View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Institutional Knowledge at Singapore Management University

Singapore Management University
Institutional Knowledge at Singapore Management University

Research Collection School Of Information Systems | School of Information Systems

5-2016

Domain-specific cross-language relevant question retrieval

Bowen XU

Zhenchang XING

Xin XIA

David LO
Singapore Management University, davidlo@smu.edu.sg

Qingye WANG

See next page for additional authors

DOI: https://doi.org/10.1145/2901739.2901746

Follow this and additional works at: https://ink.library.smu.edu.sg/sis_research

Part of the Programming Languages and Compilers Commons

Citation

XU, Bowen; XING, Zhenchang; XIA, Xin; David LO; WANG, Qingye; and LI, Shanping. Domain-specific cross-language relevant question retrieval. (2016). *Proceedings of the 2016 13th International Conference on Mining Software Repositories, Austin, United States, 2016 May 14-15*. 413-424. Research Collection School Of Information Systems.
Available at: https://ink.library.smu.edu.sg/sis_research/3562

This Conference Proceeding Article is brought to you for free and open access by the School of Information Systems at Institutional Knowledge at Singapore Management University. It has been accepted for inclusion in Research Collection School Of Information Systems by an authorized administrator of Institutional Knowledge at Singapore Management University. For more information, please email libIR@smu.edu.sg.

Author
Bowen XU, Zhenchang XING, Xin XIA, David LO, Qingye WANG, and Shanping LI

This conference proceeding article is available at Institutional Knowledge at Singapore Management University: https://ink.library.smu.edu.sg/sis_research/3562

2016 IEEE/ACM 13th Working Conference on Mining Software Repositories

Domain-Specific Cross-Language Relevant Question Retrieval

Bowen Xu[1], Zhenchang Xing[2], Xin Xia[1]*, David Lo[3], Qingye Wang[1], Shanping Li[1]
[1]College of Computer Science and Technology, Zhejiang University, China
[2]School of Computer Engineering, Nanyang Technological University, Singapore
[3]School of Information Systems, Singapore Management University, Singapore
max_xbw@zju.edu.cn, zcxing@ntu.edu.sg, xxia@zju.edu.cn
davidlo@smu.edu.sg, wqyy@zju.edu.cn, shan@zju.edu.cn

ABSTRACT

In software development process, developers often seek solutions to the technical problems they encounter by searching relevant questions on Q&A sites. When developers fail to find solutions on Q&A sites in their native language (e.g., Chinese), they could translate their query and search on the Q&A sites in another language (e.g., English). However, developers who are non-native English speakers often are not comfortable to ask or search questions in English, as they do not know the proper translation of the Chinese technical words into the English technical words. Furthermore, the process of manually formulating cross-language queries and determining the weight of query words is a tedious and time-consuming process.

For the purpose of helping Chinese developers take advantage of the rich knowledge base of the English version of Stack Overflow and simplify the retrieval process, we propose an automated cross-language relevant question retrieval (*CLRQR*) system to retrieve relevant English questions on Stack Overflow for a given Chinese question. Our *CLRQR* system first extracts essential information (both Chinese and English) from the title and description of the input Chinese question, then performs domain-specific translation of the essential Chinese information into English, and formulates a query with highest-scored English words for retrieving relevant questions in a repository of 684,599 Java questions in English from Stack Overflow. To evaluate the performance of our proposed approach, we also propose four online retrieval approaches as baselines. We randomly select 80 Java questions in SegmentFault and V2EX (two Chinese Q&A websites for computer programming) as the query Chinese questions. Each approach returns top-10 most relevant questions for a given Chinese question. We invite 5 users to evaluate the relevance of the retrieved English questions. The experiment results show that *CLRQR* system outperforms the four baseline approaches, and the statistical tests show the improvements are significant.

*Corresponding author.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

MSR'16, May 14-15, 2016, Austin, TX, USA

© 2016 ACM. ISBN 978-1-4503-4186-8/16/05…$15.00

DOI: http://dx.doi.org/10.1145/2901739.2901746

Keywords

Domain-Specific Translation, Cross-Language Question Retrieval

1. INTRODUCTION

Domain-specific community Q&A websites, such as Stack Overflow, have become a prevalent platform for knowledge sharing and acquisition. In the past 7 years, Stack Overflow has accumulated over 10 million questions (as of late August 2015), and has become a tremendous knowledge repository of developers' thoughts and practices. According to Stack Overflow 2015 Developer Survey [2], about 32 million of people visit Stack Overflow monthly, and more than 25 million are return visitors. Return visitors land on Stack Overflow an average of 6 times every month [2]. To allow developers in non-English speaking countries to participant on Stack Overflow, Stack Overflow has launched localized versions of Stack Overflow in Portuguese, Russian, and Japanese.

Ten percent of the world's programmers are in China[1]. Even without a localized version of Stack Overflow in Chinese, it would still be very desirable to support developers in China to easily access the knowledge repository of the English version of Stack Overflow. Developers in China usually graduate with a Bachelor degree. To fulfill the degree requirements, they need to pass the national college English test (Level 4). As such, developers in China are often equipped with basic English reading comprehension skills, and they could be fluent enough to read posts in English. However, most of them often are not comfortable asking questions in English. Furthermore, they often do not know how to translate Chinese technical words into proper English technical words even with the help of translation tools. This makes it difficult for them to formulate English queries to search the Internet.

This reality of English reading and writing skills of developers in China indicates a potential to make the content of the English version of Stack Overflow more easily accessible to developers in China. In this paper, we propose a domain-specific cross-language relevant question retrieval (*CLRQR*) system that takes as input a question written in Chinese and returns relevant questions written in English from Stack Overflow. These relevant questions are the keys to accessing the knowledge in English version of Stack Overflow.

Using our approach, developers can write questions in Chinese (may be mixed with English words such as programming languages, tools, parameters). Given a question, our *CLRQR* system first extracts essential information (both Chinese and English) from the

[1]http://www.drdobbs.com/tools/planet-earth-has-185-million-developers/240165016

Figure 1: A Chinese Question on *SegmentFault*

title and description of the input question, then performs domain-specific translation of the essential Chinese information into English, and finally formulates a query with scored English words for retrieving relevant questions from Stack Overflow. Our *CLRQR* system improves the efficiency of Chinese developers to find solutions to solve their technical problems by solving the problem of domain-specific technical words translation and the problem of query formulation. A key benefit of our approach is that it allows Chinese developers to more easily take advantage of high-quality English Q&A resources on Stack Overflow.

The main contributions of this paper are the following:

1. We introduce a new approach to retrieve relevant English questions for a Chinese question.
2. Based on term frequency of 30,000 Java questions from Stack Overflow, we build a domain-specific vocabulary to optimize the translation results of general Chinese-English translation tool. There are 111,174 English words in the domain-specific vocabulary.
3. We combine two keyword extraction algorithms to improve the accuracy of Chinese keyword extraction.
4. Based on the characteristics of questions on Q&A sites, we design a scored-word based question retrieval algorithm.
5. We conduct a preliminary evaluation of our approach and other four baseline approaches on 80 Java questions in Chinese from SegmentFault and V2EX. The experiment results show that *CLRQR* system outperforms the other four baseline approaches, and the statistical tests show the improvements are significant.

