Journal of the Korean Society of Marine Engineering, Vol. 42, No. 2, pp. 97~105, 2018
J. Korean Soc. of Marine Engineering (JKOSME)
https://doi.org/10.5916/jkosme.2018.42.2.97

ISSN 2234-7925 (Print)
ISSN 2234-8352 (Online)
Original Paper

Automatic detection of similar questions from QA database using predicate-argument structures and answer sentences

Yong-Seok Choi[1] · Kong-Joo Lee†

(Received January 31, 2018 ; Revised February 12, 2018 ; Accepted February 19, 2018)

Abstract: This work deals with duplicated questions in a Question and Answer (QA) database. A large volume of QA databases can be easily built from Community Question Answering (CQA) services. Questions and their corresponding answers collected from the Internet tend to be duplicated. In this work, we propose two approaches for detecting similar questions in a QA database. One is by using predicate-argument structures. Here, questions are converted to predicate-argument structures modified for an interrogative sentence. We can detect similar questions by comparing their predicate-argument structures. The other approach is to exploit answer sentences. When two questions with different predicate-argument structures have duplicated answers, these two predicate-argument structures are interchangeable in terms of detecting similar questions. We build the QA database on a computer technology domain, which consists of 6,319 QA pairs. We evaluate our module on this QA database, and obtain a 91.69 F1 accuracy in detecting sets of similar questions.

Keywords: Question-and-answer, QA database, similar question, predicate argument structure, community question answering service

1. Introduction

Research on question generation have been performed widely for educational purposes in recent years **[1]**. The automatic generation of a question is a task that has a very significant role in education by providing materials that allow practice and assessment **[2]**.

One of the obstacles for English learners in improving speaking skills is the lack of exposure to real conversational English. When learners want to prepare for English interviews such as a job interview or a university admission interview, it would be helpful if a system can automatically provide the learners with interview-related questions. In this paper, we introduce a system whose main purpose is to help learners improve their English-speaking skills in an interview situation. The questions are generated by extracting them from Question-and-Answer (QA) databases with a specific domain knowledge.

In general, the process of automatic question generation is as follows: the first step is to decide what the question is about. Once the content to be asked is determined, the next step is to decide how to form a question. Most question generation systems share common techniques on the process of question formation **[1]**. A question is generated by transforming a declarative sentence into an interrogative one, or by filling a question template with information to be asked.

In this paper, we combine the two problems—what to ask and how to form a question—into a retrieval task from a question-answer database. We adopt a retrieval-based approach **[4]**, which is widely used in a short-text conversation (STC) research field, because this approach automatically generates the most suitable response to a user's utterance by retrieving it from the QA databases **[3]**. While a STC task should retrieve an answer from QA databases, our task should retrieve a question from QA databases.

A QA database consists of pairs of a question and the corresponding answer. A learner's prior answer $\hat{A}$ is compared with a set of answers in QA databases based on various similarity measurements. Then the answer A in the (Q-A) pair that is most similar to $\hat{A}$ is selected and the corresponding question Q can be given as a next question to a learner.

† Corresponding Author (ORCID: http://orcid.org/0000-0003-0025-4230): Department of Radio Sciences & Engineering and Information Communications Engineering, Chungnam National University, 99, Daehak-ro, Yuseong-gu, Daejeon 34134, Korea, E-mail: kjoolee@cnu.ac.kr, Tel: 042-821-5662

1 Department of Radio Sciences & Engineering and Information Communications Engineering, Chungnam National University, E-mail: yseokchoi@cnu.ac.kr, Tel: 042-821-6869

Table 1: Examples of similar questions and answers collected from CQA services

Question	Answer
Q1: What is daemon thread?	**A1**: Daemon thread is a low priority thread, which runs intermittently in the back ground doing the garbage collection operation for the Java runtime system.
Q2: What about daemon threads?	**A2**: The daemon threads are basically the low priority threads that provide the background support to the user threads. It provides services to the user threads.
Q3: What is the purpose of daemon thread?	**A3**: In Java, any thread can be a Daemon thread. Daemon threads are used for background supporting tasks and are only needed while normal threads are executing.
Q4: What does it mean to be a daemon thread?	**A4**: Daemon threads in Java are similar toa service provider for other threads or objects running in the same process as the daemon thread. Daemon threads are used for background supporting tasks and are only needed while normal threads are executing.

