Emergent Possibilities and Challenges in Deep Learning for Code

1. Workshop Summary

The application of deep learning in code, spanning a diverse set of challenging tasks such as code completion, repair, synthesis, and explanation, constitutes a significant area of research within machine learning. The last two years have witnessed remarkable progress in this domain with the development of recent techniques (Rozière et al., 2023; Li et al., 2023; Guo et al., 2024; Wei et al., 2024; Yang et al., 2024a) that enables us to solve realistic and complex software development tasks. Moreover, code generation is increasingly being used as a critical step in solving complex tasks beyond code, such as reasoning (Yang et al., 2024b; Gao et al., 2023), sequential decision-making (Zhang et al., 2024; Wang et al., 2024a), robotic control (Liang et al., 2023) and machine learning for science and algorithmic discovery (Romera-Paredes et al., 2023; Mankowitz et al., 2023).

These advances underscore the increasingly pivotal role deep learning for code has come to play within the broader machine learning discipline, and have given rise to novel and increasingly complex challenges. The third Deep Learning for Code workshop (DL4C, `https://dl4c.github.io`) aims to provide a forum for researchers, practitioners, and industry professionals to share insights, collaborate, and advance the state-of-the-art in this raplidly evolving field. Building on the success of previous iterations (ICLR'22, ICLR'23), which showcased significant works through poster sessions, invited talks, and panel discussions from leading experts, the third DL4C workshop will focus on emergent possibilities and challenges in deep learning for code. The key focus areas of this year include agentic methods for programming tasks, post-training and alignment for code, human-computer/AI interaction and developer productivity for code, open science and responsible AI for code, and benchmarking for emergent tasks. By addressing these cutting-edge topics, we aim to solidify our position as a prominent platform for advancing research in the increasingly critical area of deep learning for code.

2. History and Point of Difference

The DL4C workshop debuted at ICLR'22, and was held for the second time at ICLR'23. Both were massive successes: despite covid, at the past installation, we had more than 300 attendees, 45 program committee members, 6 invited talks from a diverse set of speakers, and 19 out of 37 submissions accepted (excluding papers accepted elsewhere but presented at DL4C). The workshop also received significant attention on social media, with more than 450 followers on the workshop's social account.

Since the last iteration of the workshop, significant advances have been made in the field, including increasingly powerful code models, which have both unlocked new application areas such as multi-step, agentic code assistants for various tasks, and also introduced new challenges such as developing both benchmark datasets and human-AI workflows for such tasks. In light of these developments, the 3rd DL4C workshop will focus specifically on these *emergent possibilities and challenges*. This presents an exceptional opportunity to unite researchers and practitioners who have been investing in code generation from diverse disciplines, spanning academia and industry and representing economically and geographically varied backgrounds. With the community's rapid growth, we estimate DL4C at ICLR'25 would attract 400–500 attendees, and we have secured 70+ confirmed reviewers. We detail our hosting plan in the upcoming sections.

3. Topics

We welcome all relevant submissions to DL4C, and this year, we focus especially on emerging advances and challenges in our field, such as:

Agentic Methods for Programming Tasks Agents emerge to solve realistic coding tasks, such as solving GitHub issues (Jimenez et al., 2024) or software developing tasks (Li et al., 2024a). Several recent works (Yang et al., 2024a; Xia et al., 2024) showed that action-taking agents can achieve 3–4x higher scores than vanilla retrieval-augmented code generation in a more autonomous way (Wang et al., 2024b). This sparked substantial attention among researchers and

practitioners, with tens of new agents appearing in the past few months. We believe this is just the beginning of a long journey and welcome works with innovative coding agents.

Post-training and Alignment for Code Current code models are constrained by their capacity to learn from feedback for refinement. Recent work like Zheng et al. (2024) has developed methods to learn from execution feedback and iterative refine generated code. Further, Weyssow et al. (2024) has proposed a dataset for evaluating code model alignment with human preferences. DL4C encourages submissions in the direction of alignment for code, including but not limited to how to learn from human feedback, execution feedback, and AI feedback for better code generation.

