an example.

[2]Term Relatedness to Queries.
[3]Passages containing answers to questions.
[4]http://trec.nist.gov/

Consider the sentence "`Capture the total fixed STD cell`" as a given query and the two log files, log_A and log_B generated by two different tools, as the data resources wherein we look for answers. The answer to this question, in log_A, is expressed in the third line of its following segment.

103	Std cell utilization:	0.93% (449250/(48260400-0))
104	Chip area:	48260400 sites, bbox (210.00 210.00 4140.00 4139.60) um
105	**Std cell area:**	**449250 sites, (non-fixed:449250 fixed:0)**
106		31625 cells, (non-fixed:12345 fixed:130)
107	Macro cell area:	0 sites

While the answer, in log_B, is expressed in the last line of its following segment.

66	*** Clustered Preplaced Objects ***
67	# of preplaced io is: 18
68	# of preplaced hard macro is: 0
69	**# of preplaced standard cell is: 24678**

As shown above, the same information in two log files produced by two different tools is represented by different terms. The keywords of the question (i.e. `fixed, Std` & `cell`) exist in the answer extracted from log_A while the answer from log_B contains only the word "`cell`". Insofar as there is a *dictionary* associating the word "`Std`" with "`standard`", we can also take the word "`standard`" into account. However, if we make a query by using these two keywords and send it to an IR system on log_B, irrelevant passages of this log file are retrieved.

For example, here we retrieve the following passage located in lines 57 to 60 in log_B wherein the two keywords occur more frequently.

57	*** Clustered FPlan Seeds ***
58	# of standard cell seeds is: 0
59	# of soft standard module seeds is: 19
60	**# of instance cell group seeds is: 8**

This is because there is a significant *vocabulary mismatch* problem among log files log_A and log_B. This means that the query is expressed using the vocabulary of log_A which is different from the vocabulary of log_B. For a given question, *relevant answers* found in the logs of *some tools* do *not* necessarily contain the *keywords* of the question. Therefore, the initial query (*created by taking the keywords of the question*) may be relevant to logs generated by a tool (e.g., here log_A) but irrelevant to logs generated by another tool (e.g., here log_B) whereas we aim to answer questions regardless of the type of tools that generate the log files.

The *existence of question keywords* (or their syntactic variants) in a *passage* is an important factor to assess the *relevance* of the passage. Approaches based on the notion of common terms among questions and passages are detailed in

Sect. 3. We note that the vocabulary changes in log files are not due to lexico-semantic variation of terms, but usually different instance/notion nomination or standardization. Moreover, there is not any relevant domain dictionary or semantic resource in this context in order to automatically identify corresponding terms.

Actually, the performance of a QA system depends largely on *redundant occurrences of answers* in the corpus in which answers are sought Brill et al. (2001)Lin (2007). Tiedemann also argues in Tiedemann (2007) that *high redundancy* is desired in order to get as many relevant passages as possible to make it easier for the answer extraction modules to spot possible answers. He notes that high redundancy reduces the likelihood of selecting the wrong answer string by providing stronger evidence for the correct ones. Regarding the importance of answer redundancy, an analysis of TREC9 experimental results in Clarke et al. (2001) indicates that redundancy played an important role in the selection of answer fragments. Their experiments demonstrate that redundancy is an effective method for ranking candidate answers. In the same way, some evidence are provided in Light et al. (2001a) that show the precision of a QA system depends largely on the redundancy of answer occurrences in the corpus.

But information is rarely repeated in the log files of IC design tools. This means that for a question, there is only *one occurrence of the answer* in the corpus and thus one relevant passage containing the answer. Hence, methods based on exploiting the redundancy of answers in a corpus are not relevant. In such situation, a highly accurate passage retrieval module is required because the answer occurs in a very small set of passages Ferrés and Rodríguez (2006).

Taking all these evidences into account, the fact that there is only one occurrence of answer in log files, rises important challenge in information locating in log files. Our passage retrieval component has to achieve a high precision without being dependant on the answer redundancy.

Moreover, we note that if we do not retrieve the only existing relevant passage for a question among the top ranked candidate passages, we will not be able to extract answer at all. Whereas, when there are multiple relevant passages, we have more chance to retrieve at least one of the relevant ones among the top tanked candidates. Thus, it is necessary to find other solutions in order to enhance the performance of passage retrieval in the context of log files.

Most QA systems for a given question, extract a large number of passages which likely contain the answer. But an important point in QA systems is to limit, as much as possible, the number of passages in which the final answer extraction is performed. If a passage retrieval module returns too many irrelevant passages, the answer extraction module is likely to fail to pinpoint the correct answer due to too much noise Cui et al. (2005). Since we are situated in a very specialized domain, high precision in the final answers (i.e. the percentage of correct answers) is a very important issue. This means that our passage retrieval

system has to classify relevant passages (based on a relevance score) in the first positions among all retrieved candidate passages.

In addition, as previously said, design tools change over time, often unexpectedly. Therefore, the *data format* in the log files changes, which makes automatic data management difficult. Moreover, the language used in these logs is a difficulty that impacts information extraction methods. Although the language used in these logs is English, their contents do not usually comply with "*conventional*" grammar. In the processing of log files, we also deal with multi-format data: textual data, numerical data, alphanumerical, and structured data (e.g., tables and data blocks).

We therefore suggest expansion of initial queries in order to make them relevant to all types of logs (*generated by any kind of design tools*). By query expansion, we consider to enrich an initial query by integrating semantically similar terms coming from the vocabulary of different log files. That enables us to have a passage retrieval system which works well regardless of the type of supplied log files. This query expansion also improves the performance of passage retrieval in terms of precision as we obtain more relevant queries.

3 Related Work

In this section, we study the background work regarding passage retrieval and query expansion. We first introduce the main methods used in passage retrieval. Then, we focus on two main directions in query expansion. We finally provide a discussion about the relevance of cited methods in the context of log files.

3.1 Passage Retrieval and Locating Answers

Most passage retrieval algorithms depend on occurrences of query keywords in the corpus Ofoghi et al. (2006). We call this kind of methods statistic as they principally use metrics based on the occurrence numbers of query keywords in passages. To find relevant passages, Lamjiri et al. (2007) evaluate each passage using a scoring function based on the coverage of "question keywords" which also exist in the passage. In this category, we can note the overlap algorithm used in MITRE system Light et al. (2001b).

The term-density is also used in many work to retrieve the relevant passages. For example, in MultiText Clarke et al. (2000) passage retrieval algorithm, which is a density-based one, retrieves short passages containing many terms with high IDF values [5]. In term-density based methods, the density is first determined by the extraction of question objects: lemmas of words, the types of named entities, and the type of answer to search. Then, for each element, an average distance is calculated between the current object and other objects of the question. The

[5]The IDF (Inverse Document Frequency) measure is explained in section 4.2.

distance between terms is usually defined as the number of words occurring between them Monz (2003). This distance is then used to calculate the density score to identify the passage which is related to the question Gillard et al. (2006).

Other passage retrieval systems, such as those employed in SiteQ Lee et al. (2001) and IBM Ittycheriah and Roukos (2002), are density-based as they take into account the distances between question terms in the candidate passages. IBM's passage retrieval algorithm Ittycheriah and Roukos (2002) computes a series of distance measures for the passage notably "matching word measure", "mismatch word measure", and "cluster word measure". The first one computes the sums of IDF values of terms (or their synonyms) in the query whose appear in the passage. Mismatch word measure also functions in the same way but for query terms that do not appear in the passage. Finally, the "cluster word measure" counts the number of words that occur adjacently in both the question and the passage. These various measures are linearly combined to give the final score for a passage Tellex et al. (2003b).

In the category of statistic methods, there are also methods which take different variations of keywords into account. For example, the use of morphological and semantic variants of query keywords is studied in Chalendar et al. (2002). Kosseim and Yousefi (2008) presents different approaches to take also semantic variations into account in order to complement the syntactic variants. Semantic variations are usually determined by using semantic resources like thesauri and based on the lexical relations like synonymy.

