CodeXGLUE: A Machine Learning Benchmark Dataset for Code Understanding and Generation

Shuai Lu[*] Peking University
Daya Guo[*] Sun Yat-sen University
Shuo Ren[*] Beihang University
Junjie Huang[*] Beihang University

Alexey Svyatkovskiy Microsoft
Ambrosio Blanco Microsoft Research Asia
Colin Clement Microsoft
Dawn Drain Microsoft

Daxin Jiang Microsoft
Duyu Tang Microsoft Research Asia
Ge Li Peking University
Lidong Zhou Microsoft Research Asia

Linjun Shou Microsoft
Long Zhou Microsoft Research Asia
Michele Tufano Microsoft
Ming Gong Microsoft

Ming Zhou Microsoft Research Asia
Nan Duan Microsoft Research Asia
Neel Sundaresan Microsoft
Shao Kun Deng Microsoft

Shengyu Fu Microsoft
Shujie Liu Microsoft Research Asia

A Filtering Rules

Code search and code summarization We use the **CodeSearchNet** dataset [2] for code search and code summarization tasks. We observe that some documents contain content not directly related to the function, such as a link "http://..." that refers to external resources and an HTML image tag "<img ...>" that inserts an image. To improve the quality of the dataset, we filter it by removing the following examples.

(1) Examples whose code could not be parsed into abstract syntax tree.

(2) Examples whose document tokens number is shorter than 3 or larger than 256.

(3) Examples whose document contains special tokens such as "http://" and "<img ...>".

(4) Examples whose document is empty or not written in English.

Documentation translation To improve the data quality, we filter the corpus by removing the following examples.

(1) Pairs whose source sentence is the same as the target sentence;

(2) Pairs whose length of source language or target language is less than three words;

(3) Pairs whose length ratio between source and target languages is larger than three;

(4) Pairs whose word alignment ratio computed by $\mathrm{fast\_align}$[2] is less than 0.6.

[*] indicates equal contribution and internship at Microsoft. Authors are listed in alphabetical order. Corresponding author is Nan Duan.

[2] https://github.com/clab/fast_align.

35th Conference on Neural Information Processing Systems (NeurIPS 2021) Track on Datasets and Benchmarks.

Table 1: Training and inference time costs for each task, evaluated on two P100 GPUs.

Task	Dataset Name	Training Cost	Inference Cost
Clone Detection	BigCloneBench	3 hours	2 hours
	POJ-104	2 hours	10 minutes
Defect Detection	Devign	1 hours	5 minutes
Cloze Test	CT-all	N/A	30 minutes
	CT-max/min	N/A	5 minutes
Code Completion	PY150	25 hours	30 minutes
	Github Java Corpus	2 hours	10 minutes
Code Repair	Bugs2Fix	24 hours	20 minutes
Code Translation	CodeTrans	20 hours	15 minutes
NL Code Search	CodeSearchNet, AdvTest	5 hours	10 minutes
	CodeSearchNet, WebQueryTest	5 hours	5 minutes
Text-to-Code Generation	CONCODE	30 hours	20 minutes
Code Summarization	CodeSearchNet	On average, 12 hours for each PL	On average, 1 hours for each PL
Documentation Translation	Microsoft Docs	30 hours	1 hour

B Collection of WebQueryTest

The creation process of WebQueryTest can be divided into two stages: data collection and annotation. We first collect real user queries from the web query logs of a commercial search engine and we keep the queries with "python". Inspired by Yan et al. [3], we design some heuristics based on keyword exact matching to filter out queries without the code search intent. Then we select candidate codes for each query from the Python validation and testing sets in CodeSearchNet. To shrink the candidates to be annotated for each query, we select the top two functions with the highest query-code similarity computed by a CodeBERT-based code retrieval model, which is trained on 148K automated-minded Python Stack Overflow Question-Code (StaQC) [4] with the default parameters provided by Feng et al. [1].

We use a two-stage annotation schema to label each instance. The first step is to judge whether the query has a code-search intent. Instances labeled as "-1" are those without code search intent. The second step is to assess whether the code (with its documentation) can answer the query. Instances labeled as "1" are those where the code can answer the query. Otherwise, they are labeled as "0". Finally, the numbers of instances with labels -1, 0 and 1 are 254, 624 and 422, respectively. Since we are more interested in query-code matching, we include only the categories 0 and 1 in our final test set.

C Training Cost

Table 1 shows how long it takes to train the model and to do inference on the model in all tasks on two NVIDIA Tesla P100 GPUs.

References

[1] Zhangyin Feng, Daya Guo, Duyu Tang, Nan Duan, Xiaocheng Feng, Ming Gong, Linjun Shou, Bing Qin, Ting Liu, Daxin Jiang, et al. Codebert: A pre-trained model for programming and natural languages. *arXiv preprint arXiv:2002.08155*, 2020.

[2] Hamel Husain, Ho-Hsiang Wu, Tiferet Gazit, Miltiadis Allamanis, and Marc Brockschmidt. Codesearchnet challenge: Evaluating the state of semantic code search. *arXiv preprint arXiv:1909.09436*, 2019.

[3] S. Yan, H. Yu, Y. Chen, B. Shen, and L. Jiang. Are the code snippets what we are searching for? a benchmark and an empirical study on code search with natural-language queries. In *2020 IEEE 27th International Conference on Software Analysis, Evolution and Reengineering (SANER)*, pages 344–354, 2020. doi: 10.1109/SANER48275.2020.9054840.

[4] Ziyu Yao, Daniel S Weld, Wei-Peng Chen, and Huan Sun. Staqc: A systematically mined question-code dataset from stack overflow. In *Proceedings of the 2018 World Wide Web Conference*, pages 1693–1703, 2018.