[12] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards,

Yuri Burda, Nicholas Joseph, Greg Brockman, et al. 2021. Evaluating Large Language Models Trained on Code. arXiv

preprint arXiv:2107.03374 (2021).

[68] Frank F Xu, Uri Alon, Graham Neubig, and Vincent Josua Hellendoorn. 2022. A Systematic Evaluation of Large

Language Models of Code. In ACM SIGPLAN International Symposium on Machine Programming (MAPS). 1–10. https:

//doi.org/10.1145/3520312.3534862

[50] OpenAI. 2023. GPT-4 Technical Report. ArXiv abs/2303.08774 (2023).

[71] Zibin Zheng, Kaiwen Ning, Jiachi Chen, Yanlin Wang, Wenqing Chen, Lianghong Guo, and Weicheng Wang. 2023.

Towards an Understanding of Large Language Models in Software Engineering Tasks. arXiv preprint arXiv:2308.11396

(2023).

[31] Xinyi Hou, Yanjie Zhao, Yue Liu, Zhou Yang, Kailong Wang, Li Li, Xiapu Luo, David Lo, John Grundy, and Haoyu

Wang. 2023. Large Language Models for Software Engineering: A Systematic Literature Review. arXiv preprint

arXiv:2308.10620 (2023).

[65] Moshi Wei, Nima Shiri Harzevili, Yuchao Huang, Junjie Wang, and Song Wang. 2022. Clear: Contrastive Learning

for API Recommendation. In IEEE/ACM International Conference on Software Engineering (ICSE). 376–387. https:

//doi.org/10.1145/3510003.3510159

[18] Sakina Fatima, Taher A Ghaleb, and Lionel Briand. 2022. Flakify: A Black-box, Language Model-based Predictor for

Flaky Tests. IEEE Transactions on Software Engineering (2022). https://doi.org/10.1109/TSE.2022.3201209

[70] Jialu Zhang, Todd Mytkowicz, Mike Kaufman, Ruzica Piskac, and Shuvendu K Lahiri. 2022. Using Pre-trained Language

Models to Resolve Textual and Semantic Merge Conicts (Experience Paper). In ACM International Symposium on

Software Testing and Analysis (ISSTA). 77–88. https://doi.org/10.1145/3533767.3534396

[48] Shantanu Mandal, Adhrik Chethan, Vahid Janfaza, SM Mahmud, Todd A Anderson, Javier Turek, Jesmin Jahan Tithi,

and Abdullah Muzahid. 2023. Large Language Models Based Automatic Synthesis of Software Specications. arXiv

preprint arXiv:2304.09181 (2023).

[30] Tobias Hey, Jan Keim, Anne Koziolek, and Walter F Tichy. 2020. Norbert: Transfer Learning for Requirements

Classication. In IEEE International Requirements Engineering Conference (RE). IEEE, 169–179. https://doi.org/10.1109/

RE48521.2020.00028

[1] 2023. ChatGPT Plugins. Retrieved September 25, 2023 from https://openai.com/blog/chatgpt-plugins#code-interpreter.