Daftar Pustaka

[1] H. Zhong and Z. Su, "An empirical study on real bug fixes," *Proceedings - International Conference on Software Engineering*, vol. 1, pp. 913–923, 2015, doi: 10.1109/ICSE.2015.101.

[2] T. Britton, L. Jeng, G. Carver, P. Cheak, and T. Katzenellenbogen, "Reversible debugging software," *Judge Bus. School, Univ. Cambridge, Cambridge, UK, Tech. Rep*, vol. 229, 2013.

[3] K. Huang *et al.*, "A Survey on Automated Program Repair Techniques," *ACM Comput Surv*, vol. 37, no. 4, 2023, [Online]. Available: http://arxiv.org/abs/2303.18184

[4] J. Finnie-Ansley, P. Denny, B. A. Becker, A. Luxton-Reilly, and J. Prather, "The robots are coming: Exploring the implications of OpenAI codex on introductory programming," *ACM International Conference Proceeding Series*, pp. 10–19, 2022, doi: 10.1145/3511861.3511863.

[5] J. Finnie-Ansley, P. Denny, A. Luxton-Reilly, E. A. Santos, J. Prather, and B. A. Becker, "My AI Wants to Know if This Will Be on the Exam: Testing OpenAI's Codex on CS2 Programming Exercises," *ACM International Conference Proceeding Series*, pp. 97–104, 2023, doi: 10.1145/3576123.3576134.

[6] J. A. Prenner, H. Babii, and R. Robbes, "Can OpenAI's Codex Fix Bugs?: An evaluation on QuixBugs," *Proceedings - International Workshop on Automated Program Repair, APR 2022*, pp. 69–75, 2022, doi: 10.1145/3524459.3527351.

[7] J. White *et al.*, "A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT," 2023, [Online]. Available: http://arxiv.org/abs/2302.11382

[8] J. Wei *et al.*, "Chain-of-Thought Prompting Elicits Reasoning in Large Language Models," *Adv Neural Inf Process Syst*, vol. 35, no. NeurIPS, pp. 1–14, 2022.

[9] A. Lewkowycz *et al.*, "Solving Quantitative Reasoning Problems with Language Models," *Adv Neural Inf Process Syst*, vol. 35, no. NeurIPS, 2022.

[10] D. Lin, A. Chen, J. Koppel, and A. Solar-Lezama, "QuixBugs: A Multi-Lingual Program Repair Benchmark Set Based on the Quixey Challenge," *SPLASH Companion 2017 - Proceedings Companion of the 2017 ACM SIGPLAN International Conference on Systems, Programming, Languages, and Applications: Software for Humanity*, no. February, pp. 55–56, 2017, doi: 10.1145/3135932.3135941.

[11] Q. Zhang, C. Fang, Y. Ma, W. Sun, and Z. Chen, "A Survey of Learning-based Automated Program Repair," *ACM Transactions on Software Engineering and Methodology*, vol. 33, no. 2, 2023, doi: 10.1145/3631974.

[12] C. Le Goues, M. Pradel, and A. Roychoudhury, "Automated program repair," *Commun ACM*, vol. 62, no. 12, pp. 56–65, 2019, doi: 10.1145/3318162.

[13] K. Liu *et al.*, "A critical review on the evaluation of automated program repair systems," *Journal of Systems and Software*, vol. 171, 2021, doi: 10.1016/j.jss.2020.110817.

[14] C. Le Goues, T. V. Nguyen, S. Forrest, and W. Weimer, "GenProg: A generic method for automatic software repair," *IEEE Transactions on Software Engineering*, vol. 38, no. 1, pp. 54–72, 2012, doi: 10.1109/TSE.2011.104.

[15] H. D. T. Nguyen, D. Qi, A. Roychoudhury, and S. Chandra, "SemFix: Program repair via semantic analysis," *Proceedings - International Conference on Software Engineering*, pp. 772–781, 2013, doi: 10.1109/ICSE.2013.6606623.

[16] K. Liu, A. Koyuncu, D. Kim, and T. F. Bissyandé, "TBAR: Revisiting template-based automated program repair," *ISSTA 2019 - Proceedings of the 28th ACM SIGSOFT International Symposium on Software Testing and Analysis*, pp. 43–54, 2019, doi: 10.1145/3293882.3330577.

