VerifAI: AI Verification in the Wild

1 Workshop Summary

This workshop explores the intersection of scale-driven generative artificial intelligence (AI) and the correctness-focused principles of verification. Formal analysis tools such as theorem provers, satisfiability solvers, and execution monitoring have demonstrated success in ensuring properties of interest across a range of tasks in software development and mathematics where precise reasoning is necessary. However, these methods face scaling challenges. Recently, generative AI such as large language models (LLMs) has been explored as a scalable and adaptable option to create solutions in these settings. The effectiveness of AI in these settings increases with more compute and data, but unlike traditional formalisms, they are built around probabilistic methods – not correctness by construction. In the VerifAI: AI Verification in the Wild workshop we invite papers and discussions that discuss how to bridge these two fields. Potential angles include, but are not limited to the following:

- **Generative AI for formal methods:** Formal methods offer strong guarantees of desired or undesired properties, but they can be challenging to implement. When faced with non-halting proofs or extensive search spaces, machine learning approaches can help guide those search processes effectively [5, 14] and LLMs may even write the theorems themselves [3, 13]. How can we further integrate AI to enhance verification practices? How can we ensure that AI-generated test conditions [11, 2] align with actual desired properties?
- **Formal methods for generative AI:** Generative AI can benefit from formal methods, which provide assurance and thus build trust. For example, satisfiability solvers can be used as a bottleneck in reasoning domains [8, 10], code generated by the model can be annotated with specifications for program analysis tools to ensure its correctness [9, 15], and even simple symbolic methods such as automata simulators can steer AI generations towards more logically consistent behavior [16]. How else can we integrate formal methods into generative AI development and usage?
- **AI as verifiers:** Hard guarantees can be notoriously rigid and/or difficult to achieve. In these cases, probabilistic methods are appealing alternatives to provide "soft assurances" [17]. How can we develop more robust and trustworthy verifiers from probabilistic methods? In what settings is it appropriate to make verification more flexible using probabilistic methods?
- **Datasets and benchmarks:** The advancement of research at the intersection of generative AI and formal methods relies heavily on the availability of robust datasets and benchmarks. We welcome papers that present new datasets and benchmarks in reasoning, theorem proving, code generation, and related areas. How can we design benchmarks that accurately reflect the challenges in combining probabilistic models with formal (or informal) verification?
- **Special Theme: LLMs for Code Generation** The use of LLMs for code generation has grown significantly, with a growing body of research advocating for the integration of formal structures and tools, such as context-free grammars [12], static analyzers [1], and SMT-guided repair [7, 6]. These methods aim to improve both the safety and effectiveness of code generated by LLMs, particularly for low-resource programming languages. In the context of code

generation, LLM agents can leverage tool use and learning from execution feedback [4] to validate their generations. This year, our special theme invites researchers to explore how techniques from programming languages and formal methods communities can further enhance LLM-driven code generation.

We welcome novel methodologies, analytic contributions, works in progress, negative results, and review and positional papers that will foster discussion. We will also have a track for tiny or short papers.

2 Tentative schedule

To encourage discussion from different perspectives on this intradisciplinary topic, the program includes a series of 6 invited talks (25 minutes + 5 minutes QA), 3 highlighted talks (10 minutes) selected from submitted papers, a panel discussion (1 hour), two poster sessions (1 hour), and time allocated for unstructured conversation among attendees. A tentative schedule is shown below:

Time	Event	Time	Event
8:55 - 9:00	Opening Remarks	12:30 - 1:30	Lunch break
9:00 - 9:30	Invited Talk 1	1:30 - 2:00	Invited Talk 5
9:30 - 10:00	Invited Talk 2	2:00 - 2:30	Invited Talk 6
10:00 - 10:30	Invited Talk 3	2:30 - 3:00	Highlight Talks
10:30 - 11:00	Coffee Break	3:00 - 4:00	Poster Session 2
11:00 - 12:00	Poster Session 1	4:00 - 5:00	Discussion Panel
12:00 - 12:30	Invited Talk 4	5:00 - 5:05	Concluding Remarks

3 Invited speakers and panelists

Sida I. Wang, *Research Scientist, FAIR*, Website: **(confirmed)**: Sida I. Wang is a research scientist at Facebook AI Research (FAIR). He was previously an instructor at Princeton University and the Institute for Advanced Study. He completed his PhD in computer science at Stanford University under Chris Manning and Percy Liang. Sida researches unsupervised translation, including recent work with reasoning about programming languages. Sida himself has received various scholarships and his work has received outstanding paper awards.