Question reformulation based on *surface patterns* is a standard method used to improve the performance of QA. The technique is based on identifying various ways of expressing an answer given a natural language question Kosseim and Yousefi (2008). For example, for a question like "*Who founded the American Red Cross?*", QA systems based on surface patterns seek reformulations like "*the founder of the American Red Cross is X*" or "*X, the founder of the American Red Cross*". The question reformulation using surface patterns is also exploited in TREC9 and TREC10.

3.2 Query Expansion

QA systems also use query expansion methods to improve the retrieval performance. Delphine Bernhard argues in Bernhard (2010) that query expansion attempts to solve the vocabulary mismatch problem by adding new semantically related terms to the query. The query expansion is also studied among TREC participants. Manning et al. (2008) presents two types of Query Expansion methods: (1) global and (2) local techniques. In the following paragraphs, we introduce some background work in both categories.

The global methods are based on external lexical-semantic resources such as ontologies. Most of QA systems use a kind of knowledge base in order to identify different lexical variations of question terms Yang and Chua (2002). These

variations are used in order to better analyse and retrieve relevant passages and extract answers. At this level, the systems typically involve morphological and semantic knowledge from existing electronic dictionaries and lexical resources such as WordNet [6]. Pasca and Harabagiu (2001) present an approach in which queries are expanded using morphology, lexical derivations and semantic equivalents, a kind of highly-controlled synonymy based on hand-maintained resources. Agichtein et al. (2001) propose a query expansion based on the Web resources. They take a number of keywords from the original question to form queries, which are expanded with phrases that are likely to occur in a declarative sentence that contains an answer.

Relevance Feedback The local methods are known as relevance feedback. As explained before, they use the results obtained for an original query in order to expand it. The application of relevance feedback depends largely on the information retrieval model. Most relevance feedback methods are based on the notion of Rocchio's ideal query Rocchio (1971). The notion of Rocchio's ideal query can be described as a query that has maximum similarity to relevant documents and minimum similarity to irrelevant documents.

Xu and Croft introduce in Xu and Croft (2000) a query expansion method based on both relevance feedback (local) and global method which is called local context analysis. That is, the selection of expansion terms is enhanced by considering concepts in the top-ranked documents that frequently co-occur with "many" query terms in the whole collection. Compared to classic relevance feedback, candidate expansion terms are more relevant to the query, as they have been observed to co-occur frequently in all documents Wu et al. (2011).

Carpineto et al. (2001) propose an other method that uses information theory measures for relevance feedback based query expansion. The main hypothesis of this method is that the difference between the distribution of terms in a set of relevant documents and the distribution of the same terms in the overall document collection reveals the semantic relatedness of those terms to the query.

In most relevance feedback systems, we first use the initial query to retrieve some documents. Among the initially retrieved documents, those indicated by a user as relevant (explicit feedback) or only a few top-ranked ones (blind feedback) are analysed to extract expanding terms.

In an explicit system, feedback is obtained by some explicit evidence showing the relevance of a document. This type of feedback is explicit only when the assessors know that the feedback provided is interpreted as relevance judgements. For implicit feedback, there is not any direct information indicating the relevance of a document. The relevance of documents is inferred from user behaviours like the time spent on a document. This kind of feedback also needs the indirect user or expert intervention. However, pseudo relevance feedback, also known as blind relevance feedback, provides a method for automatic local analysis. In fact, when there are no real relevance judgements available or user

[6] http://wordnet.princeton.edu/

interaction cannot be studied, alternatively, pseudo relevance feedback may be performed. Usually, in blind relevance feedback, it is simply assumed that a small number of top-ranked documents in the initial retrieval results are relevant and finally the relevance feedback is based on this assumption Lv and Zhai (2009).

3.3 Discussing the background methods

We have identified the main directions and methods in passage retrieval and query expansion according to the background work in these domains. In the following paragraphs, we discuss the relevance of these methods in the context of log files.

NLP & Statistic Methods Despite the satisfactory results achieved by using surface patterns and syntactic variants in the mentioned work, these methods are irrelevant in the context of log files. Indeed, the main reasons for the irrelevancy of such methods are related to the fact that an answer is not reformulated in different ways in a corpus of log files. Moreover, there is a *lack of redundancy of answers* in log files. In addition, there are several technical and alphanumeric keywords in the domain of log files for which the use of syntactic or semantic variants appears to be complex or meaningless. Note that identification of lexico-semantic variations of terms usually requires an external knowledge base like ontologies or dictionaries which are not available in the context of studied log files.

Global Query Expansion based Methods Regardless of the performance issue, in such methods, one has to tackle problems of semantic ambiguity, which explains why *local analysis* has been shown to be generally more effective than *global analysis* Xu and Croft (1996). Moreover, as mentioned above, in order to determine the semantic relations we need semantic resources like WordNet or a controlled vocabulary and categorised named entity list. In many specific domains, like in our case, there are no external lexical-semantic resources and their creation is a time-consuming task requiring domain expert knowledge. In addition, we could not supply a synonymity or lexical knowledge within our system and be sure that it covers all vocabularies may be used in this domain or by all types of log files. Thus, we decided to select expansion terms based on the on-going log files which are used and processed in the execution time in the system.

Local Query Expansion based Methods In an explicit feedback case, in order to perform an initial retrieval, we need to index the training corpus, which can be time-consuming for industrial usage. In addition, we noticed that the (initially retrieved) passages are long enough to contain terms which are informative in the passages, *but* they are not really correlated to the query terms.

That is why we propose to identify small fragments of log files which represent the context of queries to reduce the search space. We develop the query context identification in Section 4.

There are also some difficulties in using classic methods of blind relevance feedback. Indeed, in log files, we have observed that passages retrieved using the initial queries are not always relevant. This issue, i.e. the top-ranked initially retrieved documents are not always relevant, is also noted in other work like Li and Zhu (2008). In our context, queries are actually questions predefined without caring about the type of data resources (log files) in which one looks for answers. That is why we could not consider the initially top-ranked passages as relevant when we do not have any external knowledge about their relevancy (blind case). That led us to define and propose a new way to select the expansion terms when the explicit relevance feedback does not exist. Our method which is based on a novel term scoring function, called TRQ measure, gives a high score to terms related to a query in a corpus. This new measure and our second phase of query enrichment is developed in Section 5.

As shown in Xu and Croft (2000), considering also the co-occurrence factor in local query expansion methods give better results. This feature is also reflected in our query expansion method by taking the dependency between terms and queries into account.

4 Passage Retrieval Enhancing by Query Expansion.

4.1 Global process

Our query enrichment approach is based on a *context learning* process and is associated with an *original term weighting function*. Our protocol of context learning is designed to determine the context of a given question by analysing the terms [7] co-occurring around the keywords of the question in the corpus. The new scoring function proposed here identifies terms related to the answers. The architecture of our approach consists of three main modules:

1. Enrichment of the initial query by context learning (explicit feedback)
2. Enrichment by terms which are likely related to the answer (pseudo feedback)
3. Passage retrieval using the enriched queries

The first module is considered as a kind of *explicit* relevance feedback. The goal is to enrich initial queries extracted from questions in natural language so that they will be relevant to all types of log files generated by different tools. At this step, after *identifying the context of a question* within a supervised learning process, we enrich the initial query by the most significant terms extracted from

[7] In this paper, the word "term" refers to both words and multi-word terms of the domain.

the context.

The second module is designed for a second query expansion in order to obtain higher accuracy. At this phase, we aim at identifying terms which are likely related to the query in order to integrate them into the initial one. The objective is to make queries more flexible and relevant just before the passage retrieval process while there is significant difference and heterogeneity between the vocabulary of the training corpus and that of test corpus. Indeed, within the learning process, the enriched query is relevant to the most kinds of log files. However, in industrial applications, we can be faced with some cases where a second query expansion phase can improve the results on log files that are significantly different from those used in the training corpus. This can happen, for example, when a user supplies some log files as resources which differ considerably from those used for training. By means of the second enrichment, we avoid a new constitution of the training corpus and learning process when the system unexpectedly needs to process log files that are significantly different from those used in the training. Moreover, while expert knowledge is not available to build the training corpus and obtain explicit feedback, the second module can be used independently to expand the initial queries. Nevertheless, we point out that the second phase is not designed to replace the learning process (first enrichment phase).