Owing to the fast growth of Community Question Answering (CQA) services, a large-scale question and answer databases can be easily collected. These databases are one of the most useful knowledge sources, and many applications are developed based on them. In this work, we exploit databases as a source for generating questions that are presented to a learner. Question-Answer pairs are prone to be duplicated because they are collected from multiple CQA services. **Table 1** shows examples of similar question-answer pairs. They look different from each other though they are the same questions. Therefore, the same (or similar) questions should be detected in advance in order to avoid giving the same question to a learner redundantly.

In this paper, we introduce two approaches for detecting similar questions. One is using predicate-argument structures of questions. A dependency tree of a question is converted into a predicate-argument structure (PAS) that is adjusted for an interrogative sentence. By comparing predicate-argument structures of questions, we can identify if they are similar or not. The second approach is to adopt a "similar-question pattern." Two questions with different predicate-argument structures can be considered similar when they have duplicated answers. We define these two different predicate-argument structures as 'similar-question pattern' in this work. We can decide that two questions are similar when their PASs have the same similar-question pattern.

In Chapter 2, the related study is reviewed and our approaches to finding similar questions are presented in Chapter 3. Experimental results are described in Chapter 4, and final remarks are made in Chapter 5.

2. Related Work

Generally speaking, similarity of sentences including questions can be determined in three aspects: words, syntactic structures, and semantic topics.

The basic idea of measuring similarity using words is based on the bag of words (BoW) concept. A question consisting of words can be represented by the sum of the meaning of the words. The most popular approach is a vector space model that adopts term frequency (TF) and inverse document frequency (IDF) **[5][6]**. Several distance measurements between two vectors, such as Manhattan distance, Euclidean distance, cosine normalized dot product, Jaccard similarity coefficient, dice measurement, and Jenson-Shannon divergence have been proposed **[7]**. This approach overlooks the importance of word sense, word order and syntactic information **[8]**.

Syntactic parsers are adopted in order to calculate similarity between questions. Questions are analyzed into a syntactic parses and similarity calculation are measured on these parse trees **[9]**. These approaches are vulnerable to parser's incompleteness.

Cao et al., suggest a question topic and a question focus to find similar question pairs **[10]**. A question topic usually means the major domain of a question that reflects users' interests. In contrast, a question focus indicates certain features of the question topic. They adopt MDL-based (Minimum Description Length) tree cut model for automatically identifying question topic and question focus. MDL is a principle of data compression and statistical estimation from information theory.

3. Automatic Detection of Similar Questions

Figure 1 shows the overall system architecture. The system asks a learner a question taking his/her previous response into account. A question is retrieved from QA database, and it should be deliberately excluded if a similar question is already presented to a learner. Therefore, we preprocess duplicated questions beforehand when building a QA database. In **Figure 1**, 'Question Filter-I' is a module that can detect duplicated

questions in a QA database. We focus on 'Question Filter-I' in this paper. The other topics will be given in a successive paper in preparation.

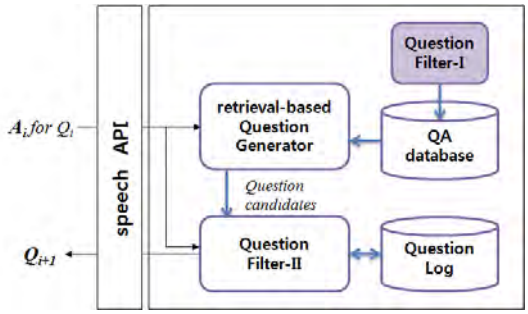

Figure 1: The overall system architecture

3.1 Detection of Similar Questions using Predicate-Argument Structures

A large collection of question-answer pairs can be easily built from Community Question Answering (CQA) services such as Yahoo! Answers (http://answers.yahoo.com), WikiAnswer (http://wiki.answers.com), and Naver KnowledgeiN (http://kin.naver.com/index.nhn). The CQA service has become one of the most useful sources for easy acquisition of knowledge and information on a wide range of topics.

When a large size of question-answer pairs is built from several CQA services, the collected question-answer pairs tend to be duplicated **[11]**. In many applications, repetition of duplicated question-answer pairs makes processes difficult, thus it is helpful to group them by the same question topics. If two questions are not identical in their surface strings, but their topics are the same, the two questions are said to be ***'similar'*** in this work. The object of this work is to find sets of similar questions from a QA database. To find similar question-answer pairs more accurately, we transform them into canonical forms, one of which is a predicate-argument structure.

