Advances in Reliability, Safety and Security

ESREL 2024
Collection of Extended Abstracts

Evaluating Code Generation Capabilities Of Large Language Models To Solve Risk, Reliability And Resilience Related Problems

Jean Meunier-Pion[a], Zhiguo Zeng[b], Anne Barros[b]

[a]*Laboratoire Génie Industriel, CentraleSupélec, Université Paris-Saclay*
[b]*Chair of Risk and Resilience of Complex Systems, Laboratoire Génie Industriel, CentraleSupélec, Université Paris-Saclay*

Keywords: Large language model, performance evaluation, code generation, reliability

Recent advancements in Large Language Models (LLMs), exemplified by GPT-3.5, have demonstrated remarkable capabilities in understanding natural language and aiding humans in solving complex problems. One particularly successful application of LLMs is in the generation of code based on natural language descriptions of functionality. For instance, CodeBERT (Feng et al., 2020) achieved notable success by training on docstrings paired with functions, showcasing strong results in code search. PyMT5 (Clement et al., 2020) utilized the T5 objective to create a system capable of translating non-overlapping subsets of function signatures, docstrings, and code bodies.

In the domain of risk, resilience, and reliability, numerous tasks necessitate the implementation of computer code. Common examples include analyzing the reliability of complex systems through Monte Carlo or discrete event simulation and developing numerical methods to estimate parameters in statistical models for lifetime or degradation testing. Efficient application of code-generation tools to these tasks could significantly enhance the productivity of the risk, reliability, and resilience community. However, existing code generation models, as previously reviewed, are primarily tailored for generic software engineering applications and lack alignment with the specific vertical application of risk, reliability, and resilience. As a result, their performance within this domain is not assured and requires evaluation before widespread implementation.

Various benchmark datasets have been proposed for evaluating the code-generation capabilities of AI models. Barone & Sennrich (2017) introduced a comprehensive dataset consisting of Python declarations, docstrings, and code bodies scraped from GitHub. The CodeSearchNet challenge (Husain et al., 2019) expanded upon this by compiling a larger corpus from GitHub, encompassing data from multiple popular programming languages. Notably, CodeXGLUE (Lu et al., 2021) aggregated diverse programming benchmarks, introducing the CodeBLEU metric (Ren et al., 2020). Two benchmark datasets for the synthesis of basic Python programs were introduced in (Austin et al., 2021): MBPP (Mostly Basic Programming Problems) consisting of problems that can be solved by entry-level programmers, and MathQA-Python, consisting of problems that require the generation of code from complex text. More recently, a novel benchmark, ClassEval, was introduced in (Du et al., 2023) for benchmarking LLMs on class-level code generation, which involves interdependencies between several code units. While relevant, these benchmarks primarily focus on generic programming tasks and lack specificity to risk, reliability, and resilience applications.

In this paper, we address this gap by introducing a novel benchmark evaluation dataset, HumanEval-R3, designed to assess the capabilities of LLMs in generating code for risk, reliability, and resilience applications. Following the format of the HumanEval dataset (Chen et al., 2021), HumanEval-R3 features original, hand-written programming problems specific to this application domain. Evaluation of LLM performance is conducted through functional correctness assessment, utilizing unit tests. A representative problem from the generated

HumanEval-R3 dataset is presented in Figure 1. The dataset will be publicly accessible on GitHub following the conclusion of the conference, fostering collaboration and further research in this specialized domain.

```
def cal_series_sys_rel(comp_rel):
    '''
    You will be given a numpy array comp_rel. Your task is to
calculate the system reliability of a series system, where the
reliability of the components in the system is given in comp_rel.

Examples:
>>> cal_series_sys_rel(np.array([.9])
.9
>>> cal_series_sys_rel(np.array([.9, .8])
.72
    '''
```

Fig. 1. A sample problem from the HumanEval-R3 dataset.

Acknowledgements

The authors appreciate the financial support from Chair of Risk and Resilience of Complex System (RRSC, Chair EDF, Orange and SNCF). The research of Zhiguo Zeng is partially funded by ANR under grant number ANR-22-CE10-0004.

References

Austin, J., Odena, A., Nye, M., Bosma, M., Michalewski, H., Dohan, D., Jiang, E., Cai, C., Terry, M., Le, Q. and Sutton, C., 2021. Program synthesis with large language models. arXiv preprint arXiv:2108.07732.

Barone, A.V.M. and Sennrich, R., 2017. A parallel corpus of python functions and documentation strings for automated code documentation and code generation. arXiv preprint arXiv:1707.02275.

Chen, M., Tworek, J., Jun, H., Yuan, Q., Pinto, H.P.D.O., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., Brockman, G. and Ray, A., 2021. Evaluating large language models trained on code. arXiv preprint arXiv:2107.03374.

Clement, C.B., Drain, D., Timcheck, J., Svyatkovskiy, A. and Sundaresan, N., 2020. PyMT5: multi-mode translation of natural language and Python code with transformers. arXiv preprint arXiv:2010.03150.

Du, X., Liu, M., Wang, H., Liu, J., Chen, Y., Feng, J., Sha, C., Peng, X. and Lou, Y., 2023. ClassEval: A manually-crafted benchmark for evaluating llms, on class-level code generation. arXiv preprint arXiv:2308.01861.

Feng, Z., Guo, D., Tang, D., Duan, N., Feng, X., Gong, M., Shou, L., Qin, B., Liu, T., Jiang, D. and Zhou, M., 2020. Codebert: A pre-trained model for programming and natural languages. arXiv preprint arXiv:2002.08155.

Husain, H., Wu, H.H., Gazit, T., Allamanis, M. and Brockschmidt, M., 2019. Codesearchnet challenge: Evaluating the state of semantic code search. arXiv preprint arXiv:1909.09436.

Lu, S., Guo, D., Ren, S., Huang, J., Svyatkovskiy, A., Blanco, A., Clement, C., Drain, D., Jiang, D., Tang, D. and Li, G., 2021. Codexglue: A machine learning benchmark dataset for code understanding and generation. arXiv preprint arXiv:2102.04664.

Ren, S., Guo, D., Lu, S., Zhou, L., Liu, S., Tang, D., Sundaresan, N., Zhou, M., Blanco, A. and Ma, S., 2020. Codebleu: a method for automatic evaluation of code synthesis. arXiv preprint arXiv:2009.10297.