In the third module, we seek the relevant passages in the logs generated by a tool that differs from that which has been used in the learning phase. That is, we have two different corpora of logs. The first one (called the *training corpus*) is used in the learning phase and the second one (called the *test corpus*) is used to retrieve relevant passages according to given questions. The logs of the test corpus have structures and a vocabulary that differ from the logs of the training corpus. We note that this part of our approach is different from machine learning based approaches. In fact, the training corpus is used to select the explicit relevance feedback from. This means that the training corpus is not a pre-annotated corpus. Moreover, the main idea in our approach is to learn the context of questions using a training corpus. The idea is not to learn rules based on which we can retrieve relevant passages in other corpus.

In our approach, we look for specialized context (hereafter called "*lexical world*") of question keywords. In order to characterize and present the specialized context of keywords, we use the terminological knowledge extracted from logs. Before explaining the query enrichment processes, we develop the concept of "lexical world" and the use of terminological knowledge in the following subsections.

Lexical World

The lexical world of a term is a small fragment of a document in which the term is seen. By determining the *lexical world* of a term in a document, we identify

terms that *tend* to *appear around it* in that document. Actually, terms located around a term (within a small fragment) generally have strong semantic and/or contextual relations. We do not put any limit on the size of lexical worlds (eg., a few lines, a few words, etc.) as it has to be determined pragmatically based on the type of documents. Considering n words which surround a word as its context is also used in some proximity-based methods Van der Plas and Tiedemann (2008). However our work differs first in the way we characterize the context of keywords and second in how we finally determine the context of questions wherein the expansion terms are selected.

In order to present the lexical world of a term, several solutions are possible. As a first solution, we characterize the lexical world of a term by a set of "single words" (*also called bag of words*) which are located around the term and present "Noun", "Verb", or "Adjective" parts of speech. As a second solution, the lexical world of a term is presented by co-occurring words like bi-grams of words (i.e., any two adjacent words) or multi-word terms (a few adjacent words, following a pre-defined part-of-speech pattern, which also form a significant term) which are seen around the term. We detail this point in the next section.

Terminological Knowledge

As mentioned above, the lexical worlds can be characterized in different ways: by "single words", "multi-word terms", or "bi-grams of words". According to our experiments, multi-word terms and words are more representative than bi-grams of words. Hence, we create two types of lexical world: (1) consisting of single words and (2) comprising multi-word terms and single words. In order to determine the multi-word terms, we extract the terminology of logs by our Exterlog approach Saneifar et al. (2009). This method, adapted to the specific characteristics of logs, extracts multi-word terms according to syntactic patterns in the log files. To choose the *relevant and domain-specific terms*, we also propose a terminology validation and filtering protocol in Saneifar et al. (2011). We finally obtain the valid and relevant multi-word terms (e.g., "scan chain", "clock pulse", "design flow") to characterize the lexical worlds.

4.2 Query Enrichment by Context Learning

We explain here the first module of our query enrichment approach. For a given question, we initially aim to learn the context of the question and characterize it by its most significant terms. Since these terms represent at best the context of the question, it is expected that passages corresponding to the question share some of these terms regardless of the type of log files.

To identify the context of a question, we lean on the lexical worlds of its keywords. Hence, for every keyword, we look for its lexical world in the training corpus by identifying the small fragment of log files wherein the keyword is

located[8]. These fragments are identified based on the recognition of logical units in log files. Then we characterize the extracted fragments to finally obtain the corresponding lexical worlds of the keyword.

At this stage, we aim at identifying the *most relevant lexical world* among all extracted ones which *presents at best the question context*. Thus, we use a vector space information retrieval model based on TF-IDF measure to order the lexical worlds of a question's keywords based on their relevance to the question. TF-IDF is a statistical weighting function which assesses how important a word is in a document of a corpus Salton and McGill (1986). The values depend on the number of occurrences of words (TF) in a document, and also on the number of documents containing the word (IDF).

Once the most correlated lexical worlds are retrieved, a user assesses the relevance of the lexical worlds by choosing the lexical world which represents the question context at best among the retrieved lexical worlds. The chosen lexical world is seen as an explicit relevance feedback. We then identify the *most representative terms* of the chosen lexical world. For this purpose, we use the term indexes (obtained by tf-idf) to measure the importance of them in the context. Finally, we select the n terms having the highest tf-idf scores Salton and Buckley (1987) as expansion terms.

Since the next phase of query enrichment based on the novel weighting function deserves to be fully developed and studied in detail, we fully devote Section 5 in this regard. Therefore, we explain, in the following subsection, how we look for relevant passages once the initial queries are enriched.

4.3 Passage Retrieval in Log Files

Here we detail the process of finding relevant passages in the test corpus of log files. First, we segment the logs of the test corpus. Segmentation is performed according to the logical units of log files like data blocks, tables, etc. Each segment is seen as a passage of log files potentially containing the answer. Second, we enrich the initial query using the relevance feedback in order to make it relevant to all types of log files. Third, we build an IR system in order to find the relevant passages. For this purpose, we adapted the Lucene search engine [9]. Lucene is a high-performance, full-featured text search engine library written entirely in Java. It offers features like scalable, high-performance indexing, powerful, accurate and Efficient search algorithms, and cross-platform solutions. In addition, Tellex et al. (2003a) notes that passage retrieval algorithms using Lucene actually achieve a higher performance on average.

Nevertheless, we first need to change the *preprocessing* and *document representation* methods in Lucene to adapt them to the log file characteristics. The special preprocessing of log files is developed in Saneifar et al. (2009). Once

[8] A given keyword can correspond to few lexical worlds according to its number of occurrences in the corpus.

[9] http://lucene.apache.org/

Lucene methods are adapted to the domain characteristics, we use its search engine based on a vectorial model to retrieve the relevant passages.

At this point, we also investigate query creation using initial keywords and expansion terms. Therefore, we need to carefully balance the original query (containing only the initial keywords) and feedback information (expansion terms) because if we over-trust the feedback information, then we may be biased in favour of a particular subset of relevant documents, but under-trusting the feedback information would not take advantage of the feedback. That is why we consider a weight for expanding terms as well as the initial terms. The relevant weights are obtained within some experiments. The details are provided in Section 7.

Several passage retrieval approaches return a considerable number of candidate passages. Our experiments conducted on real data assert that in more than 80% of cases the relevant passage is located among the three top-ranked passages. A detailed study on passage retrieval results and the impact of different parameters like the use of terminological knowledge, number of expansion terms, etc., are provided in Section 7.

5 How to Find Terms Correlated to Answers

This phase of query expansion is done just before the passage retrieval and directly using the *test corpus* (in which we seek relevant passages). At this level, we aim at expanding the query directly using information of the test corpus[10]. Using the test corpus to expand the query makes it impossible to have explicit relevance feedback. Thus, we propose to use pseudo relevance feedback in order to determine the expansion terms at this stage.

In pseudo relevance feedback, the expansion terms are extracted from the top-ranked initially retrieved documents since these documents are supposed to be the most relevant. However, we try to define a new way to determine which terms in a corpus should be selected to expand queries instead of extracting them in top-ranked initially retrieved documents which are blindly considered as relevant.

Therefore, we look for terms which are likely to exist in the relevant passage and are, therefore, related to the query. We thus propose here a *novel and original term weighting function*, called TRQ (Term Relatedness to Query), which gives a score to each term of the test corpus based on its relatedness to the initial query. Therefore, the expansion terms are only selected based on their TRQ scores described in the following paragraphs.

[10] In the previous phase, the query is enriched using the information of a training corpus.