The remainder of this paper is structured as follows. Section 2 presents a motivating example and elaborates the challenges for cross-language relevant question retrieval. Section 3 describes the overall framework and the details of our proposed approach. Section 4 introduces our experimental methods. Section 5 presents our experiments result. Section 6 discusses the qualitative analysis of some search results, and threats to validity of our proposed approach. Section 7 reviews related work. Section 8 concludes our work and discusses our future plan.

2. MOTIVATING EXAMPLE AND DESIGN CHALLENGES

In this section, we present a motivating example to illustrate how our approach formulates English query from an input Chinese Java question and how it retrieves relevant English Java questions from Stack Overflow. Using this example, we highlight the challenges in cross-language question retrieval and summarize our solutions.

Figure 2: A Relevant English Question on *Stack Overflow*

Table 1: Word List with Score

Word	Score
code	1.20
review	1.20
tool	1.20
javaweb	1.00
project	0.20
opensource	0.20

2.1 Motivating Example

Figure 1 shows a Chinese Java question (i.e., tagged with *java*) from SegmentFault (a Chinese Q&A website for computer programming). This Chinese question asks for some useful code review tools which can be used for java and javaweb project. Our *CLRQR* system first extracts essential Chinese and English information from the question, e.g. the Chinese words "项目", "开源", "代码", "审查", "工具" and the English words "java", "javaweb". Because all the questions in this work are Java-related questions, we consider the word "java" not an important word to distinguish the core issues of different questions. Therefore, we discard the English word "java" and keep only the English word "javaweb". Then *CLRQR* system performs domain-specific translation of the extracted Chinese words, and formulates a English query (i.e. a set of 6 English words with scores as shown in Table 1) that an English-speaking developer may use for the similar questions.

Our user study(details in section 4.3) shows that users consider this query accurate and useful in helping them retrieve relevant questions for the given Chinese question. Given the English query, our *CLRQR* system uses a scored-word based question retrieval algorithm to calculate the relevance between the English query and a repository of English Java questions from Stack Overflow. The algorithm returns the top-10 most relevant Java questions for the given Chinese question. Figure 2 shows one of the top-10 most relevant Java questions for the given Chinese Java question in Figure 1. This English question also asks for some useful code review tools which can be used for java. Through the user evaluation, we find that the retrieved English questions are relevant with the Chinese question and it is useful in helping developers solve the problem in the Chinese question.

2.2 Design Challenges

Cross-language relevant question retrieval is a very complex process. To achieve the above objective of the cross-language question

retrieval, we must address the following three challenges:

2.2.1 Challenges in keywords extraction

A question may contain a lot of texts. We would like to extract the essential information for query formulation. To that end, we should use keywords extraction algorithms to summarize the essential information in the question. Many keywords extraction algorithms have been proposed in the natural language processing field. Different algorithms are based on different heuristics to evaluate the importance of a word. As they are heuristic-based, some keywords extraction algorithms may perform better than others on some cases, but worse on other cases. One way to address the weaknesses of these keywords extraction algorithms is to combine them together in order to make a comprehensive judgment. In our *CLRQR* system, we use two different keywords extraction algorithms (FudanNLP [23] and IctclasNLP [1]) to extract Chinese keywords in the title and description of the Chinese question, and take the union of the two sets of keywords as the final Chinese keywords.

2.2.2 Challenges in domain-specific translation

Cross-language question retrieval has to translate the words in the source language into some appropriate words in the target language. The accuracy of this translation will directly affect the relevance of the questions retrieved in the target language. Kluck and Gey [15] point out that in many cases there exists a clear difference between the domain-specific meaning and the common meaning of a word. This means that it can be difficult to use general translation of a word for domain-specific information retrieval. For example, for the Chinese word "代码", the general translation tool returns several English translations, such as "code" and "word". In the context of software engineering, the translation "code" is more appropriate than the other translations. Similarly, for the Chinese word "审查", the translation "review" is more appropriate than the translations "investigate" or "examine".

Several studies propose domain-specific translation techniques which are based on domain-specific dictionary [24] [12] [20]. A few number of research studies have been carried out on domain-specific translation, which are based on domain-specific translation lexicon [24] [12] [20]. However, developing a domain-specific dictionary requires a significant effort. In this paper, we propose a new approach to support domain-specific translation. We analyze a corpus of 30,000 Stack Overflow questions to build a domain-specific vocabulary based on the term frequency of each English word. The corpus contains a total of 111,174 English words. Given a Chinese word, our *CLRQR* system first uses a general translation tool (Youdao translation API[2]) to obtain a few translation candidates. Then, based on the term frequency of the translation candidates in the domain-specific vocabulary, it selects those words whose term frequency are greater than the mean frequency as the translation. According to practical experience, we find that the performance of selecting words whose term frequency are greater than the mean frequency is always better than selecting the only word with highest term frequency. For example, for the Chinese word "审查", the term frequency of the translation "review", "investigate" and "examine" is 2589, 1857, 1807, respectively. As a results, "review" is selected as the translation of the Chinese word "审查". Similarly, "code" is selected as the translation of the Chinese word "代码", as "code" is more frequent used than other translation candidates of "代码" in Stack Overflow discussions. This approach has two advantages over domain-specific dictionary: 1) it is based on the general translation tool; and 2) it utilizes the crowdsourced knowledge to build a domain-specific vocabulary with a very small effort.

[2]http://fanyi.youdao.com/openapi

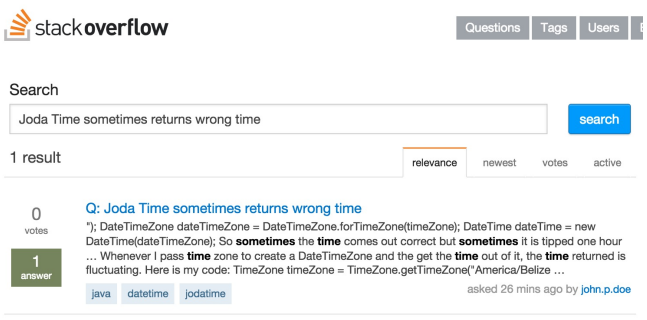

Figure 3: Search *"Joda Time sometimes returns wrong time"* on Stack Overflow

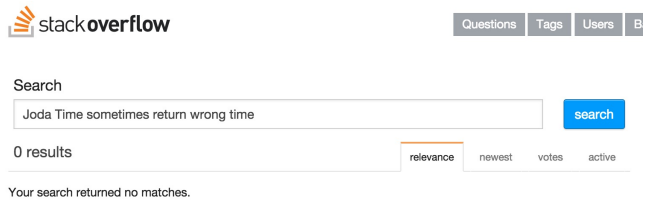

Figure 4: Search *"Joda Time sometimes return wrong time"* on Stack Overflow

2.2.3 Challenges in question retrieval algorithm

When searching on Stack Overflow, we find that the question retrieval algorithm is not very robust. For example, when we search the question *"Joda Time sometimes returns wrong time"*, we can retrieve a question on Stack Overflow successfully as shown in Figure 3. However, if we change the word "returns" to "return", Figure 4 shows that the search for *"Joda Time sometimes return wrong time"* returns no matches. It seems that Stack Overflow question retrieval algorithm does not take word stemming into consideration. Furthermore, we observe that keywords extracted from different parts of the question (such as title versus description) often have different levels of importance for question retrieval. However, existing question retrieval algorithms do not take this into account. In this paper, we design a question retrieval algorithm to address these limitations by considering word stemming and assigning different weights to the words from title and description.