Shan Lu, *Professor, University of Chicago*, Website: **(confirmed)**: Shan Lu is a professor at the University of Chicago. She was previously the Clare Boothe Luce Assistant Professor at University of Wisconsin, Madison. Her research focuses on software reliability and efficiency, particularly detecting, diagnosing, and fixing functional and performance bugs in large software systems. Shan is an ACM Distinguished Member (2019 class), an Alfred P. Sloan Research Fellow (2014), a Distinguished Educator Alumnus from Department of Computer Science at University of Illinois (2013), and NSF Career Award recipient (2010). Her co-authored papers won Google Scholar Classic Paper 2017, Best Paper Awards at ACM-SIGOPS SOSP 2019, USENIX OSDI 2016 and USENIX FAST 2013, 3 ACM-SIGSOFT Distinguished Paper Awards at ICSE 2019, ICSE 2015 and FSE 2014, an ACM CHI Honorable Mention Award 2021, an ACM-SIGPLAN Research Highlight Award at PLDI 2011, and an IEEE Micro Top Picks in ASPLOS 2006. Shan is also a member of the informal ASPLOS Hall of Fame.

Kevin Ellis, *Assistant Professor, Cornell*, Website: **(confirmed)**: Kevin Ellis is an assistant professor at Cornell University. He previously received his PhD at MIT under Joshua Tenenbaum and Armando Solar-Lezama. He researches the intersection of machine learning, program synthesis, and cognitive science. His work has been published broadly, including recognition in Nature Communications Editors' Highlights. Kevin has been recognized with the NSF CAREER Award, Google Research Scholar Award, MIT Presidential Graduate Fellowship.

Emily First, *Postdoc, UCSD*, Website: **(confirmed)**: Emily First is a postdoctoral researcher at UC San Diego working with Sorin Lerner. She previously completed her PhD at UMass Amherst under Yuriy Brun. Her research is at the intersection of software engineering, programming languages, and machine learning. She focuses on creating tools to automatically generate proofs of software correctness in Coq and Isabelle/HOL. Her work has received the ACM SIGSOFT Distinguished Paper Award.

Nikitha Rao, *PhD Student, CMU*, Website: **(confirmed)**: Nikitha Rao is a PhD student at Carnegie Mellon University, advised by Vincent Hellendoorn and Claire Le Goues. She researches the use of large language models in code generation and reasoning. She holds several patents on using machine learning in software support systems. Nikitha has received many fellowships and scholar awards for her work. Her work has been nominated for and received several best paper awards, and been featured in popular science magazines.

Luke Ong, *Distinguished University Professor, Vice President of Research, Nanyang Technological University, Singapore*, Website: **(in contact)**: Luke Ong is a Distinguished University Professor and Vice President of Research at Nanyang Technological University. His research spans semantics of computation, programming languages, verification, and probabilistic programming, with recent work focusing on higher-order logic, cyber security, and differentiable programming. Luke is renowned for co-developing game semantics and initiating higher-order model checking, and has made pioneering contributions to both theoretical and practical aspects of computer science. He has been general and founding chairs of many ACM/IEEE interest groups in logic, computation, and software theory. He is a joint recipient of the prestigious Alonzo Church Award for Outstanding Contributions to Logic and Computation.

Elizabeth Polgreen, *Assistant Professor, University of Edinburgh*, Website: **(confirmed)**: Elizabeth Polgreen is an assisant professor at the University of Edinburgh. She is interested in formal program synthesis techniques and the use of synthesis to increase the scalability of verification. She holds a research fellowship from the Royal Academy of Engineering. Her work in lifting C code to a tensor DSL has received the Best Paper Award at GPCE.

Gabriel Synnaeve, *Research Scientist, FAIR*, Website: **(in contact)**: Gabriel Synnaeve is a research scientist on the Facebook AI Research (FAIR) team, who joined as a postdoctoral researcher in 2015. Prior to Facebook, Gabriel was a postdoctoral fellow in Emmanuel Dupoux's team at École Normale Supérieure in Paris, working on reverse-engineering the acquisition of language in babies. Gabriel received his PhD in Bayesian modeling applied to real-time strategy games AI from the University of Grenoble in 2012. Gabriel programmed a bot that placed 4th in the AAAI AIIDE 2012 StarCraft AI competition. In 2009, Gabriel worked on inductive logic programming applied to systems biology at the National Institute of Informatics in Tokyo.

