Augmenting and Structuring User Queries to Support Efficient Free-Form Code Search

Raphael Sirres[1], Tegawendé F. Bissyandé[2], Dongsun Kim[2]
David Lo[3] Jacques Klein[2], Kisub Kim[2], Yves Le Traon[2]
[1]National Library of Luxembourg - Luxembourg
[2]SnT, University of Luxembourg - Luxembourg
[3]Singapore Management University - Singapore

ABSTRACT

Motivation: Code search is an important activity in software development since developers are regularly searching [6] for code examples dealing with diverse programming concepts, APIs, and specific platform peculiarities. To help developers search for source code, several Internet-scale code search engines, such as OpenHub [5] and Codota [1] have been proposed. Unfortunately, these Internet-scale code search engines have limited performance since they treat source code as natural language documents. To improve the performance of search engines, the construction of the search space index as well as the mapping process of querying must address the challenge that “no single word can be chosen to describe a programming concept in the best way” [2]. This is known in the literature as the vocabulary mismatch problem [3].

Approach: We propose a novel approach to augmenting user queries in a free-form code search scenario. This approach aims at improving the quality of code examples returned by Internet-scale code search engines by building a Code voCaBulary (CoCaBu) [7]. The originality of CoCaBu is that it addresses the vocabulary mismatch problem, by expanding/enriching/re-targeting a user’s free-form query, building on similar questions in Q&A sites so that a code search engine can find highly relevant code in source code repositories. Figure 1 provides an overview of our approach.

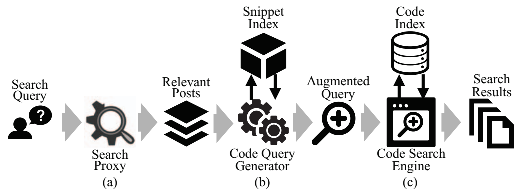

Figure 1: Overview of CoCaBu.

The search process begins with a free-form query from a user, i.e., a sentence written in a natural language:

(a) For a given query, CoCaBu first searches for relevant posts in Q&A forums. The role of the Search Proxy is then to forward developer free-form queries to web search engines that can collect and rank entries in Q&A with the most relevant documents for the query.

(b) CoCaBu then generates an augmented query based on the information in the relevant posts. It mainly leverages code snippets in the previously identified posts. The Code Query Generator then creates another query which includes not only the initial user query terms but also program elements. To accelerate this step in the search process, CoCaBu builds upfront a snippet index for Q&A posts.

(c) Once the augmented query is constructed, CoCaBu searches source files for code locations that match the query terms. For this step, we crawl a large number of repositories and build upfront a code index of program elements in the source code.

Contributions:

- CoCaBu **approach to the vocabulary mismatch problem:** We propose a technique for finding relevant code with free-form query terms that describe programming tasks, with no a-priori knowledge on the API keywords to search for.
- GitSearch **free-form search engine for GitHub:** We instantiate the CoCaBu approach based on indices of Java files built from `GitHub` and Q&A posts from `Stack Overflow` to find the most relevant code examples for developer queries.
- **Empirical user evaluation**: Comparison with popular code search engines further shows that GitSearch is more effective in returning acceptable code search results. In addition, Comparison against web search engines indicates that GitSearch is a competitive alternative. Finally, via a live study, we show that users on Q&A sites may find GitSearch’s real code examples acceptable as answers to developer questions.

Concluding remarks: As a follow-up work, we have also leveraged `Stack Overflow` data to build a practical, novel, and efficient code-to-code search engine [4].

ICSE ’18, May 27-June 3, 2018, Gothenburg, Sweden
© 2018 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-5638-1/18/05.
https://doi.org/10.1145/3180155.3182513

REFERENCES

[1] Codota. 2016. http://www.codota.com. (Mar. 2016). last accessed 12.03.2016.
[2] G. W. Furnas, T. K. Landauer, L. M. Gomez, and S. T. Dumais. 1987. The Vocabulary Problem in Human-system Communication. *Communications of the ACM (CACM)* 30, 11 (Nov. 1987), 964–971.
[3] Sonia Haiduc, Gabriele Bavota, Andrian Marcus, Rocco Oliveto, Andrea De Lucia, and Tim Menzies. 2013. Automatic Query Reformulations for Text Retrieval in Software Engineering. In *Proceedings of the 35th International Conference on Software Engineering (ICSE)*. 842–851.
[4] Kisub Kim, Dongsun Kim, Tegawende F. Bissyande, Eunjong Choi, Li Li, Jacques Klein, and Yves Le Traon. 2018. FaCoY – A Code-to-Code Search Engine. In *Proceedings of the 40th International Conference on Software Engineering (ICSE)*.
[5] OpenHub. 2016. http://code.openhub.net. (Mar. 2016). last accessed 12.03.2016.
[6] Caitlin Sadowski, Kathryn T. Stolee, and Sebastian Elbaum. 2015. How Developers Search for Code: A Case Study. In *Proceedings of the 10th Joint Meeting on Foundations of Software Engineering (FSE)*. 191–201.
[7] Raphael Sirres, Tegawende F. Bissyande, Dongsun Kim, David Lo, Jacques Klein, Kisub Kim, and Yves Le Traon. 2018. Augmenting and structuring user queries to support efficient free-form code search. *Empirical Software Engineering* (Jan. 2018).