3. THE APPROACH

In this section, we first present the overall framework of our proposed approach to cross-language question retrieval (section 3.1). Then, we describe the details of essential information extraction, domain-specific cross-language translation, and scored-word based question retrieval in section 3.2, section 3.3, section 3.4, respectively.

3.1 Overall Framework

Figure 5 presents the overall framework of our domain-specific Cross-Language Relevant Question Retrieval (*CLRQR*). Given a software-engineering-related question in Chinese, essential information extraction (Step 1) extracts Chinese keyword and other words of concerns (Chinese or English) from the title and description of the question. Given the extracted Chinese keywords, domain-specific cross-language translation (step 3) translates the Chinese

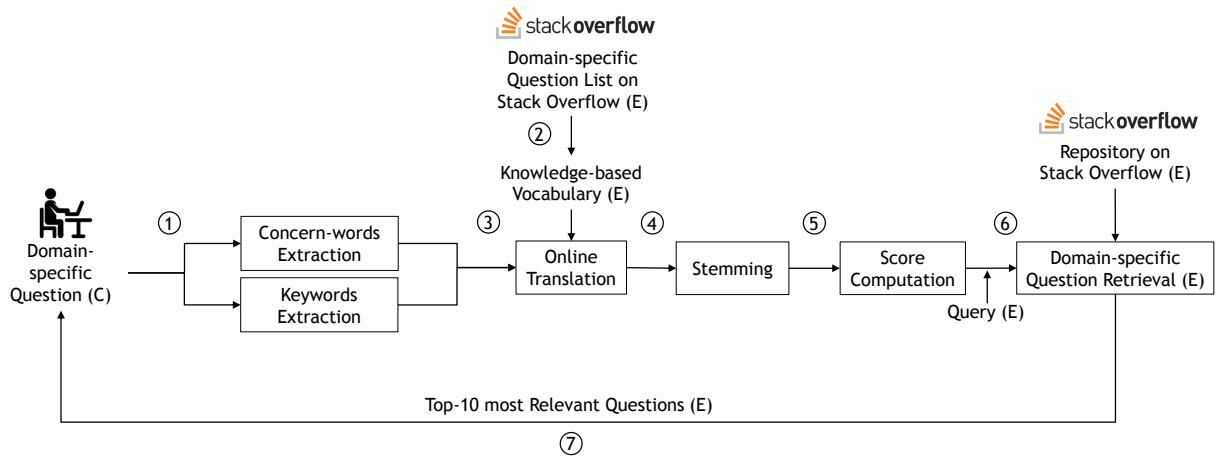

Figure 5: The Framework of Domain-Specific Cross-Language Question Retrieval

words into domain-specific English words, based on a domain-specific vocabulary derived from a corpus of Stack Overflow questions (Step 2). After translating the Chinese keywords, the system formulates an English query as follows: first, it stems the query words (Step 4), then it assigns different weights to the query words based on whether the words are originally Chinese or English and where the words are from (title or description) (Step 5), and finally it takes the top-6 words with the highest scores to formulate the English query (Step 6). The system uses the English query to search a repository of Stack Overflow questions and returns the top-10 most relevant English questions to the user (Step 7).

Algorithm 1 Chinese Keywords Extraction

Input:
$Sentence$: The question description
Output:
CKL: List of Chinese keywords in the $Sentence$
Method:
$KeywordsList1$ = FudanNLP($Sentence$);
for all Keywords $KW^i \in KeywordsList1$ **do**
    CKL.Add(KW^i);
end for
$KeywordsList2$ = IctclasNLP($Sentence$);
for all Keywords $KW^i \in KeywordsList2$ **do**
    CKL.Add(KW^i);
end for
Output CKL;

3.2 Essential Information Extraction

Given a question in Chinese, we extract the essential information based on the following two observations:

1. Question title sums up the core issue of the question better than question description.

2. Most technical words are written in English. Developers always ask some technical questions with some domain-specific words in English. Those English words are important for retrieving relevant questions.

Therefore, we divide the question essential information into two kinds: Chinese keywords and other words-of-concern (referred to as concern words). Table 2 presents the definition of the concern words and the Chinese keywords in question title and question description in our approach.

Table 2: Definition of Concern Words and Keywords

	Concern-words	Keywords
Title	Both Chinese Words and English Words	N/A
Description	Only English Words	Chinese Keywords

Concern words extraction. Each word in question title is considered as words-of-concern no matter it is Chinese or English. This is because based our observations each word in title are important for expressing the core issue of the question. On the other hand, we only take the English words in the question description as words-of-concern.

Chinese keywords extraction. Keywords can be considered as a brief summary of a text. They are a set of phrases semantically covering most of the text. Algorithm 1 presents the implementation of our composite keywords extraction algorithm. The input is the question description, the algorithm uses two different popular Chinese keywords extraction algorithms (FudanNLP & IctclasNLP) to extract Chinese keywords in the description. The algorithm of keywords extraction of FudanNLP [23] is based on TextRank algorithm. The algorithm of keywords extraction of IctclasNLP [1] is based on entropy. The two algorithms produce two different but complementary sets of Chinese keywords. To reduce the bias caused by a single method, the algorithm takes the union of the two keyword sets as the final set of Chinese keywords to summarize the Chinese question description.

For the motivating example shown in Figure 1, the system extracts three Chinese concern-words from the title: “开源”, “审查” and “工具”, and five Chinese keywords from the description: “项目”, “开源”, “代码”, “审查” and “工具”. There are no English concern-words in the title of this question. The system extracts two English concern-words from the question description: “java” and “javaweb”. Because all the questions in this work are Java-related questions, we consider the word “java” not an important word to distinguish the core issues of different questions. Therefore, we discard the English word “java” and keep only the English word “javaweb”.

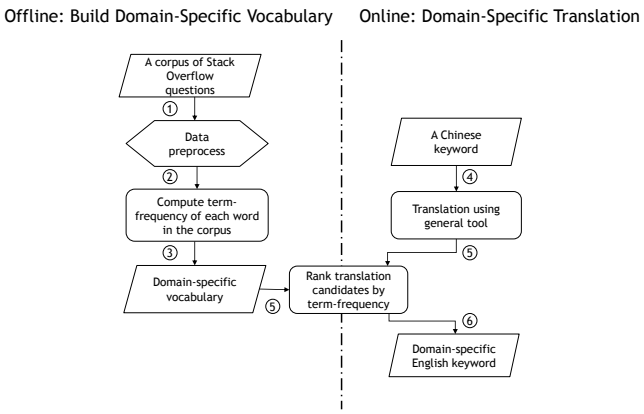

Figure 6: Domain-Specific Cross-Language Translation

3.3 Domain-specific Cross-language Translation

Translation is a critical step in cross-language information retrieval [16] [3]. Recent results show that the challenge lies in how to differentiate a word's domain-specific meaning from its common meaning. A general translation tool may not perform well for domain-specific translation. We proposed a new method to address this limitation. Figure 6 presents the details of our method. We divide the domain-specific cross-language translation into an offline vocabulary building step and an online translation step. To build a domain-specific vocabulary, we make use of crowdsourced knowledge in Stack Overflow discussions. Specifically, in this work we collect a corpus of 30,000 Stack Overflow questions tagged with *java*. The corpus contains a total of 111,174 English words. We first remove stop words (step 1), such as "hello", "the" and "you". The stop-word list we use is available at Snowball[3]. Then, we compute term frequency for each word in the corpus (step 2), and build a vocabulary of each word and their term frequency (step 3).