Nadia Polikarpova, *Associate Professor, UCSD*, Website: **(in contact)**: Nadia Polikarpova is an associate professor at UC San Diego. She is a Sloan Fellow and recipient of the Intel Rising Stars Award and NSF CAREER Award. Nadia researches how to build tools for program verification and synthesis. Her work has received distinguished paper awards at numerous venues including PLDI, OOPSLA, ICFP, and POPL.

Koushik Sen, *Professor, Berkeley*, Website: **(in contact)**: Koushik Sen is a professor in the Department of Electrical Engineering and Computer Sciences at the University of California, Berkeley. His research interest lies in Software Engineering, Programming Languages, and Formal methods. He is interested in developing software tools and methodologies that improve programmer productivity and software quality. He has received a NSF CAREER Award in 2008, a Haifa Verification Conference (HVC) Award in 2009, a IFIP TC2 Manfred Paul Award for Excellence in Software: Theory and Practice in 2010, a Sloan Foundation Fellowship in 2011, a Professor R. Narasimhan Lecture Award in 2014, an Okawa Foundation Research Grant in 2015, and an ACM SIGSOFT Impact Paper Award in 2019. He has won several ACM SIGSOFT Distinguished Paper Awards. He received the C.L. and Jane W-S. Liu Award in 2004, the C. W. Gear Outstanding Graduate Award in 2005, and the David J. Kuck Outstanding Ph.D. Thesis Award in 2007, and a Distinguished Alumni Educator Award in 2014 from the UIUC Department of Computer Science.

Pengcheng Yin, *Research Scientist, Google Deepmind*, Website: **(in contact)**: Pengcheng Yin is a research scientist at Google Deepmind, working at the intersection of natural language processing and machine learning to produce research and products for software engineering. Prior to that, he was a PhD student at Carnegie Mellon University with Graham Neubig. His work has been widely published.

4 Organizers and biographies

Celine Lee
PhD Student, Cornell
cl923@cornell.edu
Website, Google Scholar
Celine Lee is a PhD candidate at Cornell University, advised by Sasha Rush. Her research focuses on studying the capabilities of LLMs for low-level code and symbolic reasoning. Her work has been presented at top-tier conferences such as ICLR, ACL, EMNLP, and KDD. She has interned as a researcher at Intel Labs and IBM Research, and holds several patents in AI-supported program synthesis and code management.

Wenting Zhao
PhD Student, Cornell
wzhao@cs.cornell.edu
Website, Google Scholar
Wenting Zhao is a Ph.D. candidate in Computer Science at Cornell University, advised by Claire Cardie and Sasha Rush. Her research focuses on reasoning: she develops techniques that can accurately reason over real-world scenarios and creates benchmarks that reliably reflect the reasoning performance of language models when deployed in the real world (most recently: WildChat, Commit0). Her work has been presented at various top-tier conferences including ICLR, ICML, AAAI, ACL, EMNLP, and NAACL. She was named the intern of the year at AI2, and has previously organized reasoning tutorials and workshops at ACL conferences.

Ameesh Shah
PhD Student, UC Berkeley
ameesh@berkeley.edu
Website, Google Scholar
Ameesh Shah is a Ph.D. candidate in Computer Science at UC Berkeley, advised by Sanjit Seshia. His research focuses broadly at the intersection of machine learning and formal methods, with aims of building assurances for learning-enabled systems through formal logic and verification. His work, which spans across program synthesis, learning-enabled Cyber-Physical Systems (CPS) and specification mining, has appeared at top venues in ML and formal methods including NeurIPS, ICML, ICLR, and FMCAD. He is the recipient of the NDSEG Fellowship and has interned with groups at Microsoft Research and Toyota Research Institute concerning safe and trustworthy ML.

Theo X. Olausson
PhD Student, Massachusetts Institute of Technology
theoxo@mit.edu
Website, Google Scholar
Theo X. Olausson is a PhD candidate at the Massachusetts Institute of Technology, where he is advised by Armando Solar-Lezama. His research focuses on improving the reliability and scalability of LLMs and other foundation models, often by integrating symbolic tools such as interpreters or SAT solvers into the loop. Theo's work has appeared at conferences such as ICLR, ACL, POPL, and EMNLP, and was recognized with an Outstanding Paper award at EMNLP'23 as well as a Presidential Fellowship from MIT. He has interned with the Deep Learning group at Microsoft Research as well as the Simons Foundation's Center for Computational Mathematics, where he remains officially associated as a Guest Researcher.