5.1 Term Relatedness to Query (TRQ) Measure

Firstly, we use the hypothesis presented below to reduce the search spaces, wherein we look for expansion terms in some relevant fragments of the test corpus.

- Hypothesis: The most *correlated terms* to query should exist in a *lexical world* representing the *context* of the question

Based on this hypothesis, for each query keyword, we extract their lexical worlds in the *test corpus*. We note that the system has no information on the logs of the test corpus and relevant passages[11]. We finally obtain a set of lexical worlds, corresponding to the query keywords, which are extracted from the test corpus. These lexical worlds are used as search spaces wherein we seek the terms related to the query.
Our scoring function, TRQ, is then calculated based on the two assumptions:

- The final query must contain *discriminant terms* (i.e., terms which do not exist in several passages) to be efficient and relevant.
- *Most of the relevant query keywords* should be associated with the corresponding *relevant passage* (i.e., the relevant passage contains most of the query keywords)

Firstly, we seek to select discriminative terms. In other words, we look for terms with a very low frequency of occurrence in different lexical worlds. For this, we use the IDF function by considering each lexical world extracted in the previous step as a document and all lexical worlds as a corpus. In this way, we favour terms which exist in one or a very small number of lexical worlds.

Secondly, in order to favour terms which likely exist in the relevant passage, we give another score (*besides idf*) to each term based on the second assumption. For a given term t, this score that we call lwf (Lexical World Frequency) depends on the number of query keywords which are associated with the lexical world wherein the term t exists. This score presents a form of *df* (Document Frequency) where *a document* corresponds to *all lexical worlds associated* with *a query keyword*. Indeed, by this score, we measure *the importance* of the *lexical world* in which the *given term* is located. This importance is calculated according to the number of query keywords which are associated with the lexical world.

The lwf formula for the term t existing in the lexical world i is calculated as follows:

$$\mathbf{lwf_{ti}} = \frac{\mathbf{1}}{\mathbf{1} + \log(\mathbf{K}/\mathbf{K_i})}$$

K is the total number of query keywords and K_i shows the number of query keywords which are associated with the lexical world i. According to this formula,

[11] The learning process is performed on logs of the training corpus which are generated by a tool using a vocabulary and structures that differ from the tool generating the logs of the test corpus.

the maximum value of lwf is equal to 1 (i.e. when all the query keywords are associated with the lexical world i) and by decreasing the number of query keywords associated with the lexical world i, the lwf value decreases too. Thus, the lwf value always remains between 0 and 1. We also note that K_i is never equal to zero as each lexical world must be associated with at least one query keyword.

The final score of a term, i.e. TRQ (Term Relatedness to Query), is calculated using the following formula:

$$\mathbf{TRQ(ti)} = \alpha * \mathbf{lwf_{ti}} + (\mathbf{1} - \alpha) * \mathbf{idf_t} \quad \alpha \in [0, 1]$$

According to the experiments, the most relevant value of α is 0.25. This means that we give more weight to IDF and a smaller weight, but which influences the final results to lwf.

We explain, with an example, the process of selection of terms which are likely related to the answer. Supposing Q=$\{W_a, W_b, W_d\}$ is a query enriched by the first module (learning phase) and $\log_b$ is a log file containing seven segments:

$\mathbf{S_1}$=$\{W_a\ W_k\ W_m\ W_b\}$ $\mathbf{S_4}$=$\{W_a\ W_c\ W_e\ W_q\}$ $\mathbf{S_7}$=$\{W_b\ W_c\ W_k\}$
$\mathbf{S_2}$=$\{W_d\ W_k\}$ $\mathbf{S_5}$=$\{W_b\ W_e\}$
$\mathbf{S_3}$=$\{W_z\}$ $\mathbf{S_6}$=$\{W_z\}$

In this example, we suppose that the border of lexical worlds (*border of the selected fragment of text around a given term*) corresponds to the border of segments (i.e., a lexical world is not bigger than the corresponding segment). Among seven segments, five are associated with terms of Q. Thus, we obtain S_1, S_2, S_4, S_5, S_7 as the set of lexical worlds of question keywords. The following lists show the lexical worlds associated with each keyword of the question[12].

W_a:$\{S_1, S_4\}$ W_b:$\{S_1, S_5, S_7\}$ W_d:$\{S_2\}$

Here, for instance, the word W_a in the query Q is associated with two lexical worlds (S_1 and S_4). The *idf* score of the word W_k, for example, is equal to $log(\frac{5}{3}) = 0.22$ because the word W_k exists in three lexical worlds (S_1, S_2 and S_7) among five. The value of lwf for the word W_k in the segment S_1 is calculated as follows: $lwf_{k,1} = \frac{1}{1+\log(\frac{3}{2})} = 0.85$. Indeed, two words in the query Q (i.e. W_a and W_b) are associated with the lexical world S_1 (the lexical world in which the word W_k is located). We note that, for a given term, the value of lwf depends on the lexical world in which the given term is located. For example, the value of lwf for the word W_k located in segment S_2 is equal to $lwf_{2,2} = \frac{1}{1+\log(\frac{3}{1})} = 0.68$ as there is just one keyword of the query Q associated with S_2. This means that W_k located in segment S_2 is *less significant* (less related to the query) than when it is located in segment S_1.

[12]Since a word can be used in different contexts (i.e., different fragments of document), it can be associated with several lexical worlds.

5.2 Considering Terms Dependency in the TRQ measure

Another factor to consider in the selection of terms related to a query, is how they are correlated to the query keywords. For this purpose, we assess the *tendency* of terms to appear *close* to keywords of the initial query. In other words, we seek to calculate the *dependence* of *terms* to the *keywords* of the question in the studied context. To calculate the dependence of terms, several measures can be applied: Mutual Information Guiasu (1977), Cube Mutual Information Daille (1996), and Dice coefficient Rijsbergen (1979). We choose the "Dice" coefficient for this purpose as this statistical measure has good performance for text mining tasks Roche and Kodratoff (2009). Also, as noted in Tellex et al. (2003a), in general, a scoring function based on how close keywords appear to each other is common among passage retrieval algorithms. Dice value makes it enable to calculate how two terms are dependent based on their occurrence distance in a corpus.

For two terms X and Y, the Dice coefficient is defined as twice the number of times X and Y appear together over the sum of the total number of times that each one appears in the corpus.

$$Dice(X,Y) = \frac{2 * |(X,Y)|}{|X| + |Y|}$$

Since we would like to measure the *dependence of a given term to a query rather than to a keyword*, we calculate the *sum of Dice values for each pair of the term and a keyword of the query*. Thus, we obtain an extended version of the Dice coefficient which measures the dependence of a term T to a query Q.

$$Dice_{ext}(T,Q) = \sum_{i=1}^{|K_Q|} \frac{2 * |(T,k_i)|}{|T| + |k_i|}$$

The $|K_Q|$ presents the number of keywords in the query Q and k_i is a keyword of Q.

We discuss here the notion of appearing together in the Dice coefficient. For us, $|\ (T,k_i)\ |$, i.e., number of times where T and k_i occur together, corresponds to the number of times where T and k_i are located in the same line in the corpus of logs. Although we agree that the distance between T and k_i can be taken into account if necessary, but in log files we look only for the close occurrence of the two terms without limiting it to a predefined distance in the same line, which does not seem relevant. Moreover, we note that we could extend the "same line" constraint to a window of lines. This means that T and k_i occur together if there are just a predefined number of lines between them.

We finally extend the previously presented TRQ measure by integrating the extended Dice value ($Dice_{ext}$) into it. Thereby, we obtain the final extended TRQ measure which is formulated as following :

$$\mathbf{TRQ_{ext}(ti,Q)} = \alpha * (\mathbf{lwf_{ti}}) + (\mathbf{1} - \alpha) * \mathbf{idf_t} + Dice_{ext}(t,Q)$$

Thus, the $\mathbf{TRQ_{ext}}$ measure also takes into account the dependency of terms on a given query in the studied context. Note that, by calculating only the simple version of TRQ, we have many terms with equal TRQ values. This situation can happen for terms situated in passages wherein there is the same number of keywords. In such situation, taking the extended Dice value $Dice_{ext}$ makes it enable to distinguish terms who are more likely to exist in answers. We note that we could not use only Dice measure since there are terms dependent on queries, which do not exist in relevant passages. In order to best identify the expanding terms, which are likely existing in relevant passages, we have to consider all parameters of TRQ_{ext}. The assumptions based on which we have defined TRQ measure ensure that expansion terms are selected in passages which are likely the relevant ones and the terms likely exist in answers.

