Towards Building Open Knowledge Base From Programming Question-Answering Communities

Wei Emma Zhang[1], Ermyas Abebe[2], Quan Z. Sheng[1], and Kerry Taylor[3]

[1] School of Computer Science, The University of Adelaide, Australia
[2] IBM Research Australia, Australia
[3] Research School of Computer Science, Australian National University, Australia

Abstract. In this paper, we propose the first system, so-called Open Programming Knowledge Extraction (OPKE), to automatically extract knowledge from programming Question-Answering (QA) communities. OPKE is the first step of building a programming-centric knowledge base. Data mining and Natural Language Processing techniques are leveraged to identify duplicate questions and construct structured information. Preliminary evaluation shows the effectiveness of OPKE.

1 Introduction

Recent research on Open Information Extraction (Open IE) aims to automatically extract machine-understandable structured information from unstructured natural language texts [6, 4]. Existing works extract knowledge from web corpus holding text from, for example, news and articles. No effort has been made on extracting knowledge from programming Question-Answering (QA) communities, e.g. Stack Overflow (SO)[1]. SO is a code-centric QA community that provides working code solutions to natural language queries. It serves as a knowledge resource for software developers. In this paper, we provide the first work that extracts questions and answers from SO and outputs structured triples. Our method is called Open Programming Knowledge Extraction (OPKE). It is the initial step of building a programming-centric knowledge base.

OPKE extracts questions and answers for a post and form a triple formatted as ⟨ *"question"*; *"answer"*; *"tag1", "tag2",...*⟩. The tags can be easily obtained and the answers can be obtained by choosing the accepted answers marked by the questioner. But it is challenging to extract information from questions. The key challenge is that questions with similar meanings/purpose exist. This is because questioner fail to search out answers from previous questions, so they post a new one. In Stack Overflow, such questions are marked as "Duplicate". However this marking is manually performed which could not identify all the duplicate questions and not all communities provide duplicate identification. Recent works (e.g.,[1]) perform binary classification on post pairs by considering the features of these pairs. However, these works consider every two posts as a pair, which

[1] http://stackoverflow.com/

lead to high computation cost. Instead, we propose to use features of posts themselves to identify the duplicates. Specifically, we transform every post title to a numerical vector and perform clustering on a set of vectors. Then we consider the posts that are closest to a master post (i.e., the original post) as the duplicate posts. After identifying and removing duplicate posts, we extract the key information from master posts and rewrite them to the *"question"* component of knowledge triples. This can be done by using NLP techniques. Finally, triples are constructed by given question and answer, as well as the tags.

2 Building Programming Knowledge Base

The input of OPKE is the post contents of programming QA communities and the output is a set of triples. Figure 1 illustrates the extraction process which contains three main steps: Preprocessing, Question-answer Extraction and Triple Generation described as following.

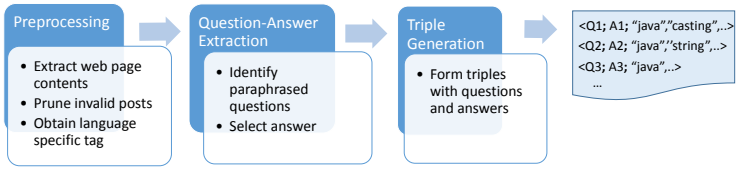

Fig. 1. Knowledge Extraction Process from Programming Community

2.1 Preprocessing

In the preprocessing step, OPKE parses the post content of the target programming community and extracts all the questions and answers. Invalid posts which have no accepted answers are pruned. For SO, post contents can be retrieved through Stack Exchange API[2]. The accepted answer and tags can be obtained by considering two parameters '*AcceptedAnswerId*' and '*Tags*' respectively.

2.2 Question-Answer Extraction

OPKE extracts the titles of the posts as the questions. It first identifies duplicate posts, then extract question information using Part Of Speech (POS) tagging and dependency parsing. The details are as follows:

– Identify duplicate questions. OPKE firstly transforms all questions to vectors leveraging word embeddings [2]. Then it performs clustering based on these vectors. We use the batch solution of K-means for processing large scale post data [5]. Then for each cluster, a K-D tree is built. We mark the K nearest neighbors of every master post as duplicate. Duplicate questions are pruned.