For online translation, given a Chinese keyword (which can be a Chinese concern-word from the question title or a Chinese keyword from the question description), the system first uses a general Chinese-English translation tool (Youdao translation API is used in this work) to obtain a list of English word candidates (step 4). Then, the system checks the term frequency of these English word candidates in the domain-specific vocabulary (step 5). Finally, the system selects the English words with the term frequency above the mean term frequency of all the candidate English words as the translation of the given Chinese keyword (step 6). If none of the candidate English words exist in the domain-specific vocabulary, the system returns all the general translation candidates as the translation results.

Table 3 shows the general and final translation results of the Chinese concern-words and keywords extracted from the Chinese question in Figure 1. Our method returns the English words "project", "opensource", "code", "review", "tool" as the final translation of the Chinese word "项目", "开源", "代码", "审查" and "工具".

3.4 Scored-word based Question Retrieval Algorithm

Our cross-language question retrieval contains three steps: stemming, word score computation and query formulation, and question retrieval.

Stemming. In the stemming step, we reduce each word to its root form, for example, words "write" and "written" are both reduced to "writ". We use a popular stemming algorithm (the Porter stemmer [21]) in this work.

[3]http://snowball.tartarus.org/algorithms/english/stop.txt

Table 3: Examples of Domain-specific Cross-language Translation

Chinese Word	General Translation Results	Term Frequency in Vocabulary	Mean	Final Result
项目	project	21,021	8,921	project
	item	2,642		
	article	3,101		
开源	opensource	124	124	opensource
代码	code	408,565	209,249	code
	word	9,932		
审查	review	2,589	2,084	review
	investigate	1,857		
	examine	1,807		
工具	tool	7,481	4,998	tool
	instrument	2,515		

Table 4: Different Scores for 4 kinds of Words

Location	Word Type	Weight	WordsetSize	Score (desc)
Title	Concern-Words (code, review, tool)	3	3	code : 1.20 review : 1.20 tool : 1.20 javaweb : 1.00 project : 0.20 opensource : 0.20
Description	Concern-Words (javaweb)	1	1	
	Keywords (tool, code, review, project, opensource)	1	5	

Score computation. Considering different contributions of a word to express the core issue of the question, we assign different weights to each kind of words as defined in Table 2. We set the weights of different kinds of words based on the following observations. We observe that question title usually sums up the core issue of the question better than question description. Therefore, we set the weight of the concern-words in the title higher than the concern-words in the description. However, question description usually contains more technical details than question title. Setting a too high weight for the concern-words in the title will overwhelm the technical words in the question description. Therefore, we set the weight of the concern-words in the title three times higher than the concern-words in the description. The description of a Chinese question usually mixes the Chinese words and English words. In fact, most technical words are usually written in English. These English technical words are as important as the Chinese keywords explaining the question. Therefore, we set the same weight to the concern-words and the Chinese keywords in the question description.

For each English word (either original English words extracted from question title and question description or the translated English words using our domain-specific translation method), we update its *Score* by $(termfrequency \times Weight)/WordsetSize$. If a word belongs to different kinds at the same time, the score of the word will be accumulated. Table 4 also shows the score computation for the words in the question shown in Figure 1. For example, the word "tool" appears in both the title-concern-words and the description-keywords. For the title-concern-words, we calculate the score of the word "tool" as 1, i.e., $(1 \times 3)/3 = 1.00$. For the description-keywords, we calculate the score of the word "tool" as 0.20, i.e., $(1 \times 1)/5 = 0.20$. Thus, we get the final score for the word "tool" as 1.20.

According to practical experience in web log analysis [10, 17],

we use up to six keywords to express the core issue of a question. After computing the score of all the words, we select up-to six words with the highest score to generate an English query. In this example, the English query contains six query words: “code”, “review”, “tool”, “javaweb”, “project” and “opensource”.

Algorithm 2 Domain-specific Cross-language Relevant Question Retrieval Algorithm

```
1: Input:
2: QwMap: Map<Query word, Score>
3: QEQL: Domain-specific English Question List
4: Output:
5: TRQ: Top-10 most Relevant Questions
6: Method:
7: Map<Question, Relevance> QSMap = NULL;
8: SIZE = QwMap.size();
9: for all Question Q^i ∈ QEQL do
10:    Relevance = 0;
11:    ContainNum = 0;
12:    for all Entry<Query word, Score> Qw^i ∈ QwMap do
13:       if Q^i.title.contains(Qw^i) or Q^i.desc.contains(Qw^i) then
14:          ContainNum++;
15:       end if
16:       tf_Title = calcTF(Q^i.title, Qw^i) ;
17:       tf_Desc = calcTF(Q^i.desc, Qw^i);
18:       Relevance += (tf_Title*2 + tf_Desc) * (Qw^i.Score) ;
19:    end for
20:    ContainRatio = ContainNum / SIZE;
21:    Relevance *= (ContainRatio);
22:    QSMap.put(Q^i, Relevance);
23: end for
24: TRQ = getTop-10Questions(QSMap);
25: Output TRQ;
```

Question Retrieval. Considering the characteristics of our work, we design a new approach to retrieve relevant questions in a repository of Stack Overflow questions. Algorithm 2 calculates the relevance between the query words and each question in the repository, and recommends the top-10 most relevant questions to users that may help them solve the problem in the given Chinese question. Given a query word, the algorithm calculates the term frequency of word in the title and description of each English question in the question repository (lines 16-17). Then, it computes the relevance between a query word (*Qw*) and an English question (*Q*) by the equation 1 (line 18). Due to the observation in section 3.2, we consider that the word in *QwMap* from the title of an English question is more important than those from the question description. Thus, we set different weights to the words from the title and the description(i.e., 2 and 1 respectively). To reduce the bias caused by a word with too high term frequency, the algorithm then counts the number of times (i.e., $ContainNum$) that a word appears in the title and description of the English question, and divide the times by the total number of query words (i.e. the size of $QwMap$). Finally, the algorithms multiplies the contain words ratio following equation 2 and the query-question relevance is used as the final relevance score for ranking the relevant questions (lines 21).

$$Relevance(Qw, Q) = (tf\_Q.Title \times 2 + tf\_Q.Desc) \times Qw.score \quad (1)$$

$$Relevance(Query, Q) = \sum_{i=1}^{M} Relevance(Qw, Q) \times ContainRatio \quad (2)$$

Query	S	p	r	...	t	<space>	t	y	p	e
Index	1	2	3	...	137	138	139	140	141	142

Reserve | Remove

Figure 7: Example for Query Formulation of Baseline Approach by the Stack Overflow Search Engine

4. EXPERIMENTAL METHODS

This section describes the experiment design for evaluating our approach.

4.1 Baseline Building

Table 5 presents the four baseline approaches we use to compare the effectiveness of our approach. In the translation phase, *BaselineApproach1* and *BaselineApproach3* translate only the title of the Chinese question, while *BaselineApproach2* and *BaselineApproach4* translate the title and description of the Chinese question. Same as our approach, all the baseline approaches use the Youdao translation API. After the translation, we also remove English stop words from translated words which is same as our approach. We refer the rest of the words as the ‘Query Words’.