The following procedure is for converting questions to predicate-argument structures.

(1) Analyze question sentences syntactically and output dependency trees.
(2) Analyze answer sentences syntactically and output dependency trees.
(3) Convert the result of (1) into predicate-argument structures.
(4) Extract topic words from the result of (2).

We use the Stanford parser **[12]** to parse question and answer sentences. A parser's output, a dependency tree, is converted into a predicate-argument structure (PAS) by using the transformation rules **[13]**. The format of predicate-argument structure we use in this work is as follows:

[SUBJ]-**PREDICATE**-[OBJ]

Table 2 is a resultant dependency tree for the input sentence 'How the locks can be classified?'. As the main verb of the sentence is 'classify' and the interrogative 'how' is an adjunctive argument of the main verb, 'how' is not included in a predicate-argument structure. As a predicate-argument structure focuses on both a subject and an object of a main predicate in a sentence, an adjunctive word such as an adverb is generally not included in PAS. However, interrogative words such 'what', 'how', and 'why' play an important role in a question sentence even though their syntactic roles are adjunctive. Therefore, we modify a basic predicate-argument structure in order to include an interrogative word into it.

Table 2: The parse tree for the example sentence

Question: How the locks can be classified?						
	words			dependency tree		
	surface	lemma	POS	relation	head	dependent
1	How	how	WRB	root	0	6
2	The	the	DT	advmod	6	1
3	locks	lock	NNS	det	3	2
4	can	can	MD	nsubjpass	6	3
5	be	be	VB	aux	6	4
6	classified	classify	VBN	auxpass	6	5
7	?	?	.	punct	6	7

We convert a syntactic dependency tree for a question into a predicate-argument structure (PAS) by the following basic rules. In a dependency tree, a root (verb) node becomes a main predicate **'PREDICATE'**, and the nodes connected to the root by the relations 'subj' and 'obj' become [SUBJ] and [OBJ], respectively. Apart from these basic rules, additional rules are necessary for retaining question-related information in a PAS. The followings are the additional processes on PAS:

(1) Chunk multi-word interrogative expressions into a node:
ex) How many types of operator support?
⇒ [how_many(type_of_opetator)]-**support**-[]

(2) Includes an interrogative adjective that modify a main subject or object:
ex) Which class acts as a base class for all arrays in C#?
⇒ [which(class)]-**act**-[]

(3) Convert a passive form into an active form:
- change relation 'nsubjpass' into 'OBJ'
- change relation 'nmod:agent' into 'SUBJ'
- change the tense of a main VERB from 'VBN' to 'VB'

ex) How the locks can be classified? ⇒ []-**classify**-[lock]
ex) Which query language is used by Neo4J?
⇒ [Neo4J]-**use**-[which(query_language)]

(4) Include an adjunctive interrogative word in the 4th additional slot of PAS
ex) How the locks can be classified? ⇒ [_]-**classify**-[lock]-*how*

(5) When a subject (or an object) contains the property of a topic word,
represent it as 'property(topic)'
ex) What are the advantages of using SVG?
⇒ [advantage(using)]-**what**-[]
property: ('purpose', 'difference', 'use', 'type', 'advantage', 'usage', 'feature', 'class', 'pattern', 'function', 'method', 'component', 'object', 'variable', 'value', 'benefit', 'disadvantage', 'role', 'scope', 'property', 'importance')

(6) Process a few of exceptions:
ex) 'you/we_mean_by': What do you mean by synchronicity?
⇒ [synchronicity]-**mean**-[what]
'be_used_to_VERB':
What command is used to rename the database?
⇒ [what(command)]-**rename**-[database]

(7) Replace all topic words with 'TOPIC' and all pronouns with None:
ex) How the locks can be classified? ⇒ []-**classify**-[lock]-*how*
⇒ []-**classify**-[TOPIC]-how
ex) When you will face SQLCODE-911?
⇒ [you]-**face**-[TOPIC]-when ⇒ []-**face**-[TOPIC]-when

Table 3 shows examples of questions and their corresponding predicate-argument structures.