[17] T. Lutellier, H. V. Pham, L. Pang, Y. Li, M. Wei, and L. Tan, "CoCoNuT: Combining context-aware neural translation models using ensemble for program repair," *ISSTA 2020 - Proceedings of the 29th ACM SIGSOFT International Symposium on Software Testing and Analysis*, pp. 101–114, 2020, doi: 10.1145/3395363.3397369.

[18] N. Jiang, T. Lutellier, and L. Tan, "CURE: Code-aware neural machine translation for automatic program repair," *Proceedings - International Conference on Software Engineering*, no. x, pp. 1161–1173, 2021, doi: 10.1109/ICSE43902.2021.00107.

[19] C. S. Xia, Y. Wei, and L. Zhang, "Automated Program Repair in the Era of Large," 2022, [Online]. Available: http://arxiv.org/abs/2210.14179

[20] Q. Zhang *et al.*, "A Systematic Literature Review on Large Language Models for Automated Program Repair," 2024, [Online]. Available: http://arxiv.org/abs/2405.01466

[21] S. B. Hossain *et al.*, "A Deep Dive into Large Language Models for Automated Bug Localization and Repair," *Proceedings of the ACM on Software Engineering*, vol. 1, no. FSE, pp. 1471–1493, 2024, doi: 10.1145/3660773.

[22] R. McCauley *et al.*, “Debugging: A review of the literature from an educational perspective,” *Computer Science Education*, vol. 18, no. 2, pp. 67–92, 2008, doi: 10.1080/08993400802114581.
[23] S. Abbaspour Asadollah, D. Sundmark, S. Eldh, H. Hansson, and W. Afzal, “10 Years of research on debugging concurrent and multicore software: a systematic mapping study,” *Software Quality Journal*, vol. 25, no. 1, pp. 49–82, 2017, doi: 10.1007/s11219-015-9301-7.
[24] E. Kasneci *et al.*, “ChatGPT for good? On opportunities and challenges of large language models for education,” *Learn Individ Differ*, vol. 103, no. February, 2023, doi: 10.1016/j.lindif.2023.102274.
[25] A. James Thirunavukarasu *et al.*, “Large language models in medicine,” *Nat Med*, vol. 29, no. 8, pp. 1–46, 2023.
[26] J. Liu, C. S. Xia, Y. Wang, and L. Zhang, “Is Your Code Generated by ChatGPT Really Correct? Rigorous Evaluation of Large Language Models for Code Generation,” no. NeurIPS, 2023, [Online]. Available: http://arxiv.org/abs/2305.01210
[27] DeepSeek-AI *et al.*, “DeepSeek LLM: Scaling Open-Source Language Models with Longtermism,” 2024, [Online]. Available: http://arxiv.org/abs/2401.02954
[28] H. Touvron *et al.*, “LLaMA: Open and Efficient Foundation Language Models,” 2023, [Online]. Available: http://arxiv.org/abs/2302.13971
[29] M. Finlayson, X. Ren, and S. Swayamdipta, “Logits of API-Protected LLMs Leak Proprietary Information,” pp. 1–15, 2024, [Online]. Available: http://arxiv.org/abs/2403.09539
[30] C. S. Xia and L. Zhang, “Less training, more repairing please: revisiting automated program repair via zero-shot learning,” *ESEC/FSE 2022 - Proceedings of the 30th ACM Joint Meeting European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pp. 959–971, 2022, doi: 10.1145/3540250.3549101.
[31] J. Li♂, G. Li, Y. Li, and Z. Jin, “Structured Chain-of-Thought Prompting for Code Generation,” *ACM Transactions on Software Engineering and Methodology*, 2024, doi: 10.1145/3690635.
[32] G. G. Lee, E. Latif, X. Wu, N. Liu, and X. Zhai, “Applying large language models and chain-of-thought for automatic scoring,” *Computers and Education: Artificial Intelligence*, vol. 6, pp. 1–48, 2024, doi: 10.1016/j.caeai.2024.100213.
[33] J. Miao, C. Thongprayoon, S. Suppadungsuk, P. Krisanapan, Y. Radhakrishnan, and W. Cheungpasitporn, “Chain of Thought Utilization in Large Language Models and Application in Nephrology,” *Medicina (Lithuania)*, vol. 60, no. 1, 2024, doi: 10.3390/medicina60010148.
[34] W. Yuan *et al.*, “CIRCLE: Continual repair across programming languages,” *ISSTA 2022 - Proceedings of the 31st ACM SIGSOFT International Symposium on Software Testing and Analysis*, no. Dl, pp. 678–690, 2022, doi: 10.1145/3533767.3534219.
[35] Q. Zhu *et al.*, “A syntax-guided edit decoder for neural program repair,” *ESEC/FSE 2021 - Proceedings of the 29th ACM Joint Meeting European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pp. 341–353, 2021, doi: 10.1145/3468264.3468544.
[36] B. Yang and J. Yang, “Exploring the Differences between Plausible and Correct Patches at Fine-Grained Level,” *IBF 2020 - Proceedings of the 2020 IEEE 2nd International Workshop on Intelligent Bug Fixing*, pp. 1–8, 2020, doi: 10.1109/IBF50092.2020.9034821.
[37] C. S. Xia and L. Zhang, “Keep the Conversation Going: Fixing 162 out of 337 bugs for $0.42 each using ChatGPT,” 2023, [Online]. Available: http://arxiv.org/abs/2304.00385
[38] S. Shekhar, T. Dubey, K. Mukherjee, A. Saxena, A. Tyagi, and N. Kotla, “Towards Optimizing the Costs of LLM Usage,” 2024, [Online]. Available: http://arxiv.org/abs/2402.01742
[39] TIOBE, “TIOBE Index.” Accessed: Jan. 15, 2024. [Online]. Available: https://www.tiobe.com/tiobe-index/
[40] D. Sobania, M. Briesch, C. Hanna, and J. Petke, “An Analysis of the Automatic Bug Fixing Performance of ChatGPT,” *Proceedings - 2023 IEEE/ACM International Workshop on Automated Program Repair, APR 2023*, pp. 23–30, 2023, doi: 10.1109/APR59189.2023.00012.
[41] Artificial Analysis, “Artificial Analysis AI Review.” Accessed: Dec. 09, 2024. [Online]. Available: https://artificialanalysis.ai/
[42] C. Arora, A. I. Sayeed, S. Licorish, F. Wang, and C. Treude, “Optimizing Large Language Model Hyperparameters for Code Generation,” pp. 1–10, 2024, [Online]. Available: http://arxiv.org/abs/2408.10577
[43] C. S. Xia, Y. Wei, and L. Zhang, “Automated Program Repair in the Era of Large Pre-trained Language Models,” *Proceedings - International Conference on Software Engineering*, pp. 1482–1494, 2023, doi: 10.1109/ICSE48619.2023.00129.
[44] T. Kojima, M. Reid, and S. S. Gu, “Large Language Models are Zero-Shot Reasoners,” no. NeurIPS, 2022.