BaselineApproach1 and *BaselineApproach2* use Stack Overflow search engine, while *BaselineApproach3* and *BaselineApproach4* use Google search engine. After removing the stop words from the question title and the question description, we formulate the query for the baseline approaches from the first word of the question title and description up to the length limits of the query that a search engine allows. Different search engines have different limits for the length of a query. For the Stack Overflow search engine, the query cannot exceed 140 characters. If the length of the question title and description exceeds 140 characters, we keep the complete words up to the largest character index below or equal to 140. Figure 7 presents an example. As the word “type” is across the limit 140, we keep the words before the word “type” as the query for the Stack Overflow search engine. For the Google search engine, the query cannot exceed 32 words. Thus, we only keep the first 32 words as query if the question title and description contain more than 32 words.

Furthermore, in order to make the comparison more fair, we optimize the search scope for baseline approaches based on their corresponding search engine. When formulating the query, *BaselineApproach1* and *BaselineApproach2* append “[java] is:question” to the query, which instructs the Stack Overflow search engine search only questions tagged with *Java*. *BaselineApproach3* and *BaselineApproach4* append “site:StackOverflow.com” to the query, which instructs Google search engine to search only the Stack Overflow web site.

Both the four baseline approaches and our approach return the top-10 most relevant English questions for each Chinese question.

4.2 Chinese and English Question Databases

We build two databases for Java questions in Chinese and English, respectively. We crawl 200 Java questions from SegmentFault and V2EX as query Chinese question set and randomly choose 80 questions for the experiment. We extract 714,599 Java questions from Stack Exchange Data Dump[4] released by Stack Exchange, Inc. We use 30,000 Java questions to build domain-specific vocab-

[4]https://archive.org/download/stackexchange

Table 5: Four Baseline Approaches for Cross-language Relevant Question Retrieval

	Baseline Approach1	**Baseline Approach2**	**Baseline Approach3**	**Baseline Approach4**
Translation Item	Title Only	Title and Description	Title Only	Title and Description
Query Formulation	"[java] is:question" + "Title Query Words"	"[java] is:question" + "Title Query Words" + "Description Query Words"	"Title Query Words" + "site:StackOverflow.com"	"Title Query Words" + "Description Query Words" + "site:StackOverflow.com"
Search Engine	Stack Overflow Search Engine (not exceeds 140 characters)		Google Search Engine (not exceeds 32 words)	

ulary and another 684,599 as the repository of English questions for question retrieval.

4.3 User Study

We conduct a user study to evaluate the top-10 most relevant questions generated by four baseline approaches and our approach. The evaluator group included 5 master students, all of whom have industrial experience in Java programming (ranging from 3–6 years) and pass the national college English test (Level 4). We provide these five users 80 Chinese Java questions from SegmentFault and V2EX. For each Chinese question, we provide a questionnaire of the top-10 most relevant English questions generated by *BaselineApproach1*, *BaselineApproach2*, *BaselineApproach3*, *BaselineApproach4* and our approach respectively. The user study evaluation has two steps. First, we ask the participants to read the same question at the same time. Second, in order to make reasonable judgments, users can discuss whether the English questions are actually relevant with the corresponding Chinese question or not and make a common decision. In the process of validation, the participants do not know which result is generated by which approach.

4.4 Evaluation Metrics

We use the following metrics to compare the baseline approaches and our approach:

- **Precision@K.** Precision is the percentage of actually relevant questions out of the questions that a retrieval method returns for a Chinese question. It is defined as:

$$Precision@k = \frac{ARQs\ in\ top\text{-}k}{k} \tag{3}$$

  In the above equation, *ARQs* refers to those actually relevant questions. In this paper, we set k = 1, 5 and 10.

- **Recall@K.** Recall is the percentage of actually relevant questions returned for a Chinese question out of all the relevant questions in the repository. It is defined as:

$$Recall@k = \frac{ARQs\ in\ top\text{-}k}{ARQs\ in\ Repository} \tag{4}$$

  It is impractical to check every question in repository to determine its relevance to the input Chinese question. Therefore, *ARQs in Repository* refers to the union of actually relevant questions returned by the four baseline approaches and our approach for a Chinese question. In this paper, we set k = 1, 5 and 10.

- **Top-K Accuracy.** Top-k accuracy is the percentage of Chinese questions for which at least one actually relevant question is ranked within the top-k position in the returned lists of English questions. Given a Chinese question *CQ*, if at least one of the top-k most relevant English questions is actually relevant, we consider the retrieval to be successful, and set the value $Success(CQ, top-k)$ to 1; else we consider the retrieval to be unsuccessful, and set the value $success(CQ, top-k)$ to 0. Given a set of Chinese questions, denoted as *CQs*, its top-k accuracy Top@k is computed as:

$$Top@k = \frac{\sum_{CQ \in CQs} Success(CQ, top\text{-}k)}{|CQs|} \tag{5}$$

  The higher the top-k accuracy score is, the better a relevant question retrieval approach performs. In this paper, we set k = 1, 5 and 10.

- **Mean Reciprocal Rank (MRR).** MRR is a popular metric used to evaluate an information retrieval technique [5]. Given a query (in our case: a Chinese Java question), its reciprocal rank is the multiplicative inverse of the rank of the first correct document (in our case: actually relevant English question) in a rank list produced by a ranking technique (in our case: relevant question retrieval approach). Mean Reciprocal Rank (MRR) is the average of the reciprocal ranks of all Chinese questions in a set of Chinese questions. The MRR of a set of Chinese questions is computed as:

$$MRR(R) = \frac{1}{|CQs|} \sum_{CQ \in CQs} \frac{1}{Rank(CQ)} \tag{6}$$

  In the above equation, *Rank(CQ)* refers to the position of the first actually relevant English question in the ranked list of English questions returned by a cross-language relevant question retrieval approach for a Chinese question.

- **Mean Average Precision (MAP).** MAP is a single-figure measure of quality, and it has been shown to have especially good discrimination and stability to evaluate ranking techniques [5]. Different from top-k accuracy and MRR that only consider the first correct result, MAP considers all correct results. For a query (in our case: a Chinese Java question), its *average precision* is defined as the mean of the precision values obtained for different sets of top k documents (in our case: actually relevant English questions) that were retrieved before each relevant document is retrieved, which is computed as:

$$AvgP(CQ) = \frac{\sum_{j=1}^{M} P(j) \times Rel(j)}{ARQs\ in\ Repository} \quad (7)$$

In the above equation, *M* is the number of English questions in a ranked list, *Rel(j)* indicates whether the English question at position *j* is actually relevant or not (in our case: the question is judged by the users as actually relevant or not), and *P(j)* is the precision at the given cut-off position *j* and is computed as:

$$P(j) = \frac{ARQs\ in\ top\ j\ positions}{j} \quad (8)$$

Then the MAP for a set of Chinese questions *CQs* is the mean of the average precision scores for all Chinese questions in CQs:

$$MAP = \frac{\sum_{CQ \in CQs} AvgP(CQ)}{|CQs|} \quad (9)$$

In relevant question retrieval, a Chinese question may have a number of relevant English questions. We use MAP to measure the average performance of the baseline approaches and our approach to retrieve all of the relevant questions. The higher the MAP value, the better the relevant question retrieval approach performs.