The following two questions have the same PAS, hence they are similar questions.

ex) How the locks can be classified? ⇒ []-**classify**-[lock]-how
How can you classify the locks? ⇒ []-**classify**-[lock]-how

However, the following questions are similar though their PASs are different. Therefore, we need to define interchangeable predicate-argument structures considering an interrogative sentence. The following section deals with these cases.

ex) What do you mean by program reentrance?
⇒ [program_reentrance]-**mean**-[*what*]
What is program reentrance? ⇒ [program_reentrance]-**what**-[]

ex) What is the purpose of using PL/SQL?
⇒ [purpose(TOPIC)]-**what**-[]
What is the advantage of using PL/SQL?
⇒ [advantage(TOPIC)]-**what**-[]

Table 3: Examples of questions and their predicate-argument structures

Questions	PAS (predicate-argument structure)
What is the difference between local variable and global variable in C?	[difference(TOPIC1, TOPIC2)]-**what**-[] TOPIC1: local variable TOPIC2: global variable
What are the advantages of using SVG?	[advantage(TOPIC)]-**what**-[] TOPIC: using SVG
What is the purpose of using PL/SQL?	[purpose(TOPIC)]-**what**-[] TOPIC: using PL/SQL
How the locks can be classified?	[]-**classify**-[TOPIC]-how TOPIC: lock
What is deadlock?	[TOPIC]-**what**-[] TOPIC: deadlock
How many layers are in TCP/IP?	[how_many(TOPIC)]-**be**-[] TOPIC: layer
How can you handle exceptions in QTP?	[]-**handle**-[TOPIC]-how TOPIC: exception
Which number is denoted by leading zero in Java?	[]-**denote**-[which(TOPIC)] TOPIC: number
Which class acts as a base class for all arrays in C#?	[which(TOPIC)]-**act**-[] TOPIC: class
Does MongoDB remove it from disk?	[TOPIC]-**remove**-[] TOPIC: MongoDB
How can you update Memcached when data changes?	[]-**update**-[TOPIC]-how TOPIC: Memcached
When you will face SQLCODE-911?	[]-**face**-[TOPIC]-when TOPIC: SQLCODE-911
If a transaction takes anupdate lock on some data, then other transactionscan get what type of lock ?	[TOPIC1]-**get-**[what(TOPIC2)] TOPIC1: transaction TOPIC2: type_of_lock
Which method is used to save the state of the given instance from the underlying database?	[which(TOPIC1)]-**save**-[TOPIC2] TOPIC1: method TOPIC2: state_of_instance
Which header of HTTP response provides the date of the resource?	[which(TOPIC1)]-**provide**-[TOPIC2] TOPIC1: header_of_HTTP_response TOPIC2: data_of_resource
What is meant by program reentrance?	[TOPIC]-**mean**-[what] TOPIC: program_reentrance
How XForms is used to separate data from presentation?	[TOPIC1]-**separate**-[TOPIC2]-how TOPIC1: XFormTOPIC2: data

3.2 Detection of Similar Questions using Answer Sentences

Even though predicate-argument structures of questions are different from each other, their answers can be duplicated then in this case two questions are similar. Questions with different predicate-argument structures but with the same (or duplicated) answers are grouped into a cluster. Their PASs are different, however they are regarded as the same one. In this work, we define those PASs as 'similar-question pattern'. **Table 6** displays examples of similar-question patterns. We create 13 similar-question patterns containing 45 predicate-argument structures.

In this paper, we consider two question-answer (Q, A) and (Q', A') to be similar when the following two conditions are met simultaneously.

Condition 1: PAS(Q) == PAS(Q') or
PAS(Q) and PAS(Q') are in the same similar-question pattern
Condition 2: TOPIC(Q) == TOPIC(Q') or
Jaccard coefficient (TOPIC(A), TOPIC(A')) > threshold

4. Experimental Results

4.1 QA database of the computer technology domain

We build a QA database using the computer technology domain from the four CQA sites. Computer technology is one of the largest domains where we can collect QA pairs from the Internet. The four CQA sites are the followings:

https://www.javatpoint.com/interview-questions-and-answers
https://www.tutorialspoint.com/tutorialslibrary.htm

https://career.guru99.com/top-100-core-java-interview-questions/
https://www.javacodegeeks.com/2014/04/java-interview-questions-and-answers.html

Table 4: QA database for the computer technology domain

sub-domain	number of QA pairs
C/C++/C#/Java Programming languages	885
Other Programming languages	1,155
Database	700
Web related	1,217
JavaTech	880
etc.	1,482
TOTAL	6,319

The collected question-answer pairs are classified into 6 sub-domains. **Table 4** depicts the number of the QA pairs we collected in this work. The total number of the QA pairs is 6,319. **Table 5** shows the examples of the QA pairs. The average number of sentences in questions and answers are 1.04 and 2.25, respectively. The average numbers of words in questions and answers are 9.13 and 15.99, respectively.