Developer Productivity and HCI for Code With the increasing capability of code models, the adaptation of such models for end-user scenarios, such as feature development, troubleshooting, and writing unit test, is a critical challenge especially since recently (Peng et al., 2023; Ma et al., 2023; Jaworski & Piotrkowski, 2023; Liu & Li, 2024; Mozannar et al., 2024; Nguyen et al., 2024; Prather et al., 2024). DL4C welcomes submissions on developer productivity and human-AI interaction studies for code from all disciplines (Machine Learning, Human-Computer Interaction, and Software Engineering, etc.).

Open Science and Responsible AI for Code Openness is a key component of responsible AI. We welcome contributions from researchers who follow responsible AI practices and strive for openness and transparency in their work and who are willing to share their code, models, and data, e.g., (Li et al., 2023; Kocetkov et al., 2022; Wei et al., 2024; Lozhkov et al., 2024; Li et al., 2024c; Bhatt et al., 2024). We also welcome contributions from researchers interested in developing open science practices for deep learning for code.

Benchmarking and Evaluation for Code Code is often nuanced and complicated, making accurate benchmarking of code tasks challenging. There has been a surge in benchmarks for code that evaluate various attributes, such as evolving execution-based benchmark (Jain et al., 2024), code understanding (Gu et al., 2024), code efficiency (Huang et al., 2024; Liu et al., 2024b), and project-level context (Ding et al., 2024; Li et al., 2024b). Further, recent work has started to look into how we evaluate code generation using model-as-a-judge, focusing on accuracy, efficiency, security, and human preference (Liu et al., 2024a). We welcome submissions that build better and more realistic benchmarks and ultimately promote the development of the entire community.

Other topics of interest include but are not limited to: Reinforcement Learning for Code, Pre-training Methods and Representation for Code, From Natural Language To Code, Code Generation for Applications Beyond Code such as Reasoning, Decision Making, and Algorithmic Discovery, Formal Methods for Code, and various applications of code models such as Program Repair, Code Translation, Code Explanation, etc. Due to space constraints, the detailed explanation is omitted here and will be provided in the Call for Papers.

4. Speakers

We have secured six confirmed speakers from academia and industry with a diverse set of backgrounds, representing both early career and senior researchers, and a broad range of topics in DL4C. Below are their details (in order of last name):

- Michele Catasta (he) is the President at Replit. In the past, he was VP of AI at Replit and Head of Applied Research at Google Labs, working on AI applied to Source Code and Large Language Models. Before then, he was a Research Scientist and Instructor in AI at Stanford University, and received his Ph.D. in Computer Science from EPFL. His talk will be on Code AI Applications in Production Environments.
- Xinyun Chen (she) is a Senior Research Scientist at Google DeepMind. Her research interests are large language models, code generation, and AI security. She obtained my Ph.D. degree in Computer Science from UC Berkeley in 2022. Her talk will be on LLM for Code and Reasoning.
- Daniel Fried (he) is an Assistant Professor at the Language Technologies Institute in the School of Computer Science at Carnegie Mellon University. He is also a Research Scientist at Meta. His talk will be on Code and Natural Language Interaction.
- Stefania Druga (she) is a Principal Researcher at the University of Chicago. She received her Ph.D. in Creative AI Literacies at the University of Washington Information School. Her talk will be on HCI for AI Coding Assistants.

- Baptiste Rozière (he) is a Research Scientist at Meta AI in Paris working in the code generation team. He works on large language models, with a special interest in applications to code. Baptiste contributed to Llama and started Code Llama. Before that, he worked on model pre-training and machine translation for programming languages. His talk will be about new development of Code Llama.
- Tao Yu (he) is an Assistant Professor of Computer Science at The University of Hong Kong. He spent one year in the UW NLP Group working with Noah Smith, Luke Zettlemoyer, and Mari Ostendorf. He completed his Ph.D. in Computer Science from Yale University, advised by Dragomir Radev. His talk will be on agentic methods for code.