5. EXPERIMENTAL RESULTS

In our experiment, we are interested in the following two research questions:

RQ1: ***How effective is our CLRQR system in cross-language relevant question retrieval? How much improvement can it achieve over the four baseline approaches?***

Motivation. The more accurate *CLRQR* is, the more benefit *CLRQR* would give to its users. Thus, in this research question, we evaluate the effectiveness of *CLRQR* and compare it with the four baseline approaches.

Approach. We evaluate our approach and the four baseline approaches record the precision@k, recall@k, top-k accuracies (k = 1, 5 and 10), MRR, and MAP. To check if the differences in the performance of *CLRQR* and the baseline approaches are statistically significant, for the each dataset, we apply the Wilcoxon signed-rank test [31] at 95% significance level on the paired data which corresponds to the precision@k, recall@k, top-k accuracies, MRR, and MAP scores of our approach and the best baseline approach respectively.

Result. Tables 6 presents the Precision@k, Recall@k, Top-k accuracies (k=1, 5 and 10), MRR and MAP for *CLRQR* compared with the four baseline approaches, respectively. We notice that our *CLRQR* achieves the best performance in all the evaluated metrics by a substantial margin, compared with the four baseline approaches.

From Table 6, we notice that the BaselineApproach3 achieves the best performance among the four baseline approaches, while the *BaselineApproach2* achieves the worst performance (indeed, all the metrics of the *Baseline Approach2* are 0). The main cause for the poor performance of the *BaselineApproach2* is that Stack Overflow search engine is very sensitive to input query. For example, the retrieval result of "difference between A and B" is different from "difference between B and A". The sensitiveness is also illustrated in Figure 3 and Figure 4. Thus, retrieving relevant question by Stack Overflow search engine is very difficult for those developers whose native language is not English.

Tables 6 shows that *BaselineApproach1* outperforms *BaselineApproach2*, and *BaselineApproach3* outperforms *BaselineApproach4*. *BaselineApproach1* and *BaselineApproach3* translate only the title of a given Chinese question. *BaselineApproach2* and *BaselineApproach4* translate both the title and description of a given Chinese question. The title and description together usually contain more Chinese words. Because general Chinese-English translation does not know how to translate Chinese words into domain-specific English words, the more the translated words, the more translation errors are accumulated, which affects the retrieval accuracy. However, it is necessary to consider the question description, because only the title is often not enough to express the technical details of a question. This result confirms the need to help developers make domain-specific translation. We also notice that *BaselineApproach3* outperforms *BaselineApproach1* and *BaselineApproach4* outperforms *BaselineApproach2*. *BaselineApproach1* and *BaselineApproach2* retrieve relevant questions using Stack Overflow search engine. *BaselineApproach3* and *BaselineApproach4* retrieve relevant questions using Google search engine. We find that Google search engine is more general than Stack Overflow. Comparing the best baseline approach (i.e., *BaselineApproach3*) with our approach, we notice that *CLRQR* achieves precision@1, precision@5, precision@10, recall@1, recall@5, recall@10, top-1 accuracy, top-5 accuracy, top-10 accuracy, MRR and MAP of 0.56, 0.42, 0.32, 0.10, 0.35, 0.49, 0.56, 0.69, 0.71, 0.62, and 0.26 which outperforms the best baseline approach (i.e., *Baseline Approach3*) by 60%, 121%, 167%, 100%, 133%, 145%, 60%, 28%, 20%, 41% and 271%, respectively. We find that our approach which makes good use of the domain-specific knowledge in the question title and description effectively improves the retrieval accuracy.

We apply Wilcoxon signed-rank test to compare our approach with the best baseline approach(i.e., *BaselineApproach3*). Table 7 shows that the improvement of our *CLRQR* over the *BaselineApproach3* is significant at the confidence level of 95% score.

CLRQR outperforms the four baseline approaches, and the statistical tests show the improvements are significant.

RQ2: ***How is the time efficiency of our CLRQR system?***

Motivation. The runtime efficiency of the proposed method will affect its practical usage. In our approach, the main time cost is to build domain-specific vocabulary and to retrieve relevant questions on the fly. The time for building domain-specific vocabulary refers to the time required to count the term frequency of each word in our dataset. Question retrieval time refers to the time to translate Chinese keywords on-the-fly, compute keyword weights, formulate English query, and finally retrieve relevant questions in question repository.

Approach. The experimental environment is an Intel(R) Core(TM) i7 2.5 GHz laptop with 16GB RAM running OS X Version 10.11.1 (64-bit). To answer RQ2, we record the start time and the end time of program execution to obtain time performance of our approach.

Result. For building domain-specific vocabulary, it takes about 8 seconds to analyze 30,000 questions. For question retrieval time, on average our system, which runs on a single laptop, needs about 14 seconds to return relevant questions in the repository of 684,599 questions which is running only on a single laptop. Since, the relevance calculation between a given query and each English question in repository is independent with one another, it is possible to reduce question retrieval time by distributed computing.

CLRQR system is efficient. The domain-specific vocabulary building time can be completed in about 8 seconds. Question retrieval can be completed in about 14 seconds.

Table 6: Precision@k, Recall@k, Top-k Accuracies (k=1, 5 and 10), MRR and MAP for *CLRQR* compared with the Other Four Baseline Approaches

Approach	Precision@1	Precision@5	Precision@10	Recall@1	Recall@5	Recall@10
CLRQR	**0.56**	**0.42**	**0.32**	**0.10**	**0.35**	**0.49**
Baseline Approach1	0.20	0.09	0.06	0.03	0.06	0.08
Baseline Approach2	0.00	0.00	0.00	0.00	0.00	0.00
Baseline Approach3	0.35	0.19	0.12	0.05	0.15	0.2
Baseline Approach4	0.23	0.08	0.05	0.03	0.05	0.08

Approach	Top-1 Accuracy	Top-5 Accuracy	Top-10 Accuracy	MRR	MAP
CLRQR	**0.56**	**0.69**	**0.71**	**0.62**	**0.26**
Baseline Approach1	0.20	0.25	0.25	0.22	0.04
Baseline Approach2	0.00	0.00	0.00	0.00	0.00
Baseline Approach3	0.35	0.54	0.59	0.44	0.07
Baseline Approach4	0.23	0.25	0.29	0.24	0.04

Table 7: P-values of *CLRQR* compared with the Best Baseline Approaches (i.e., *Baseline Approach3*)

	Precision@1	Precision@5	Precision@10	Recall@1	Recall@5	Recall@10
P-value	$2.752e^{-7}$	$1.463e^{-6}$	$9.554e^{-8}$	$6.221e^{-2}$	$1.64e^{-4}$	$3.541e^{-5}$

	Top-1 Accuracy	Top-5 Accuracy	Top-10 Accuracy	MRR	MAP
P-value	$3.187e^{-3}$	$2.415e^{-2}$	$6.055e^{-2}$	$4.403e^{-3}$	$1.434e^{-5}$

6. DISCUSSION

This section presents the qualitative analysis of an example using our approach and discusses threats to validity of our experiment.