4.2 Evaluation of Detecting Sets of Similar Questions

The overall questions and answers in the database are analyzed syntactically first, and then their resultant dependency trees are converted into predicate-argument structures. For each question (Q, A) we find sets of similar questions that meet the two conditions described in Section 3.2. In this experiment, the threshold value in Condition 2 is set to 0.7.

Table 5: Examples of questions and answers in the QA database

sub-domain	Question	Answer
Java	What is Function Overriding and Overloading in Java ?	Method overloading in Java occurs when two or more methods in the same class have the exact same name, but different parameters. On the other hand, method overriding is defined as the case when a child class redefines the same method as a parent class
C	What is the difference between method overloading and method overriding in C#?	Method parameters must be different in method overloading whereas it must be same in method overriding.
C	How C# supports static polymorphism?	Static polymorphism. They are: Function overloading Operator overloading
C	What is polymorphism?	In object-oriented programming paradigm, polymorphism is often expressed as 'one interface, multiple functions'.
Java	Can we override a method by using same method name and arguments but different return types?	The basic condition of method overriding is that method name, arguments as well as return type must he exactly same as is that of the method being overridden.
Java	Can we use different return types for methods when overridden?	The basic requirement of method overriding in Java is that the overridden method should have same name, and parameters
Database	What are the advantages of using a package?	Modularity Easy to design the applications Better performance Hiding information Added functionality Overloading
Java	What is covariant return type?	Since java5, it is possible to override any method by changing the return type if the return type of the subclass overriding method is subclass type.
C	Distinguish between shallow copy and deep copy.	Deep copy is achieved using copy constructor and or overloading assignment operator.
Programming	What is the use of final keyword?	PHP 5 introduces the final keyword, which prevents child classes from overriding a method by prefixing the definition with final.

Table 6 shows the examples of the sets of similar questions identified in this work.

The sets (1)–(9) are correctly detected as similar questions. The result (10) can be sets of similar questions from a learner's point of view because learners generally learn all the questions in set (10) at once. The questions in set (11) deal with the same topic though they ask about different languages.

Sets (12)–(15) are detected incorrectly as similar questions. The predicate-argument structures of the two questions in set (12) are the same because the modifying phrase '(without) using recursion' is not included in the PAS. Therefore, it is incorrectly detected as a similar question set. The two questions in set (13) are not similar, however they are identified to be similar in this work. Their answers are overlapped a lot because they are programming source codes that contain a finite set of reserved programming keywords. In the case of set (14), the two different questions are detected to be similar because their PASs are the same and 'read only' and 'idempotent' are missing in their topic words. The two questions in set (15) have the same PAS, because their dependency trees are incorrectly analyzed by the parser. The words 'tell' and 'join' do not modify 'function', instead they are clausal complements of the main verb 'explain'.

For evaluation, we build reference sets of similar questions manually, in which the total number of sets of similar questions is 191, and the QA pairs in these sets are 449.All sets of similar questions are mutually exclusive. **Table 7** shows the evaluation results.

Table 6: Example of sets of similar questions detected in this work

(1)	Define an array.	What is an array?
(2)	How can you classify the locks in DB2 ?	How the locks can be classified ?
(3)	What is a column family in Cassandra ?	What do you mean by column family in Cassandra ?
(4)	What is the purpose of next statement ?	What is the purpose of continue statement ?
(5)	How can you send email in PHP ?	How will you send an email using PHP ?
(6)	When to use a SAX Parser ?	How a SAX Parser works ?
(7)	How can I return string result from Action in ASP.Net MVC ? How to return the JSON from action method in ASP.Net MVC ?	
(8)	Explain the loosely coupled architecture of web services . What do you mean by loosely coupled architecture of Web services ?	
(9)	What are the four parameters that have to be passed in Selenium ? What are the four parameters that you need to pass in Selenium ?	
(10)	What is the purpose of cmp operator? What is the purpose of le operator? What is the purpose of eq operator?	
(11)	What is the purpose of Continue statement in VB.NET ? What is the usage of continue statement in Go programming language ? What is the purpose of continue statement ?	
(12)*	Write a program to print factorial of given number without using recursion ? Write a program to print factorial of given number using recursion ?	
(13)*	What are some examples of dynamic programming algorithms ? What are some examples of divide and conquer algorithms ?	
(14)*	Which type of Webservices methods are to be read only ? Which type of Webservices methods are to be idempotent ?	
(15)*	Explain tell function in Perl ? Explain join function in Perl ?	