Considering the workshop is scheduled for 6+ months from the time of writing, we have reserved one final slot to accommodate emerging research in the field that may arise between the present time and the event date. Alternatively, we could host a speaker that we have contacted but haven't decided on attending ICLR in-person, e.g., Graham Neubig (CMU&lead of OpenHands) and Loubna Ben Allal (lead of BigCode).

Panel Discussions To complement the individual talks, we will host a panel discussion focusing on emergent possibilities and challenges in deep learning for code. This panel will feature diverse perspectives from both academia and industry, represented by junior and senior researchers. By leveraging the expertise of our invited speakers in this interactive format, we will encourage audience participation and facilitate a rich exchange of ideas. We anticipate that this discussion will provide valuable insights into the rapidly evolving field of deep learning for code, particularly benefiting students and professionals seeking to understand future directions and potential challenges. By bringing together varied viewpoints, we aim to foster a comprehensive and nuanced dialogue on the current state and future prospects of this dynamic research area.

5. Schedule

The workshop (tentative schedule in Table 1) will be about nine hours long, including a 1-hour lunch break, seven 30-minute invited talks with additional 10 minutes Q&A each, one spotlight session for outstanding work, one poster session, one panel discussion, and two rounds of coffee & social. We encourage participation from everyone in these sessions, including the panel. There are plenty of opportunities for discussions and socials throughout the day.

6. Remote Accommodations

While the workshop will be in person, in the case of unforeseen travel restrictions (e.g., denied visas) affecting participants, we will ensure their participation and accessibility. We plan to host 1) live-streaming (with Q/A) talks/discussions via Zoom, 2) virtual poster session via Gather/Slack, and 3) make both the videos and slides available on our website. We have expertise in all of these given past organizational experience, and we will work with ICLR Workshop Chairs if there is centralized support for these.

7. Contributed Program

We anticipate receiving approximately 100 paper submissions for the workshop. The review process will be double-blind, and we will ensure that for each paper, the final decisions are made without any conflict of interest. We will use OpenReview for paper reviews, and we have secured 70+ **confirmed** reviewers (see §10.2) to ensure sufficient review coverage with a reasonable reviewing workload. A tentative timeline of the review process can be found in Table 2.

The workshop will feature both regular and tiny paper tracks, welcoming submissions on ongoing or novel machine learning work. To accommodate varying levels of progress, from simple yet novel ideas to fully developed topics, we will not enforce strict paper length requirements. However, we recommend authors limit their submissions to 8-10 pages for regular papers and 2-4 pages for tiny papers. We welcome all kinds of papers: research papers, position papers, system demonstrations, and more. Both paper tracks will have the opportunity to be presented as poster or spotlight talk at the workshop. Reproducibility and accessibility of research resources (datasets, code) will be a major consideration in the evaluation process.

8. Outreach

DL4C will be promoted through its website, `https://dl4c.github.io`, via twitter/X, @DL4Code, and various relevant email lists and slack workspaces. We will update the website to reflect the DL4C 2024 speakers and program, with pro-

08:00 - 08:10	Introduction and Opening Remarks
08:10 - 09:30	Invited Talk 1 & 2
09:30 - 09:45	Coffee Break & Social
09:45 - 10:30	Spotlight Sessions
10:30 - 11:50	Invited Talk 3 & 4
11:50 - 12:50	Lunch & Social
12:50 - 14:50	Invited Talk 5 & 6 & 7
14:50 - 15:10	Coffee Break & Social
15:10 - 16:00	Panel Discussion
16:00 - 17:00	Poster Session
17:00 - 17:10	Closing Remarks

Table 1: Tentative schedule of the workshop. We will adjust it based on ICLR workshop chair's guidance, e.g., to follow conference's specific lunch hours and coffee breaks.

Dec 8	Release Call for Papers
Feb 3	Paper submission deadline
Mar 1	Due date for reviews
Mar 5	Accept/reject notifications are sent
Apr 5	Camera-ready

Table 2: Tentative review process timeline.