6.1 Qualitative Analysis

Here, we want to perform a qualitative analysis to illustrate why our approach performs better than the baseline approaches. Figure 8 presents a Chinese question from V2EX[5] (a Chinese Q&A website for computer programming). The meaning of this Chinese question is to ask for the route error of Play framework. Our approach extracts essential information from the Chinese question and then translate Chinese keywords to simulate the English query that users may enter as follows (sorted by their weights in descending order):

play, framework, route, error, api, spring

Then our approach returns three relevant questions in our question repository. The result shows that the extracted keywords well represent the core issue of the given question. We observe that our approach can capture Chinese and English keywords well at the same time if the Chinese question askers state their question clearly and sum up the core issue in the title. Moreover, we find that the sentence in question description are incomplete, and thus it does not have much impact on our keyword extraction. On the other hand, if we translate the title of this Chinese question into English by Youdao translation API and then retrieve relevant questions in Stack Overflow by Google search engine, it returns only one relevant question.

6.2 Threats to Validity

There are several threats that may potentially affect the validity of our study. Threats to internal validity relate to errors in our experiments and implementation. We have double checked our experiments and implementation. We have also manually checked the retrieved questions in our dataset to ensure that they are really tagged with *java*. Still, there could be errors that we have not noticed.

Threats to external validity relates to the generalisability of our results. We conducted an user study to evaluate whether the retrieved relevant questions are actually relevant or not. To reduce this threat, we invite 5 master students for the user study. All participants have industrial experience in Java programming (ranging from 3–6 years) and pass the national College-English-Test Level 4. In the future, we plan to reduce this threat further by analyzing more Chinese questions and building larger English question repository.

[5]https://www.v2ex.com/t/137913

Figure 8: A Chinese Question on *V2EX*

7. RELATED WORK

In this section, we first review some previous studies on software information sites. Next, we describe cross-language studies in software engineering. Finally, we give a review on information retrieval in software engineering.

7.1 Studies on Software Information Sites

A considerable amount of research has been done on software information sites during the last decade. Two position papers written by Storey et al. [27] and Begel et al. [6] describe the outlook of research in social media for software engineering. They propose a set of research questions about the impact of social media for soft-

ware engineering at team, project and community levels. Surian et al. collect information on SourceForge.net. They use random walk with restart (RWR) method to build a large-scale developer collaboration network to recommend suitable developers [28]. They also employ graph mining and graph matching techniques to find collaboration patterns on sourceForge.net [29]. Hong et al. compare developer social networks and general social networks and examine how developer social networks evolve over time [13].

An automatic tag recommendation method proposed by Xia et al. analyzes information objects in Stack Overflow and Freecode to recommend tags to users [33]. Wang et al. [30] extend the work of Xia et al. by proposing an approach that combines frequentist and Bayesian inference to better recommend tags to objects in StackExchange websites. Zhang et al. propose an automated approach named DupPredictor that takes a new question as input and detects potential duplicates of this question by considering multiple factors (e.g., titles, descriptions, and tags) [34]. Palakorn et al. create an observatory of software-related microblogs for people to browse many software-related microblogs and visually identify patterns [4]. Correa and Sureka use a machine learning framework to build a predictive model to judge whether a Stack Overflow question would be closed or not [8]. They also built a predictive model to detect whether a question will be deleted or not [9].

Our work is orthogonal to the above mentioned studies: we focus on retrieving relevant questions in Stack Overflow for a given Chinese question, which is different from the above mentioned studies. We observe that many Chinese developers have this need. To help these developers, we propose a new approach to retrieve relevant question in Stack Overflow automatically for a given Chinese question.

7.2 Cross-language Studies in Software Engineering

Many studies have been carried out on cross-language issues in Software Engineering. Xia et al. propose a cross-language bug localization algorithm named *CrosLocator* [32]. *CrosLocator* uses multiple translators to convert a non-English textual description of a bug report into English - each bug report would then have multiple translated versions. For each translated version, *CrosLocator* applies a bug localization technique to rank source code files. Finally, *CrosLocator* combines the multiple ranked lists of source code files. Hayes et al. propose a translation-based method for traceability recovery [11]. They use Google translator to translate Italian words into English and then recover the links. Chang and Lee present a cross-language video Q&A system i.e. CLVQ, which could process the English questions, and find answers in Chinese videos [26]. Saggion et al. focus on the resources that are made available for the research community [25]. They provide data and tools for evaluation of extractive, non-extractive, single and multi-document summarization.

Our work also focuses on cross-language question retrieval between Chinese and English and aims improve the accuracy of translation of domain-specific Chinese words based on domain knowledge derived from Stack Overflow.

7.3 Studies on Information Retrieval in Software Engineering

Many studies have been reported on information retrieval in software engineering. Marcus et al. propose an information retrieval approach to concept Location in source code [19]. Poshyvanyk et al. recast the problem of feature location in source code as a decision-making problem in the presence of uncertainty. They point out that the solution to feature location can be formulated as a combination of the opinions of different experts [22]. Canfora and Cerulo outline an approach to in automated bug assignment based on information retrieval in which they report recall levels of around 20% for Mozilla [7].

Cubranic and Murphy apply information retrieval as well as other matching techniques to recover the implicit traceability among different kinds of artifacts of open source projects (i.e., source file revisions, change or bug tracks, communication messages, and documents) [35]. They develop the Hipikat tool which can recommend some software development artifacts to newcomers under current tasks. Lucia et al. observe that the main drawback of existing software artifact management systems is the lack of automatic or semi-automatic traceability link generation and maintenance [18]. Hence, they improve an artifact management system with a traceability recovery tool based on Latent Semantic Indexing (LSI), an information retrieval technique. Even more, they assess the strengths and limitations of LSI for traceability recovery and devise the need for an incremental approach. Kluck and Gey describe the domain-specific cross-language information retrieval (CLIR) task of CLEF, why and how it is important and how it differs from general cross-language retrieval problem associated with the general CLEF collections [14].

Our work mainly focuses on how to adapt information retrieval technology with the heuristic for extracting information from different parts of a questions for better retrieving and ranking relevant questions.

8. CONCLUSION AND FUTURE WORK

We propose a new approach to retrieve relevant English questions for a given Chinese question. We mine domain-specific knowledge from Stack Overflow to improve the accuracy of translation of domain-specific Chinese terms. Considering the difference between question title and description, we assign terms from title and descriptions with different weights in query formulation and query retrieval. The steps of our system can be automated, and thus it can help developers save time in terms of translation, query formulation, and question retrieval. As a result, it could help Chinese developers improve their efficiency to solve the technical problem. Our proposed framework is general. It can be used for natural languages, not limited to Chinese and English. Thus, we can use our approach to connect the Q&A resources in different localized versions of Stack Overflow or Q&A sites for computer programming in different languages.

Some retrieved questions are not very relevant due to the limitation of the question repository we build for experimentation. We can crawl more questions on Stack Overflow or some other Q&A sites. This will help our proposed approach improve the relevance and usefulness of retrieved questions. Furthermore, we observe that most of the questions in Stack Overflow contain code. In the current approach, we treat code as natural language text. In the future, we can extract code segments of questions and analyze them in a different way from regular English texts. Another improvement is to expand our domain-specific stop words list and vocabulary to improve the accuracy of translation.

Acknowledgment

This research was supported by NSFC Program (No.61572426) and National Key Technology R&D Program of the Ministry of Science and Technology of China under grant 2015BAH17F01. Moreover, we would like to thank the anonymous reviewers for their helpful feedback and suggestions.

