**Rpt**: Effective and Efficient Retrieval of Program Translations from Big Code

Binger Chen  
TU Berlin  
Berlin, Germany  
chen@tu-berlin.de

Ziawasch Abedjan  
Leibniz Universität Hannover & L3S Research Center  
Hannover, Germany  
abedjan@dbs.uni-hannover.de

**ABSTRACT**

Program translation is a growing demand in software engineering. Manual program translation requires programming expertise in source and target language. One way to automate this process is to make use of the big data of programs, i.e., Big Code. In particular, one can search for program translations in Big Code. However, existing code retrieval techniques are not designed for cross-language code retrieval. Other data-driven approaches require human efforts in constructing cross-language parallel datasets to train translation models. In this paper, we present Rpt, a novel code translation retrieval system. We propose a lightweight but informative program representation, which can be back-translation method. However, by only training data but can compete with existing translation models.

In this paper, we propose Rpt, a novel program translation retrieval system. Given a raw program in a given source PL and a target PL, Rpt efficiently retrieves similar programs as potential translations from Big Code, using a generalizable program representation, an appropriate index structure, and a hierarchical filtering mechanism. Our approach does not require training data but can compete with existing translation models.

**1 INTRODUCTION**

Nowadays, numerous programs are being developed that require translations in other programming languages (PLs) to be further studied, reproduced, or applied on heterogeneous platforms. When the developers do not make the program translation efforts themselves, users have to manually rewrite the software in the needed PL, which is a time-consuming and error-prone process. Since traditional methods based on rule-based compilers or cross-language interpreters are hard-wired and require heavy human intervention for adaptation, the data-driven techniques are getting more traction. Reuse of code from existing "Big Code" repositories, such as GitHub and Bitbucket, has the potential to support many programming tasks including program translation.

Existing data-driven techniques for program translation are based on statistical models, such as 1pSMT [4], mppSMT [5] and Tree2tree [1], which train a program translation model. These approaches usually require a parallel dataset, in which programs in different PLs are semantically aligned via manual efforts, to supervised learn the translation model. To avoid generating parallel datasets, recent work leverages a transfer learning approach from NLP [6]. They first train a model that denoises a randomly corrupted program and use it as a pre-trained program translation model, which is then optimized by back-translation method. However, by only relying on NLP features their approach neglects the special features of PLs. Furthermore, programs translated through the aforementioned approaches suffer from grammar mistakes because they are machine-generated programs. Therefore, they usually require additional human-supervision. Also, they are often confined to a few PLs because it is not trivial to extract general features that apply to every PL. Another promising direction is to retrieve similar code in target language directly from Big Code as potential translations.

However, existing retrieval systems lack the proper capabilities for cross-language code retrieval [2, 3]. And instead of raw program input, they rely on queries consisting of semantically expressive keywords, descriptions, or user specifications.

In this paper, we propose Rpt, a novel program translation retrieval system. Given a raw program in a given source PL and a target PL, Rpt efficiently retrieves similar programs as potential translations from Big Code, using a generalizable program representation, an appropriate index structure, and a hierarchical filtering mechanism. Our approach does not require training data but can compete with existing translation models.

**2 OUR APPROACH**

We first discuss the necessary program representation and then the employed retrieval process of Rpt.

**2.1 Program Representation**

To identify cross-language code similarity, we need a unified representation that can be efficiently extracted from any given piece of code. Different from existing methods, Rpt considers both structural and textual features and their dependencies. Structural features can be captured by either a comprehensive concrete syntax tree (CST) or an abstract syntax tree (AST). The CST retains all the details, making it complicated and verbose with redundant information. The AST has more abstract but less informative syntax. And the abstraction strongly differs for different PLs. Thus, we fall back on the low-level CST as a basis and take the philosophy of AST to construct a unified abstract representation. As shown in Figure 1, Rpt first employs a left-to-right parser to parse the source code and generate the original CST, which contains all the nodes and branches of the program structure. Then it is pruned to a simplified syntax tree to reduce the computation complexity. The simplified tree of a JavaScript code example is shown in Figure 2.

![Figure 1: Rpt overview](image-url)
we divide the frequency into multiple buckets and use the bucket index structure further. RPT makes our approach scalable on big code, we implement a hierarchical filtering mechanism and a novel index structure to determine the final candidate. For the source program and each candidate, RPT calculates the weighted sum of both similarities.

3 EXPERIMENTS

Experiments. We apply our approach on a Java to C# parallel dataset used in previous work [1, 4, 5]. We compare the results of effectiveness and efficiency of RPT with state-of-the-art baselines 1pSMT, mppSMT, TransCoder, and TransCoder. Our metric is program accuracy [1]: the percentage of the retrieved translations that are functionality the same as the ground truth in the dataset. The results in Table 1 show that RPT is competitive to all baselines despite the fact that RPT is fully unsupervised and does not reuse existing data without training any models. Moreover, we observe that for the failed cases the translations tend to appear in the retrieved top 10 list. Further, the efficiency of our retrieval based system is shown to be comparable to other baselines. We also compare our index Pbi to a simple path type index. Pbi leads to a runtime improvement by two orders of magnitude at a scale of 3.8GB database.

4 CONCLUSION AND FUTURE WORK

We proposed RPT, a code-retrieval approach to support program translation with Big Code, which is competitive with existing translation methods. We published our code on https://github.com/BigDaMa/RPT.

In future work, will augment the retrieval system with program generation to overcome the limitations of the database.

ACKNOWLEDGMENT

This work was funded by the German Ministry for Education and Research as BIFOLD - Berlin Institute for the Foundations of Learning and Data (ref. 01IS18025A and ref. 01IS18037A).

REFERENCES

[1] Xinyun Chen, Chang Liu, and Dawn Song. 2018. Tree-to-tree neural networks for program translation. In NeurIPS 2547–2557.
[2] Xiaodong Gu, Hongyu Zhang, and Sunghun Kim. 2018. Deep code search. In JCSS 933–944.
[3] Fei Lv, Hongyu Zhang, Jian-guang Lou, Shaowei Wang, Dongmei Zhang, and Jianjun Zhao. 2015. Codetool: Effective code search based on api understanding and extended boolean model. In ASE.
[4] Anh Tuan Nguyen, Tung Thanh Nguyen, and Tien N Nguyen. 2013. Lexical statistical machine translation for language migration. In ESEC/FSE 651–654.
[5] Anh Tuan Nguyen, Tung Thanh Nguyen, and Tien N Nguyen. 2015. Divide-and-conquer approach for multi-phase statistical migration for source code (t). In ASE 585–596.
[6] Baptiste Bousier, Marie-Anne Lachaux, Lwik Chassatte, and Guillaume Lample. 2020. Unsupervised Translation of Programming Languages. In NeurIPS.