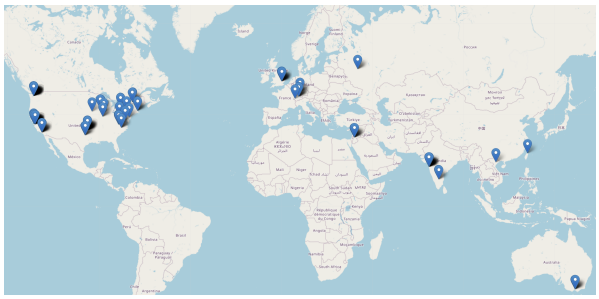


Figure 1: Author demographics at DL4C (2022 & 2023)

ceedings online for accessibility. The website is a part of our strategy to encourage participation in the DL4C workshop. We will also approach relevant contact points in academia and industry, and call for papers via various mailing lists and to past DL4C participants.

9. Diversity Commitment

We seek to organize a workshop with diverse organizers, speakers and attendees. Our organizers and speakers represent different races, genders, geographies, academic/industry affiliations, levels of seniority, and background. This diversity is also reflected in the submissions we elicit, with previous iterations of the workshop have featured work from 37 institutions across 10 countries, spanning a diverse range of socio-cultural backgrounds (Figure 1). We have also secured a large and diverse set of Program Committee Members (see §10.2 for details). We are committed to provide necessary supports for people that are in need, be it travel funds, remote accessibility, or other help needed.

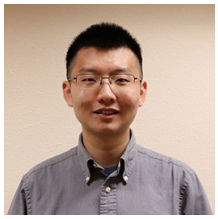
DL4C welcomes submissions not only in the Machine Learning community, but also relevant fields like Human-Computer Interaction, Software Engineering, Programming Language, and Natural Language Processing communities. We hope DL4C can be a venue to unite researchers of diverse disciplines working on all aspects of deep learning for code.

DL4C subscribes to the ICLR Code of Conduct. DL4C is committed to providing a harassment-free experience for everyone, regardless of gender, age, sexual orientation, disability, physical appearance, race, ethnicity and religion.

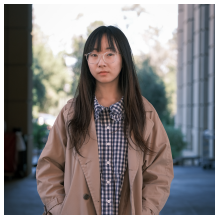
10. Organizers and Program Committee Members

10.1. Organizers

Our organizing team is composed of a mixture of academia and industry researchers with diverse demographics, background, seniority, expertise, and organizational experiences.



Zijian Wang (he/him; 🇨🇳 in 🇺🇸) is an Applied Science Manager at AWS AI Labs, in the Amazon Q Developer team. His current research focuses on building better generative models for code, especially on training, evaluating, and deploying these models at scale. In addition, he is interested in relevant large language model topics, for example, model compression, zero/few-shot learning, and robustness. Zijian co-organized the DL4C workshop at ICLR 2023, and the LLM4Code workshop at ICSE 2025 (top conference in Software Engineering).
https://zijianwang.me/ zijwan@amazon.com
@zijianwang30 Scholar



Ying Sheng (she/her; 🇨🇳 in 🇺🇸) will be an Assistant Professor at UCLA from 2026. She received her PhD from Stanford University. Her research is centered on building and deploying large language model applications, with an emphasis on accessibility, scalability, programmability, and verifiability. She is a core member of the LMSYS Org, where she has developed open models, datasets, systems, and evaluation tools (such as Vicuna, Chatbot Arena, SGLang, etc.).
https://sites.google.com/view/yingsheng ying1123@stanford.edu
@ying11231 Scholar

Giovanni Zappella (he/him; 🇮🇹 in 🇩🇪) is a Principal Applied Scientist at Amazon in Berlin (Germany). His research interests include code generation for complex real-world tasks. He is one of the technical leaders behind the Amazon Q Developer Agent. He previously worked on AutoML and recommendations problems focusing on methods for exploration/exploitation. Additionally, he is one of the organizers of the AutoML seminars.
https://giovannizappella.github.io/ zappella@amazon.de
@610v4nn1__ Scholar