5.3 Term Selection using the TRQ measure

Once the TRQ_{ext} score of all terms of lexical worlds is calculated, we identify the n *highest scores* and select the terms having these scores. Note that, for improving the *performance* of our query expansion approach, we first calculate the simple TRQ of terms (i.e., without calculating extended Dice values). Then, we process to calculate the extended Dice values only for n terms having the highest simple TRQ score.

Once we ranked the terms based on their TRQ_{ext} scores, we have to decide about how to choose the expansion terms. According to our experiments, the selection of terms depends on the occurrence frequency of the answer in the corpus. In other words, it depends on if there are more than one relevant passage in the corpus for a given question. In the context of log files, as there are rarely more than one relevant passage for a question, we first give a score to each passage based on the TRQ_{ext} score of terms that it contains. This score is calculated as the sum of TRQ_{ext} values of the passage terms. Then, although there are only one relevant passage, we select the m top-ranked passages to make the system tolerable to the potential unexpectedness. The expansion terms are finally selected in the m top-ranked passages based on the highest TRQ_{ext} values. The system automatically integrates the m top-ranked terms into the query in an autonomous mode. The enriched query will be used in passage retrieval.

In a context where there are more than one relevant passage, we need to adapt some part of TRQ score (i.e., calculation of idf) as well as how we select the terms. This issue is developed in detail in Section 6 where we discuss how to adapt our approach to the context of open domains.

In the selection of expansion terms, there are some parameters which can impact the relevance of selected terms. The value of these parameters like the use of terminological knowledge, the number of expansion terms, or the weight of expansion terms in the final queries are determined within an exhaustive

experimental protocol whose results are presented in Section 7.

6 Application of the Query Expansion Approach in Open Domains

Despite the fact that our approach is designed by default for the restricted domain presented before (i.e., IC design log files), here we study its application to open domains. That reveals how we can adapt our query expansion methods in order to also work well on general fields.

We present here a brief introduction of the general documents used in this study. The test results are presented and discussed in section 7. We use a corpus of general documents which are used in TREC'04 Novelty Track. This data is designed to investigate the system abilities to locate relevant information within a set of documents concerning a TREC topic. Given a TREC topic and an ordered list of documents, systems must find relevant and novel sentences that should be returned to the user from this set. This task integrates aspects of passage retrieval and information filtering National and Soboroff (2004). Some topics are about air flight disasters, ship cloning, political opinions, sport events, etc.

This corpus differs in some principal aspects from the corpus of log files, which requires some adaptation of the query expansion approach. Firstly, the general documents are written in a natural language and they comply with its grammars. Thus, there are more relevant semantic and syntactic relations between the terms of the corpus. Moreover, there are no meaningful special characters or terms containing such characters. Secondly, the document structures and vocabulary are not different in the open domain corpus. This means that it is less challenging to reformulate the relevant expanded queries. In open domains, contrary to the restricted domains, it is not expected that there will be any new documents which would likely have the structures and vocabulary that differ significantly from those of training corpus. Thirdly, in the open domain corpus, we have more than one relevant passage for a given query. Thus, in the explicit relevance feedback phase, we can have more than one relevant passage wherein we have to look for the expansion terms. That means that we need to change how we select the expansion terms in the training phase.

According to these differences, we adapt some parts of our approach. To consider the semantic and syntactic relations in the general documents, we study the impact of using terminological knowledge in query expansion in open domains. For this purpose, we compare the test results obtained by using terminological knowledge in the restricted domain with results obtained in the open domain application.

Based on the second difference, the impact of the second query expansion

phase is evaluated in the experiments. Since vocabulary changes in the open domain test corpus are not expected to be significant, the impact of the second query expansion phase should not be as same as its impact in the restricted domain. This issue is also studied in the experiments in Section 7.

Finally, the last difference requires some changes in expansion term selection in the *first phase of enrichment* (i.e., context learning). As we can have more than one relevant passage in the training phase, we change the term selection method at this stage as follows. By default, the expansion terms are extracted in one relevant passage (identified within the learning process) as in the corpus of log files there is only one relevant passage for a given question. In the case of open domains, the user can choice more than one passage as the relevant passages in the phase of training. Thus, we first select the most representative terms of each selected relevant passage. Then, to select the most relevant expansion terms, we seek terms which are seen in most of the relevant passages. Thus, terms having a higher frequency of occurrence in the relevant passages are selected for expansion.

Calculation of the IDF score in the TRQ measure is also slightly changed to take into account the fact that there is surely more than one relevant passage in the open domain corpus. In fact, in the case of log files, as there is only one relevant document, we calculate the IDF score by considering each extracted lexical world as a document and their collection as the corpus. However, in the case of open domains, there are usually few lexical worlds which are likely all relevant. As by the IDF score we favour terms having less occurrence in different documents, we can eliminate representative terms of *relevant* lexical worlds because they likely occur in *all relevant* lexical worlds. That is why we merge all extracted lexical worlds to create a single document. We calculate the IDF, while this created document as well as other segments of the test corpus constitute a new corpus.

As by merging the lexical worlds into a single document, we increase the occurrence frequency of representative terms in the document, we highlight the IDF score by the TF score of terms in the new document built by merging the lexical worlds. The impact of these changes and the required configurations in the open domain case are evaluated in the experiments presented in the next section.

7 Experiments

Here we present the results of tests performed to evaluate the performance of our approach. We also determine the best values for every parameter. The tests are performed in two principal categories. Firstly, we present our experiments in the restricted domain of log files. Then the same tests are carried out to evaluate application of the approach in an open domain (TREC data). As explained in Section 6, the query expansion approach is adapted to some characteristics of open domain texts which do not exist in the restricted domain of log files.

We study different values for only three parameters of our approach: (1) *Usage of terminological knowledge*, (2) *Weight of expansion terms*, and (3) *Number of expansion terms*. Our test protocol allows us to determine the best values for every parameter in each category. Moreover, we compare the impact of each parameter in a category to its impact in the other one.

7.1 Experiments on Log Files

We test the performance of our approach on a corpus of log files from the real industrial world (*data from Satin Technologies*). The questions, expressed in natural language, are all extracted from a *standard check-list* designed by societies like IEEE. The choice of questions rises some difficulties as we require them to be extracted from real industrial quality verification check-list. In the same time, we have to have access to real log files which contain the answers of selected questions. We finally obtained 36 questions which are used in real industrial applications.

The test corpus that we use for the experiments contains 625 segments and is about 950 KB. Each segment consists of approximately 150 words. We present below two examples of segments found in the log files used for experiments.

```
Verification undefined
----------------------
Reference design:  r:/WORK/fifo
Implementation design:  i:/WORK/fifo
582 Passing compare points
--------------------------------------------------------------------------------
Matched Compare Points   BBPin   Loop   BBNet   Cut   Port   DFF    LAT
TOTAL
--------------------------------------------------------------------------------
Passing (equivalent)     0     0     0     0     36     34     512
582
Failing (not equivalent)   0    0     0     0     0     0     0
undefined


Number of Macro Cells:    2
Number of Module Cells:    1254
Number of Pins:    8234
Number of IO Pad Cells:    40
Number of IO Pins:    38
Number of Nets:    1438
Average Pins Per Net (Signal):    3.30829
```

The training corpus used in the context learning phase consists of logs reporting the same information as logs of the test corpus, but generated by a totally different tool – *thus different vocabulary and segmentation*. For a given question, the initial query is obtained on the basis of the question keywords.