[45] Anthropic, “Crafting effective examples.” Accessed: Oct. 31, 2024. [Online]. Available: https://docs.anthropic.com/
[46] B. Okken, “Python Testing with Pytest,” *J Chem Inf Model*, vol. 53, no. 9, pp. 1689–1699, 2013.
[47] D. T. Kurniasari and S. Rochimah, “An Evaluation Model of Website Testing Framework Based on ISO 25010 Performance Efficiency,” vol. 37, no. 2, pp. 1130–1139, 2025, doi: 10.11591/ijeecs.v37.i2.pp1130-1139.
[48] OpenAI, “Learning to reason with LLMs.” Accessed: Jan. 15, 2025. [Online]. Available: https://openai.com/index/learning-to-reason-with-llms/
[49] Z. Yu, L. He, Z. Wu, X. Dai, and J. Chen, “Towards Better Chain-of-Thought Prompting Strategies: A Survey,” 2023, [Online]. Available: http://arxiv.org/abs/2310.04959
[50] DeepSeek-AI, “DeepSeek-V3 Technical Report,” vol. 2024, pp. 1–53, 2024.
[51] H. Ye, M. Martinez, T. Durieux, and M. Monperrus, “A comprehensive study of automatic program repair on the QuixBugs benchmark,” *Journal of Systems and Software*, vol. 171, 2021, doi: 10.1016/j.jss.2020.110825.