Qian Liu (he/him; 🇨🇳 in 🇸🇬, **Local Organizer in Singapore**) is a Research Scientist at Sea AI Lab, Singapore. His research interests include code generation and natural language reasoning, including the work of TAPEX, LoraHub and Sailor. He was awarded as the KAUST AI Rising Star 2024 and nominated for the Baidu Scholarship 2020. He has made significant contributions to several collaborative projects, including StarCoder 1/2, OctoPack, and Lemur, all focusing on code generation. Additionally, he was also one of the co-founders of the MLNLP community, a well-known NLP community in China. Qian has served as the organizer for the 3rd Table Representation Learning workshop at NeurIPS 2024 and is also a co-organizer for the Open Science for Foundation Models workshop under review at ICLR.
https://siviltaram.github.io/ liuqian@sea.com
@sivil_taram Scholar

Devjeet Roy (he/him; 🇮🇳 in 🇺🇸) is a final-year PhD student at the Washington State University. His research revolves around understanding cognitive factors in software engineering activities using tools and techniques from neuroscience. Previously, worked at X and Google Labs, investigating ways in which we can improve human-machine collaboration in the context of deep learning for code, with a focus on uncertainty estimation techniques. Devjeet co-organized the DL4C workshop in 2023.
https://devjeetr.github.io/ devjeet.roy@wsu.edu
@devjeetrr Scholar

Gabriel Orlanski (he/him; 🇦🇷 in 🇺🇸) is a third-year PhD student in Computer Science student at The University of Wisconsin-Madison. His research interests are at the intersection of Machine Learning and Programming Languages, with a specific focus on self-supervised learning. He earned his Master's in Computer Science from New York University and his Bachelor's in Computer Science from Rensselaer Polytechnic Institute. He is an active member of the LatinX in AI community. Gabe co-organized the DL4C workshop in 2022 and 2023.
https://gabeorlanski.github.io gorlanski@cs.wisc.edu
@GOrlanski Scholar

Zora Zhiruo Wang (she/her; 🇨🇳 in 🇺🇸) is a second-year PhD student at Carnegie Mellon University. Her research is centered on building verifiable and autonomous agentic systems with language model writing programs. She previously worked at Microsoft Research as an assistant researcher, focusing on tabular data intelligence via domain-specific programs. She has actively contributed to community open-source projects such as StarCoder and evaluation harness in BigCode. Zora has organized the first agent workshop at CMU, and gave tutorials about LLM for Tabular Data at SIGIR.
https://zorazrw.github.io/ zhiruow@cs.cmu.edu
@ZhiruoW Scholar

Wen-Ding Li (he/him; 🇹🇼 in 🇺🇸) is a final-year PhD student at Cornell University. His research focuses on machine learning for program synthesis and on making them more trustworthy. Previously, he worked at Google, where he focused on code generation for the data science domain and the integration of LLM code generation into the Jupyter notebook-like environment. He has also helped with community open-source projects, including BigCode and Aya multilingual projects.
https://wending.dev wl678@cornell.edu
@xu3kev Scholar

10.2. Program Committee Members

We have successfully secured 70 confirmed Program Committee Members. We plan to invite additional PC members if our proposal is accepted. Our reviewers come from diverse backgrounds, as illustrated in Figure 2. With the 70 PC members already committed, we can ensure thorough review coverage, and we aim to keep workload light for each reviewer with no more than three reviews per person. The following is a list of the accepted PC members (in random order):