In order to evaluate the performance of our approach in different conditions, we use the *Mean Reciprocal answer Rank* (MRR) used in TREC as a performance evaluation measure Voorhees (1999). This measure takes into account

the rank of the correct answer among the ranked list of candidate answers.

$$MRR = \frac{1}{nb(Question)} \sum_{i=1}^{nb(Question)} \frac{1}{rank(answer)}$$

We also calculate the *Recall* as the *number of questions* for which the *relevant passage* is found among the *top five retrieved passages*.

Moreover, we demonstrate by $P(n)$ the *percentage of questions* for which the *relevant passage* is *ranked as* n among the retrieved candidate passages as possibilities. In the following sections, we show the results for the first three ranks.

Firstly, we present the results of different tests using the different configurations. Once we obtain the best configuration, we compare the passage retrieval performance using enriched queries with the performance of passage retrieval using the initial queries (not enriched). That shows how our query enrichment approach improves the relevancy of the initial queries.

7.1.1 Usage of Terminological Knowledge

In this test, we use the training and test corpus described previously. First, we do not use the domain terminological knowledge in the query expansion phases. This means that we do not extract the domain terms and thus there are not any multi-word terms among the expansion candidate terms. Second, we conduct the same test, but by using the terminological knowledge of the domain. In both tests, all other parameters remain unchanged.

	Explicit FB	Pseudo FB
P(1)	29	24
P(2)	0	7
P(3)	3	3
MRR	**0.84**	**0.80**
Recall	**0.88**	**1**

(a)

	Explicit FB	Pseudo FB
P(1)	29	26
P(2)	1	7
P(3)	3	3
MRR	**0.84**	**0.85**
Recall	**0.91**	**1**

(b)

Table 1: Performance of passage retrieval in log files **(a)** not using the domain terminological knowledge, **(b)** using the terminological knowledge

As the results show (see Table 1 [b]), we have a performance gain by considering the terminological knowledge of the domain. Multi-word terms are more significant and discriminative than simple terms. Hence extracting multi-word terms to characterize the context of questions (lexical worlds) helps find more significant expansion terms. Although the MMR value obtained by the queries expanded in the first phase is close to the MMR value obtained using the queries also expanded in the second phase, we observe that the second query expansion phase using the TRQ measure helps improve the Recall of passage retrieval.

Hence, by using the second query enrichment phase, all relevant passages are situated among the five top-ranked retrieved passages. According to this test, we use the terminological knowledge in all following tests.

7.1.2 Weight of Expansion Terms

Here we present some tests to evaluate the best value for the weight of expansion terms in the enriched queries. In fact, we give the value 1 as weight to the initial keywords of queries and a fraction of one to expansion terms. We choose three values: 0.2, 0.5, 0.7. Each test is performed using one of the values as the weight of expansion terms while the other parameters remain the same. Table 2 presents the test results regarding the different expansion term weight values.

	Explicit FB	Pseudo FB
P(1)	29	27
P(2)	1	7
P(3)	3	2
MRR	**0.85**	**0.87**
Recall	**0.94**	**1**

(a) a

	Explicit FB	Pseudo FB
P(1)	29	26
P(2)	1	7
P(3)	3	3
MRR	**0.84**	**0.85**
Recall	**0.91**	**1**

(b) b

	Explicit FB	Pseudo FB
P(1)	27	24
P(2)	3	6
P(3)	3	6
MRR	**0.82**	**0.81**
Recall	**0.94**	**1**

(c) c

Table 2: **(a)** weight of expansion terms = **0.2**, **(b)** weight of expansion terms = **0.5**, **(c)** weight of expansion terms = **0.7**. Tests performed on log files.

According to the results, we obtain the best performance by using 0.2 as the weight of expansion terms in the enriched queries. By increasing the weight of expansion terms, the MRR value decreases slightly. Therefore we use 0.2 in all of the following terms as the weight of expansion terms.

7.1.3 Number of Expansion Terms

We try to find the best number of expansion terms to include in the initial queries. An irrelevant number of terms can bias the performance. Actually, including all expansion candidate terms is not relevant because this can bias the relevance of queries and significantly decrease the importance of the initial keywords. A few numbers of terms can be irrelevant because we may add insufficient amount of information to the initial queries. Thus, we select three values (3,5,7) as the number of expansion terms to integrate into initial queries. We present the results in Table 3.

	Explicit FB	Pseudo FB
P(1)	29	27
P(2)	1	7
P(3)	3	2
MRR	**0.85**	**0.87**
Recall	**0.94**	**1**

(a) a

	Explicit FB	Pseudo FB
P(1)	28	25
P(2)	2	7
P(3)	3	2
MRR	**0.83**	**0.82**
Recall	**0.94**	**1**

(b) b

	Explicit FB	Pseudo FB
P(1)	28	25
P(2)	2	5
P(3)	1	2
MRR	**0.82**	**0.81**
Recall	**0.91**	**1**

(c) c

Table 3: **(a)** number of expansion terms = **3**, **(b)** number of expansion terms = **5**, **(c)** number of expansion terms = **7**. Tests performed on log files.

As shown in this table, the best result is obtained while we select only the *three* most relevant expansion terms. As we observed in the case of the weight of expansion terms, by increasing the number of expansion terms, the relevance of the expanded queries decreases.

7.1.4 Initial Queries Vs. Expanded Queries

We aim at evaluating the performance gain while we perform passage retrieval using the expanded queries. We thus perform the passage retrieval in log files once using the initial queries (not enriched) and once using the enriched ones. We use the best values obtained in the previous tests for the corresponding parameters of our query expansion approach. Table 4 presents the results for both tests.

	Initial Queries	Expanded Queries
P(1)	24	27
P(2)	2	7
P(3)	2	2
MRR	**0.71**	**0.87**
Recall	**0.80**	**1**

Table 4: Performance of passage retrieval in log files obtained by using the ***non-enriched queries*** vs. performance obtained by using ***enriched queries***

As shown in this table, by using the non-enriched queries, we obtain an MRR value equal to 0.71 in the best conditions. While by enriching the initial queries using our approach, the MRR significantly improves and reaches 0.87 in

the best conditions.
According to the results, in the best configuration and by *using our query enrichment approach*, the relevant passage is ranked in 75% of cases as the first passage among the candidate passages returned by the system. Moreover, in 100% of cases, the relevant passage is located (ranked) among the five top-ranked passages returned by the system when there are about 650 passages in the corpus.
In the next section, we evaluate our approach on documents from general domains.

7.2 Experiments on TREC Data

In this evaluation, we aim at studying the performance of our query expansion approach in general domains. For this purpose, we use the documents used in the TREC'04 Novelty Track [13]. In this corpus, there are several files each one containing several small documents. For a given topic, there are some corresponding documents in every file. That is why we consider the corpus as a collection of documents regardless of their physical location in several files. Since every document contains between 10 and 20 lines, we consider them as passages. In the corpus, we have 1420 documents (passages) and there are 50 search topics. Each topic, like "Find opinions about the partial birth abortion ban", presents a query. For a given topic, there are on average 10 relevant documents (passages).

The test performances are calculated using the 3-fold cross validation. We follow the same test procedure that we used in the case of log files.

7.2.1 Usage of Terminological Knowledge

We focus on studying the impact of using terminological knowledge within our query expansion approach in the open domains. Table 5 presents the passage retrieval results based on the usage and non usage of terminological knowledge.

	Explicit FB	Pseudo FB
P(1)	45	45
P(2)	1	2
P(3)	0	1
MRR	**0.91**	**0.92**
Recall	**0.96**	**1**

(a) a

	Explicit FB	Pseudo FB
P(1)	49	49
P(2)	1	1
P(3)	0	0
MRR	**0.99**	**0.99**
Recall	**1**	**1**

(b) b

Table 5: Performance of passage retrieval in TREC data (open domain), **(a)** not using the domain terminological knowledge, **(b)** using the terminological knowledge is.