Table 7: Accuracy of detecting sets of similar questions

	number of sets of similar questions (number of questions)	precision	recall	F1
Model 1: PAS of questions + similar-question patterns + Answer sentences	182 (260)	93.96	89.53	91.69
Model 2: PAS of questions + similar-question patterns	146 (223)	69.78	66.49	68.10
Model 3: Answer sentences (n-gram)	133 (167)	40.66	38.74	39.68

Model 1 using both PAS, similar-question patterns and answer sentences can detect 182 similar question sets and 260 questions in these sets.

It has a 91.69 F1 accuracy. Model 3, which looks up only answer sentences to identify similar questions, has the lowest accuracy. The largest part of incorrectly detected sets in Model 3 has answer sentences on how to use a specific command with the same pattern.

5. Conclusion

In this work, we detected sets of similar questions in a QA database. Owing to the proliferation of Community Question Answering (CQA) services, it is easy to build a large size of QA databases from them. QA pairs collected from multiple knowledge sources are prone to duplication. Therefore, similar questions should be grouped together when we build a QA database. We used a predicate-argument structures of questions and their corresponding answers to detect similar questions. We also proposed similar-question patterns that are interchangeable predicate-argument structures in terms of their detection of similar questions. We evaluated our module on the QA database consisting of 6,319 QA pairs, and obtained a 91.69 F1 accuracy in detecting sets of similar questions.

Acknowledgements

This work was supported by research fund of Chungnam National University.

References

[1] L. Nguyen-Thinh, T. Kojiri, and N. Pinkwart, “Automatic question generation for educational applications–the state of art,” Advanced Computational Methods for Knowledge Engineering. Springer, Cham, pp. 325-338, 2014.

[2] J. F. Aquino, D. D. Chua, R. K. Kabiling, J. N. Pingco, and R. Sagum, “Text2Test: Question generator utilizing information abstraction techniques and question generation methods for narrative and declarative text,” Proceedings of the 8th National Natural Language Processing Research Symposium, pp. 29-34, 2011.

[3] Z. Yan, N. Duan, J. Bao, P. Chen, M. Zhou, Z. Li, and J. Zhou. “DocChat: An Information Retrieval Approach for Chatbot Engines Using Unstructured Documents,” Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, pp. 516–525, Berlin, Germany, August 7-12, 2016.

[4] J. Zongcheng, L. Zhengdong, and L. Hang. “An information retrieval approach to short text conversation,” arXiv preprint arXiv:1408.6988, 2014.

[5] J. Allan, C. Wade, and A. Bolivar, “Retrieval and novelty detection at the sentence level,” Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval, pp. 314-321, 2003.

[6] T. C. Hoad and J. Zobel, “Methods for identifying versioned and plagiarized documents,” Journal of the American Society for Information Science and Technology, vol. 54, no. 3, pp. 203-215, 2003.

[7] D. Jurafsky, Speech & Language Processing, Pearson Education India, 2000.

[8] W. N. Zhang, T. Liu, Y. Yang, L. Cao, Y. Zhang, and R. Ji. “A topic clustering approach to finding similar questions from large question and answer archives,” PloS one, vol. 9, no. 3, 2014.

[9] K. Wang, Z. Ming and T. S. Chua. “A syntactic tree matching approach to finding similar questions in community-based qa services.” In: Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval. ACM, pp. 187-194, 2009.

[10] H. Duan, Y. Cao, C. Y. Lin and Y. Yu. “Searching questions by identifying question topic and question focus.” Proceedings of ACL-08: HLT, pp. 156-164, 2008.

[11] J. Jeon, W. B. Croft and J. H. Lee. "Finding similar questions in large question and answer archives." Proceedings of the 14th ACM international conference on Information and knowledge management. ACM, pp. 84-90, 2005.

[12] D. Klein, C. D. Manning. “Accurate unlexicalized parsing.” In: Proceedings of the 41st annual meeting of the association for computational linguistics. Pp. 423-430, 2003

[13] R. Krestel, R. Witte, S. Bergler. “Predicate-argument extractor (pax).” New Challenges For NLP Frameworks Programme, pp. 51-54, 2010.