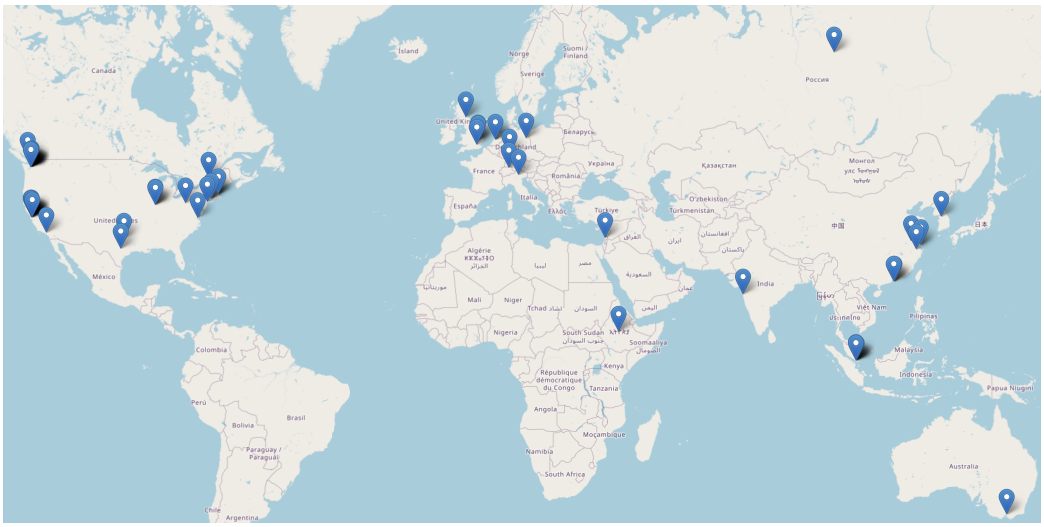


Figure 2: Program Committee Demographics

José Cambronero, Naman Jain, Qing Sun, Jacob Austin, Federico Federico, Youngkyoung Kim, Yuxiang Wei, Yifeng Ding, Jiawei Liu, Fan Zhou, Haochen Li, Jiayi Wei, Martin Weyssow, Kexun Zhang, Indraneil Paul, Gabriel Gordon-

Hall, Ziyang Luo, Nickil Maveli, Rauf Kurbanov, Benjamin Lipkin, Sanjay Krishna Gouda, Ansong Ni, Yanjun Shao, Sedrick Keh, Mingyue Shang, Denys Poshyvanyk, Pingchuan Ma, Zongjie Li, Alex Bezzubov, Manish Shetty, Virendra Mehta, Xiaoxuan Liu, Edward Wang, D. Q. Bui Nghi, Manasi Patwardhan, Egor Bogomolov, Eran Yahav, Jiyang Zhang, Ryan Carelli, Michael Feil, Michele Tufano, Shweta Garg, Noor Nashid, Zhensu SUN, Linghui Luo, Alexander Buchholz, Matthias Seeger, Sachit Kuhar, Jinjun Peng, Yu Yu, Laurent Callot, Yuhao Zhang, Madhav Singhal, Henok Ademtew, Patrick Haluptzok, Xin Zhou, Binyuan Hui, Kisub Kim, Jingxuan He, Xiangci Li, Wen-Hao Chiang, Tianbao Xie, Siheng Zhao, Jixuan Chen, Terry Yue Zhuo, Siddharth Goyal, Christian Bock.

References

Manish Bhatt, Sahana Chennabasappa, Yue Li, Cyrus Nikolaidis, Daniel Song, Shengye Wan, Faizan Ahmad, Cornelius Aschermann, Yaohui Chen, Dhaval Kapil, et al. Cyberseceval 2: A wide-ranging cybersecurity evaluation suite for large language models. *arXiv preprint arXiv:2404.13161*, 2024.

Yangruibo Ding, Zijian Wang, Wasi Ahmad, Hantian Ding, Ming Tan, Nihal Jain, Murali Krishna Ramanathan, Ramesh Nallapati, Parminder Bhatia, Dan Roth, et al. Crosscodeeval: A diverse and multilingual benchmark for cross-file code completion. *Advances in Neural Information Processing Systems*, 36, 2024.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. Pal: Program-aided language models. In *International Conference on Machine Learning*, pp. 10764–10799. PMLR, 2023.

Alex Gu, Baptiste Rozière, Hugh Leather, Armando Solar-Lezama, Gabriel Synnaeve, and Sida I Wang. Cruxeval: A benchmark for code reasoning, understanding and execution. *arXiv preprint arXiv:2401.03065*, 2024.