[13] http://trec.nist.gov/data/t13_novelty.html

The MMR value is equal to 0.91 in the case of non usage of terminological knowledge, as compared to 0.99 in the case of using this knowledge. As the results show, we have a considerable performance gain when we use the terminological knowledge in our query expansion approach. By comparing these results to those obtained on log files (cf. Table 1), we observe that the usage of terminological knowledge is more significant on open domains. It is due to the fact that there are sufficient significant multi-word terms in open domain texts. However, in log files, multi-word terms are not numerous, and their extraction and validation are more challenging than their extraction in open domains. We note at the same time that the usage of terminological knowledge in the case of log files is nevertheless relevant and helps to improve the query expansion performance.

7.2.2 Weight of Expansion Terms

We also perform some tests to evaluate the best values for the weight of expansion terms in the context of open domains. As in the case of log files, we chose three values: 0.2, 0.5, 0.7. Each test is performed using one of the values as the weight of expansion terms while the other parameters remain the same. We show the results obtained in Table 6.

	Explicit FB	Pseudo FB
P(1)	49	49
P(2)	0	0
P(3)	0	1
MRR	**0.98**	**0.98**
Recall	**0.98**	**1**

(a) a

	Explicit FB	Pseudo FB
P(1)	49	49
P(2)	1	1
P(3)	0	0
MRR	**0.99**	**0.99**
Recall	**1**	**1**

(b) b

	Explicit FB	Pseudo FB
P(1)	49	49
P(2)	0	0
P(3)	0	1
MRR	**0.98**	**0.98**
Recall	**0.98**	**1**

(c) c

Table 6: **(a)** weight of expansion terms = **0.2**, **(b)** weight of expansion terms = **0.5**, **(c)** weight of expansion terms = **0.7**. Tests performed on TREC data.

According to the results, the weight of expansion terms does not considerably influence the relevance of the expanded queries in the TREC data context. We obtain the best results while the weight value is set at 0.2.

7.2.3 Number of Expansion Terms

We now aim to determine the best number of expansion terms to include in the initial queries in the context of open domains. We again select three values

(3,5,7) as the number of expansion terms. Table 7 presents the obtained results based on the number of expansion terms.

	Explicit FB	Pseudo FB
P(1)	49	49
P(2)	1	1
P(3)	0	0
MRR	**0.99**	**0.99**
Recall	**1**	**1**

(a)

	Explicit FB	Pseudo FB
P(1)	48	48
P(2)	0	1
P(3)	0	0
MRR	**0.96**	**0.97**
Recall	**0.96**	**0.98**

(b)

	Explicit FB	Pseudo FB
P(1)	47	48
P(2)	0	0
P(3)	1	0
MRR	**0.94**	**0.96**
Recall	**0.96**	**0.96**

(c)

Table 7: **(a)** number of expansion terms = **3**, **(b)** number of expansion terms = **5**, **(c)** number of expansion terms = **7**. Tests performed on TREC Data.

As shown in Table 7, the performance change based on the different number of expansion terms is not considerable. However, the best result is obtained when we select only the *three* most relevant expansion terms. By increasing the number of expansion terms, the relevance of the expanded queries slightly decreases.

7.2.4 Initial Queries Vs. Expanded Queries

Here we aim to evaluate the performance gain while we perform passage retrieval using expanded queries in the open domain context. As we did in the case of log files, we perform passage retrieval once using the expanded queries and once using the initial ones. Table 8 presents the results of both tests.

	Initial Queries	Expanded Queries
P(1)	37	49
P(2)	2	1
P(3)	1	0
MRR	**0.75**	**0.99**
Recall	**0.80**	**1**

Table 8: Performance of passage retrieval in TREC data obtained by using the ***not enriched queries*** vs. performance obtained by using the ***enriched queries.***

According to the results, we obtain an MRR value equal to 0.75 while we

use the initial queries. When expanding the initial queries using our approach, we obtain a significant improvement in terms of MRR value (equal to 0.99). Moreover, in such configurations, the Recall of system is equal to 1, which means that all relevant passages are ranked among the first five.

By *using our query expansion approach*, the relevant passage is ranked in 98% of cases as the first passage among the candidate passages returned by the system. Moreover, in 100% of cases, the relevant passage is located (ranked) among the five top-ranked passages returned by the system when there are about 490 passages in the test corpus.

8 Conclusions

We have presented a query expansion approach based on an explicit and a new kind of pseudo relevance feedback. The query expansion is performed in order to improve the passage retrieval performance in log files. The heterogeneity of the vocabulary and structures of log files and the fact that the keywords used in the questions expressed in natural language do not necessarily exist in the logs makes passage retrieval difficult. Despite these characteristics, our approach makes it possible to adapt an initial query to all types of corresponding log files regardless of their vocabulary. According to the results, by our query enrichment protocol, which is based on our novel weighting function called TRQ (Term Relatedness to Query), we obtained an MRR value of 0.87, while the MRR value was 0.71 by using the *non-enriched* queries. We also tested our approach on open domain documents. The results show that our approach, after a few adaptations, is also relevant in open domains. We obtained an MRR value of 0.99 in this domain.
We plan to evaluate the impact of other parameters, such as the distance of expansion terms from the query keywords, on the query expansion performance. The distance of terms from the query keywords is usually an informative parameter in general domain documents. Moreover, the use of other kinds of Information Retrieval models like probabilistic model is another prospect of this work. We also consider studying the use of Languge Modeling in passage retrieval from log files.

References

Agichtein, E., Lawrence, S., Gravano, L., 2001. Learning search engine specific query transformations for question answering. In: Proceedings of the 10th International Conference on World Wide Web. WWW'01. ACM, New York, NY, USA, pp. 169–178.

Bernhard, D., August 2010. Query expansion based on pseudo relevance feedback from definition clusters. In: Coling 2010: Posters. Coling 2010 Organizing Committee, Beijing, China, pp. 54–62.

Brill, E., Lin, J., Banko, M., Dumais, S., Ng, A., 2001. Data-intensive question answering. In: Proceedings of the Tenth Text REtrieval Conference (TREC). pp. 393–400.

Buscaldi, D., Rosso, P., Gómez-Soriano, J., Sanchis, E., 2010. Answering questions with an n-gram based passage retrieval engine. Journal of Intelligent Information Systems 34, 113–134.

Carpineto, C., de Mori, R., Romano, G., Bigi, B., January 2001. An information-theoretic approach to automatic query expansion. ACM Transactions on Information Systems 19, 1–27.

Chalendar, G., Dalmas, T., Elkateb-Gara, F., Ferret, O., Grau, B., Hurault-Plantet, M., Illouz, G., Monceaux, L., Robba, I., Vilnat, A., 2002. The question answering system qalc at limsi, experiments in using web and wordnet. In: TREC.

Clarke, C. L. A., Cormack, G. V., Lynam, T. R., 2001. Exploiting redundancy in question answering. In: Proceedings of the 24th annual International ACM SIGIR Conference on Research and development in information retrieval. SIGIR'01. ACM, New York, NY, USA, pp. 358–365.

Clarke, C. L. A., Cormack, G. V., Tudhope, E. A., January 2000. Relevance ranking for one to three term queries. Information Processing and Management 36, 291–311.

Cui, H., Sun, R., Li, K., Kan, M.-Y., Chua, T.-S., 2005. Question answering passage retrieval using dependency relations. In: Proceedings of the 28th annual International ACM SIGIR Conference on Research and development in information retrieval. SIGIR'05. ACM, New York, NY, USA, pp. 400–407.

Daille, B., 1996. Study and Implementation of Combined Techniques for Automatic Extraction of Terminology. In: The Balancing Act: Combining Symbolic and Statistical Approaches to Language. The MIT Press, Cambridge, Massachusetts, pp. 49–66.

Ferrés, D., Rodríguez, H., 2006. Experiments adapting an open-domain question answering system to the geographical domain using scope-based resources. In: Proceedings of the Workshop on Multilingual Question Answering. MLQA'06. Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 69–76.