9. REFERENCES

[1] Ictclasnlp. *Available at http://ictclas.nlpir.org/docs, 2015.*

[2] Stackoverflow developer survey. *http://StackOverflow.com/research/developer- survey- 2015, 2015.*

[3] R. M. Aceves-Pérez, M. Montes-y Gómez, and L. Villaseñor-Pineda. Enhancing cross-language question answering by combining multiple question translations. In *Computational Linguistics and Intelligent Text Processing*, pages 485–493. Springer, 2007.

[4] P. Achananuparp, I. N. Lubis, Y. Tian, D. Lo, and E.-P. Lim. Observatory of trends in software related microblogs. In *Proceedings of the 27th IEEE/ACM International Conference on Automated Software Engineering*, pages 334–337. ACM, 2012.

[5] R. Baeza-Yates, B. Ribeiro-Neto, et al. *Modern information retrieval*, volume 463. ACM press New York, 1999.

[6] A. Begel, R. DeLine, and T. Zimmermann. Social media for software engineering. In *Proceedings of the FSE/SDP workshop on Future of software engineering research*, pages 33–38. ACM, 2010.

[7] G. Canfora and L. Cerulo. How software repositories can help in resolving a new change request. *STEP 2005*, page 99, 2005.

[8] D. Correa and A. Sureka. Fit or unfit: analysis and prediction of'closed questions' on stack overflow. In *Proceedings of the first ACM conference on Online social networks*, pages 201–212. ACM, 2013.

[9] D. Correa and A. Sureka. Chaff from the wheat: Characterization and modeling of deleted questions on stack overflow. In *Proceedings of the 23rd international conference on World wide web*, pages 631–642. ACM, 2014.

[10] H. Cui, J.-R. Wen, J.-Y. Nie, and W.-Y. Ma. Probabilistic query expansion using query logs. In *Proceedings of the 11th international conference on World Wide Web*, pages 325–332. ACM, 2002.

[11] J. H. Hayes, H. Sultanov, W. K. Kong, and W. Li. Software verification and validation research laboratory (svvrl) of the university of kentucky: traceability challenge 2011: language translation. *Selab.netlab.uky.edu*, pages 50–53, 2011.

[12] D. Hiemstra, F. de Jong, and W. Kraaij. A domain specific lexicon acquisition tool for cross-language information retrieval. 1997.

[13] Q. Hong, S. Kim, S. Cheung, and C. Bird. Understanding a developer social network and its evolution. In *Software Maintenance (ICSM), 2011 27th IEEE International Conference on*, pages 323–332. IEEE, 2011.

[14] M. Kluck and F. C. Gey. The domain-specific task of clef - specific evaluation strategies in cross-language information retrieval. In *In C. Peters(Ed.), Proceedings of the CLEF 2000 evaluation forum*, pages 48–56, 2001.

[15] M. Kluck and F. C. Gey. The domain-specific task of clef-specific evaluation strategies in cross-language information retrieval. In *Cross-Language Information Retrieval and Evaluation*, pages 48–56. Springer, 2001.

[16] W. Kraaij, J.-Y. Nie, and M. Simard. Embedding web-based statistical translation models in cross-language information retrieval. *Computational Linguistics*, 29(3):381–419, 2003.

[17] X. Liu, Y. Gong, W. Xu, and S. Zhu. Document clustering with cluster refinement and model selection capabilities. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 191–198. ACM, 2002.

[18] A. D. Lucia, F. Fasano, R. Oliveto, and G. Tortora. Recovering traceability links in software artifact management systems using information retrieval methods. acm transactions on software engineering and methodology, 16(4), 50. *Acm Transactions on Software Engineering & Methodology*, 16, 2007.

[19] A. Marcus, A. Sergeyev, V. Rajlich, J. Maletic, et al. An information retrieval approach to concept location in source code. In *Reverse Engineering, 2004. Proceedings. 11th Working Conference on*, pages 214–223. IEEE, 2004.

[20] A. Peñas, B. Magnini, P. Forner, R. Sutcliffe, Á. Rodrigo, and D. Giampiccolo. Question answering at the cross-language evaluation forum 2003–2010. *Language resources and evaluation*, 46(2):177–217, 2012.

[21] M. Porter. An algorithm for suffix stripping. *Program, 1980.*

[22] D. Poshyvanyk, Y.-G. Gueheneuc, A. Marcus, G. Antoniol, and V. C. Rajlich. Feature location using probabilistic ranking of methods based on execution scenarios and information retrieval. *Software Engineering, IEEE Transactions on*, 33(6):420–432, 2007.

[23] X. Qiu, Q. Zhang, and X. Huang. Fudannlp: A toolkit for chinese natural language processing. In *ACL (Conference System Demonstrations)*, pages 49–54. Citeseer, 2013.

[24] P. Resnik and I. D. Melamed. Semi-automatic acquisition of domain-specific translation lexicons. In *Proceedings of the fifth conference on Applied natural language processing*, pages 340–347. Association for Computational Linguistics, 1997.

[25] H. Saggion, D. Radev, S. Teufel, W. Lam, and S. M. Strassel. Developing infrastructure for the evaluation of single and multi-document summarization systems in a cross-lingual environment. *Ann Arbor*, 1001:48109–1092, 2002.

[26] D. Shepherd, L. Pollock, and T. Tourwé. Using language clues to discover crosscutting concerns. *Acm Sigsoft Software Engineering Notes*, 30:1–6, 2005.

[27] M.-A. Storey, C. Treude, A. van Deursen, and L.-T. Cheng. The impact of social media on software engineering practices and tools. In *Proceedings of the FSE/SDP workshop on Future of software engineering research*, pages 359–364. ACM, 2010.

[28] D. Surian, N. Liu, D. Lo, H. Tong, E.-P. Lim, and C. Faloutsos. Recommending people in developers' collaboration network. In *Reverse Engineering (WCRE), 2011 18th Working Conference on*, pages 379–388. IEEE, 2011.

[29] D. Surian, D. Lo, and E.-P. Lim. Mining collaboration patterns from a large developer network. In *Reverse Engineering (WCRE), 2010 17th Working Conference on*, pages 269–273. IEEE, 2010.

[30] S. Wang, D. Lo, B. Vasilescu, and A. Serebrenik. Entagrec: an enhanced tag recommendation system for software information sites. In *Software Maintenance and Evolution (ICSME), 2014 IEEE International Conference on*, pages 291–300. IEEE, 2014.

[31] F. Wilcoxon. Individual comparisons by ranking methods. *Biometrics bulletin*, pages 80–83, 1945.

[32] X. Xia, D. Lo, X. Wang, C. Zhang, and X. Wang. Cross-language bug localization. In *Proceedings of the 22nd International Conference on Program Comprehension*, pages 275–278. ACM, 2014.

[33] X. Xia, D. Lo, X. Wang, and B. Zhou. Tag recommendation

in software information sites. In *Proceedings of the 10th Working Conference on Mining Software Repositories*, pages 287–296. IEEE Press, 2013.

[34] Y. Zhang, D. Lo, X. Xia, and J.-L. Sun. Multi-factor duplicate question detection in stack overflow. *Journal of Computer Science and Technology*, 30(5):981–997, 2015.

[35] D. Čubranić and G. C. Murphy. Hipikat: recommending pertinent software development artifacts. In *Software Engineering, 2003. Proceedings. 25th International Conference on*, pages 408–418, 2003.