Daya Guo, Qihao Zhu, Dejian Yang, Zhenda Xie, Kai Dong, Wentao Zhang, Guanting Chen, Xiao Bi, Y Wu, YK Li, et al. Deepseek-coder: When the large language model meets programming–the rise of code intelligence. *arXiv preprint arXiv:2401.14196*, 2024.

Dong Huang, Jie M Zhang, Yuhao Qing, and Heming Cui. Effibench: Benchmarking the efficiency of automatically generated code. *arXiv preprint arXiv:2402.02037*, 2024.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. *arXiv preprint arXiv:2403.07974*, 2024.

Mateusz Jaworski and Dariusz Piotrkowski. Study of software developers' experience using the github copilot tool in the software development process. *arXiv preprint arXiv:2301.04991*, 2023.

Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik R Narasimhan. SWE-bench: Can language models resolve real-world github issues? In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=VTF8yNQM66.

Denis Kocetkov, Raymond Li, Loubna Ben Allal, Jia Li, Chenghao Mou, Carlos Muñoz Ferrandis, Yacine Jernite, Margaret Mitchell, Sean Hughes, Thomas Wolf, et al. The stack: 3 tb of permissively licensed source code. *arXiv preprint arXiv:2211.15533*, 2022.

Bowen Li, Wenhan Wu, Ziwei Tang, Lin Shi, John Yang, Jinyang Li, Shunyu Yao, Chen Qian, Binyuan Hui, Qicheng Zhang, et al. Devbench: A comprehensive benchmark for software development. *arXiv preprint arXiv:2403.08604*, 2024a.

Jia Li, Ge Li, Yunfei Zhao, Yongmin Li, Zhi Jin, Hao Zhu, Huanyu Liu, Kaibo Liu, Lecheng Wang, Zheng Fang, et al. Deveval: Evaluating code generation in practical software projects. *arXiv preprint arXiv:2401.06401*, 2024b.

Raymond Li, Loubna Ben Allal, Yangtian Zi, Niklas Muennighoff, Denis Kocetkov, Chenghao Mou, Marc Marone, Christopher Akiki, Jia Li, Jenny Chim, et al. Starcoder: may the source be with you! *arXiv preprint arXiv:2305.06161*, 2023.

Tianlin Li, Qian Liu, Tianyu Pang, Chao Du, Qing Guo, Yang Liu, and Min Lin. Purifying large language models by ensembling a small language model. *CoRR*, abs/2402.14845, 2024c. doi: 10.48550/ARXIV.2402.14845. URL https://doi.org/10.48550/arXiv.2402.14845.

Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as policies: Language model programs for embodied control. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 9493–9500. IEEE, 2023.

Jiangyue Liu and Siran Li. Toward artificial intelligence-human paired programming: A review of the educational applications and research on artificial intelligence code-generation tools. *Journal of Educational Computing Research*, pp. 07356331241240460, 2024.

Jiawei Liu, Thanh Nguyen, Mingyue Shang, Hantian Ding, Xiaopeng Li, Yu Yu, Varun Kumar, and Zijian Wang. Learning code preference via synthetic evolution. *arXiv preprint arXiv:2410.03837*, 2024a.

Jiawei Liu, Songrun Xie, Junhao Wang, Yuxiang Wei, Yifeng Ding, and Lingming Zhang. Evaluating language models for efficient code generation. *arXiv preprint arXiv:2408.06450*, 2024b.

Anton Lozhkov, Raymond Li, Loubna Ben Allal, Federico Cassano, Joel Lamy-Poirier, Nouamane Tazi, Ao Tang, Dmytro Pykhtar, Jiawei Liu, Yuxiang Wei, et al. Starcoder 2 and the stack v2: The next generation. *arXiv preprint arXiv:2402.19173*, 2024.

Wanlun Ma, Yiliao Song, Minhui Xue, Sheng Wen, and Yang Xiang. The" code"of ethics: A holistic audit of ai code generators. *arXiv preprint arXiv:2305.12747*, 2023.