[2] https://api.stackexchange.com/

– Parse master questions. We parse the master questions and rewrite it to capture the meaning of the questions. Our method is inspired by Open IE work [6], where dependency parsing has been applied to detect query meaning. We discuss OPKE parsing process using an example question *"Java - How do I convert from int to String?"*: i) remove language specific words if exist (e.g. *Java*), ii) parse dependencies of questions (using method in [3], see Figure 2(a)), iii) identify subject (I), if subject is first person pronoun, then parse object part under the root in following steps (see the circled part in Figure 2(a)), iv) identify root action (*covert*), v) identify relationship ($\langle convert; fromint, tostring \rangle$) and vi) rewrite question (*"convert int to string"*). For more details, please refer to [6].

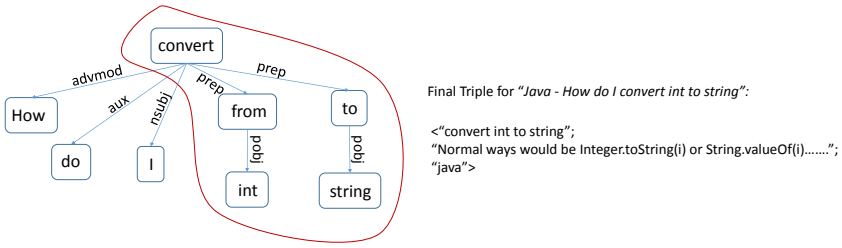

Fig. 2. Parse Questions and Generate Triples

OPKE simply uses the accepted answer as the final answer for a post. Although it might not be the best solution with highest votes, we believe that the questioner has the judgment on the solutions.

2.3 Triple Generation

After obtaining the questions and answers from previous steps, OPKE generate triples with the format of $\langle$ *"question"*; *"answer"*; *"tag1","tag2",...*$\rangle$ where question and answers are obtained from Section 2.2 and tags are obtained from Section 2.1. So for the example question, the triple is depicted as in Figure 2(b).

3 Experiment

The performance of OPKE system has been evaluated by performing extraction on Stack Overflow dump datasets[3] (the version is 2016-06-13). We choose valid posts with non-empty titles and have accepted answers as the master posts. We obtain the duplicate and linked posts of these master posts. These result in

[3] https://archive.org/details/stackexchange

339,990 posts in total. We define the evaluation metrics, clustering recall, as the number of correctly labeled posts (i.e., duplicate or not) divided by the total posts. Figure 3 reports the clustering recall rate on different number of clusters for duplicate and linked questions. Duplicate has higher recall rate than linked ones because linked posts do not always represent the same meaning with the master questions. When the cluster number is the 1/2000 of the total number of posts, the recall rate achieves the best.

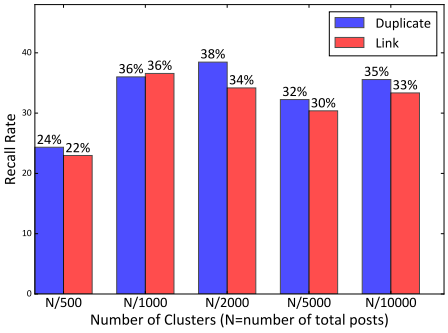

Fig. 3. Clustering Recall Rate on Various Number of Clusters

4 Conclusion and Future Works

In this work, we discuss our attempt to extract structured information from programming QA communities. The method, which combines data mining and NLP techniques, can also be applied to other QA communities. In the future, we will develop more fine-grained methods to identify paraphrased questions and consider the evolvement of programming language/tools. A programming-centric knowledge base and corresponding QA system will be built.

References

1. M. Ahasanuzzaman, M. Asaduzzaman, C. K. Roy, and K. A. Schneider. Mining Duplicate Questions in Stack Overflow. In *Proc. of MSR 2016*.
2. Q. V. Le and T. Mikolov. Distributed Representations of Sentences and Documents. In *Proc. of ICML 2014*.
3. M.-C. D. Marneffe, B. MacCartney, and C. D. Manning. Generating Typed Dependency Parses From Phrase Structure Parses. In *Proc. of LREC 2006*.
4. S. Nam, Y. Hahm, S. Nam, and K. Choi. SRDF: korean open information extraction using singleton property. In *Proc. of ISWC 2015 (Posters & Demonstrations Track)*.
5. D. Sculley. Web-Scale K-Means Clustering. In *Proc. of WWW 2010*.
6. P. Yin, N. Duan, B. Kao, J. Bao, and M. Zhou. Answering Questions with Complex Semantic Constraints on Open Knowledge Bases. In *Proc. of CIKM 2015*.