Gillard, L., Bellot, P., El-Bèze, M., 2006. Influence de mesures de densité pour la recherche de passages et l'extraction de réponses dans un système de questions-réponses. In: CORIA'06. Université de Lyon, pp. 193–204.

Guiasu, S., 1977. Information Theory with Applications. McGraw-Hill, New York.

Ittycheriah, A., Roukos, S., 2002. Ibm's statistical question answering system-trec 11. In: TREC.

Kaszkiel, M., Zobel, J., 1997. Passage retrieval revisited. In: Proceedings of the 20th annual International ACM SIGIR Conference on Research and development in information retrieval. SIGIR'97. ACM, New York, NY, USA, pp. 178–185.

Keikha, M., Gerani, S., Crestani, F., 2011. Temper: a temporal relevance feedback method. In: Proceedings of the 33rd European conference on Advances in information retrieval. ECIR'11. Springer-Verlag, Berlin, Heidelberg, pp. 436–447.

Kosseim, L., Yousefi, J., 2008. Improving the performance of question answering with semantically equivalent answer patterns. Data Knowl. Eng. 66 (1), 53–67.

Lamjiri, A. K., Dubuc, J., Kosseim, L., Bergler, S., 2007. Indexing low frequency information for answering complex questions. In: 8th International Conference on Computer-Assisted Information Retrieval (RIAO'07). Carnegie Mellon University, Pittsburgh, PA, USA.

Lee, G. G., Seo, J., Lee, S., Jung, H., hyun Cho, B., Lee, C., Kwak, B.-K., Cha, J., Kim, D., An, J., Kim, H., Kim, K., 2001. Siteq: Engineering high performance qa system using lexico-semantic pattern matching and shallow nlp. In: Proceedings of the Tenth Text REtrieval Conference (TREC). pp. 442–451.

Lee, K.-S., Croft, W. B., 2013. A deterministic resampling method using overlapping document clusters for pseudo-relevance feedback. Inf. Process. Manage. 49 (4), 792–806.

Li, X., Zhu, Z., 2008. Enhancing relevance models with adaptive passage retrieval. In: Proceedings of the IR research, 30th European Conference on Advances in information retrieval. ECIR'08. Springer-Verlag, Berlin, Heidelberg, pp. 463–471.

Light, M., Mann, G. S., Riloff, E., Breck, E., December 2001a. Analyses for elucidating current question answering technology. Natural Language Engineering 7, 325–342.

Light, M., Mann, G. S., Riloff, E., Breck, E., December 2001b. Analyses for elucidating current question answering technology. Natural Language Engineering 7, 325–342.

Ligozat, A.-L., Grau, B., Tribout, D., 2012. Morphological resources for precise information retrieval. In: 15th International Conference,"Text, Speech and Dialogue". pp. 689–696.

Lin, J., 2007. An exploration of the principles underlying redundancy-based factoid question answering. ACM Transactions on Information Systems 25 (2), 6.

Llopis, F., Vicedo, J. L., Ferrández, A., 2002. Passage selection to improve question answering. In: proceedings of the 2002 Conference on multilingual summarization and question answering - Volume 19. MultiSumQA'02. Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 1–6.

Lv, Y., Zhai, C., 2009. Adaptive relevance feedback in information retrieval. In: Proceeding of the 18th ACM Conference on Information and knowledge management. CIKM'09. ACM, New York, NY, USA, pp. 255–264.

Manning, C. D., Raghavan, P., Schtze, H., 2008. Introduction to Information Retrieval. Cambridge University Press, New York, NY, USA.

Monz, C., November 2003. From document retrieval to question answering. Ph.D. thesis, Universiteit van Amsterdam, Amsterdam.

National, I. S., Soboroff, I., 2004. Overview of the trec 2004 novelty track.

O'Connor, J., 1975. Retrieval of answer-sentences and answer-figures from papers by text searching. Information Processing & Management 11 (5-7), 155 – 164.

Ofoghi, B., Yearwood, J., Ghosh, R., 2006. A semantic approach to boost passage retrieval effectiveness for question answering. In: ACSC'06: Proceedings of the 29th Australasian Computer Science Conference. Australian Computer Society, Inc., Darlinghurst, Australia, Australia, pp. 95–101.

Pasca, M. A., Harabagiu, S. M., 2001. High performance question/answering. In: Proceedings of the 24th annual International ACM SIGIR Conference on Research and development in information retrieval. SIGIR'01. ACM, New York, NY, USA, pp. 366–374.

Rijsbergen, C. J. V., 1979. Information Retrieval. Butterworth.

Rocchio, J., 1971. Book of Relevance Feedback in Information Retrieval. pp. 313–323.

Roche, M., Kodratoff, Y., 2009. Text and Web Mining Approaches in Order to Build Specialized Ontologies. Journal of Digital Information 10 (4), 6.

Salton, G., Buckley, C., 1987. Term weighting approaches in automatic text retrieval. Tech. rep., Ithaca, NY, USA.

Salton, G., Buckley, C., 1997. Improving retrieval performance by relevance feedback. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 355–364.

Salton, G., McGill, M. J., 1986. Introduction to Modern Information Retrieval. McGraw-Hill, Inc., New York, NY, USA.

Saneifar, H., Bonniol, S., Laurent, A., Poncelet, P., Roche, M., 2009. Terminology extraction from log files. In: Proceedings of 20th International Conference on Database and Expert Systems Applications. DEXA'09. Springer, pp. 769–776.

Saneifar, H., Bonniol, S., Laurent, A., Poncelet, P., Roche, M., 2011. How to rank terminology extracted by exterlog. In: Knowledge Discovery, Knowlege Engineering and Knowledge Management. Vol. 128 of Communications in Computer and Information Science. pp. 121–132.

Selberg, E. W., 1997. Information retrieval advances using relevance feedback. URL `http://selberg.org/ speed/papers/generals/generals.pdf`

Tellex, S., Katz, B., Lin, J., Fernandes, A., Marton, G., 2003a. Quantitative evaluation of passage retrieval algorithms for question answering. In: Proceedings of the 26th annual International ACM SIGIR Conference on Research and development in informaion retrieval. SIGIR'03. ACM, New York, NY, USA, pp. 41–47.

Tellex, S., Katz, B., Lin, J., Fernandes, A., Marton, G., 2003b. Quantitative evaluation of passage retrieval algorithms for question answering. In: Proceedings of the 26th annual International ACM SIGIR Conference on Research and development in informaion retrieval. SIGIR'03. ACM, New York, NY, USA, pp. 41–47.

Tiedemann, J., 2007. Comparing document segmentation strategies for passage retrieval in question answering. In: Proceedings of the Conference on Recent Advances in Natural Language Processing (RANLP'07). Borovets, Bulgaria.

Tiedemann, J., Mur, J., 2008. Simple is best: experiments with different document segmentation strategies for passage retrieval. In: Coling 2008: Proceedings of the 2nd workshop on Information Retrieval for Question Answering. IRQA'08. Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 17–25.

Van der Plas, L., Tiedemann, J., 2008. Using lexico-semantic information for query expansion in passage retrieval for question answering. In: Coling 2008: Proceedings of the 2nd workshop on Information Retrieval for Question Answering. IRQA'08. Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 50–57.

Voorhees, E. M., 1999. The trec-8 question answering track report. In: Proceedings of TREC-8. pp. 77–82.

Wade, C., Allan, J., 2005. Passage retrieval and evaluation. Tech. rep.

Wu, J., Ilyas, I., Weddell, G., 2011. A study of ontology-based query expansion. Tech. rep., University of Waterloo.

Xu, J., Croft, W. B., 1996. 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.

Xu, J., Croft, W. B., 2000. Improving the effectiveness of information retrieval with local context analysis. ACM Transactions on Information Systems 18, 79–112.

Yang, H., Chua, T.-S., 2002. The integration of lexical knowledge and external resources for question answering. In: Proceedings of the Eleventh Text REtrieval Conference (TREC'2002). Maryland, USA, pp. 155–161.