Daniel J Mankowitz, Andrea Michi, Anton Zhernov, Marco Gelmi, Marco Selvi, Cosmin Paduraru, Edouard Leurent, Shariq Iqbal, Jean-Baptiste Lespiau, Alex Ahern, et al. Faster sorting algorithms discovered using deep reinforcement learning. *Nature*, 618(7964):257–263, 2023.

Hussein Mozannar, Gagan Bansal, Adam Fourney, and Eric Horvitz. Reading between the lines: Modeling user behavior and costs in ai-assisted programming. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, pp. 1–16, 2024.

Sydney Nguyen, Hannah McLean Babe, Yangtian Zi, Arjun Guha, Carolyn Jane Anderson, and Molly Q Feldman. How beginning programmers and code llms (mis) read each other. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, pp. 1–26, 2024.

Sida Peng, Eirini Kalliamvakou, Peter Cihon, and Mert Demirer. The impact of ai on developer productivity: Evidence from github copilot. *arXiv preprint arXiv:2302.06590*, 2023.

James Prather, Brent Reeves, Juho Leinonen, Stephen MacNeil, Arisoa S Randrianasolo, Brett Becker, Bailey Kimmel, Jared Wright, and Ben Briggs. The widening gap: The benefits and harms of generative ai for novice programmers. *arXiv preprint arXiv:2405.17739*, 2024.

Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Matej Balog, M Pawan Kumar, Emilien Dupont, Francisco JR Ruiz, Jordan S Ellenberg, Pengming Wang, Omar Fawzi, et al. Mathematical discoveries from program search with large language models. *Nature*, pp. 1–3, 2023.

Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, et al. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*, 2023.

Xingyao Wang, Yangyi Chen, Lifan Yuan, Yizhe Zhang, Yunzhu Li, Hao Peng, and Heng Ji. Executable code actions elicit better llm agents. *arXiv preprint arXiv:2402.01030*, 2024a.

Xingyao Wang, Boxuan Li, Yufan Song, Frank F Xu, Xiangru Tang, Mingchen Zhuge, Jiayi Pan, Yueqi Song, Bowen Li, Jaskirat Singh, et al. Opendevin: An open platform for ai software developers as generalist agents. *arXiv preprint arXiv:2407.16741*, 2024b.

Yuxiang Wei, Zhe Wang, Jiawei Liu, Yifeng Ding, and Lingming Zhang. Magicoder: Source code is all you need. *ICML*, 2024.

Martin Weyssow, Aton Kamanda, and Houari Sahraoui. Codeultrafeedback: An llm-as-a-judge dataset for aligning large language models to coding preferences. *arXiv preprint arXiv:2403.09032*, 2024.

Chunqiu Steven Xia, Yinlin Deng, Soren Dunn, and Lingming Zhang. Agentless: Demystifying llm-based software engineering agents. *arXiv preprint arXiv:2407.01489*, 2024.

John Yang, Carlos E. Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. Swe-agent: Agent-computer interfaces enable automated software engineering, 2024a.

Ke Yang, Jiateng Liu, John Wu, Chaoqi Yang, Yi R Fung, Sha Li, Zixuan Huang, Xu Cao, Xingyao Wang, Yiquan Wang, et al. If llm is the wizard, then code is the wand: A survey on how code empowers large language models to serve as intelligent agents. *arXiv preprint arXiv:2401.00812*, 2024b.

Kechi Zhang, Jia Li, Ge Li, Xianjie Shi, and Zhi Jin. Codeagent: Enhancing code generation with tool-integrated agent systems for real-world repo-level coding challenges. *arXiv preprint arXiv:2401.07339*, 2024.

Tianyu Zheng, Ge Zhang, Tianhao Shen, Xueling Liu, Bill Yuchen Lin, Jie Fu, Wenhu Chen, and Xiang Yue. Opencodeinterpreter: Integrating code generation with execution and refinement. *arXiv preprint arXiv:2402.14658*, 2024.