Tao Yu
Assistant Professor, The University of Hong Kong
tao.yu.nlp@gmail.com
Website, Google Scholar
Tao Yu is an Assistant Professor at The University of Hong Kong. He completed his Ph.D. in Computer Science from Yale University. His research aims to build language model agents that transform ("grounding") language instructions into code or actions executable in real-world environments, including databases, web applications, and the physical world. Tao has received the Google and Amazon faculty research awards (Google Research Scholar Award 2023, Amazon Research Award 2022). Tao has served on the organizing committee of ACL 2023, SUKI: Structured and Unstructured Knowledge Integration Workshop@NAACL 2022, and IntEx-SemPar: Interactive and Executable Semantic Parsing Workshop@EMNLP 2020.

Sean Welleck
Assistant Professor, CMU
wellecks@cmu.edu
Website, Google Scholar
Sean Welleck is an Assistant Professor at Carnegie Mellon University, where he leads the Machine Learning, Language, and Logic (L3) Lab. His areas of focus include generative models, algorithms for large language models, and AI for code and mathematics. Sean received a Ph.D. from New York University. He was a postdoctoral scholar at the University of Washington and AI2. Sean has co-organized several workshops on AI + Math (Math AI for Education, NeurIPS 2021; Math AI, Neurips 2022; Math AI, Neurips 2023; AI for Math, ICML 2024).

5 Marketing and anticipated audience size

We will create and maintain a workshop website advertising the topic, call for papers, speaker list, and tentative schedule. We will promote the workshop website and call-for-papers through social media platforms, email lists, and research communities to reach researchers from various disciplinary and biographic backgrounds.

We aim for our workshop to attract researchers from the machine learning and programming languages community alike. The number of research papers in code generation and code reasoning has also been increasing in recent years. Based on these factors and recent attendance in related workshops, we anticipate ~ 300 attendees will participate in this workshop.

6 Diversity Commitment

Diversity of organizers and speakers. Our organizing committee spans multiple universities across multiple countries (Cornell, Berkeley, MIT, CMU, HKU). Our team spans PhD students to assistant professors, and has a diverse expertise in machine learning, language modeling, code generation, reasoning, executable language grounding, formal methods, program synthesis, neurosymbolic AI, code verification, theorem proving, and evaluation.

We have also invited 10 distinguished speakers/panelists from different affiliations (FAIR, Google Deepmind, UCSD, Cornell, NTU, University of Edinburgh, University of Chicago, Berkeley, CMU), selected to represent a wide range of perspectives on the workshop topic.

Between the organizing committee and speakers, half of the workshop team are female researchers. The home institutions of our team cover North America, parts of Europe and parts of Asia.

Diversity of participants. We plan to ensure the presence of all related communities by broadcasting our workshop to faculty, researchers, and students working in these areas. When calling for participation, we will refer to the BIG Directory[1] to encourage submissions/participation from the underrepresented groups. We will use part of any external funding we secure to support such participants.

7 Access

The workshop will be primarily held in person, but to accommodate those who cannot physically attend, we will maintain and share a virtual livestream through which remote attendees can participate. At all times, an organizer will monitor the stream for questions directed to the the organizers and/or speakers. All accepted papers, talks, and discussions will also be posted to our aforementioned website for perusal during and after the workshop.

8 Previous related workshops

While past workshops have discussed related topics, none have specifically addressed our goal with VerifAI: AI Verification in the Wild.

- Deep Learning for Code (DL4C)
- LLMs for Code (LLM4Code)
- Mathematical Reasoning and AI (MATH-AI)
- ML for Systems
- Workshop on Formal Verification and Machine Learning (WFVML)
- Symposium on AI Verification (SAIV)

Previous workshops have fostered conversation about deep learning methods for code, computer systems, software engineering (e.g. DL4C, LLM4Code, ML for Systems) and mathematic reasoning (e.g. MATH-AI) but without a particular focus on verification. Other workshops have focused on formal verification for and with a broad range of AI tools (e.g. WFVML, SAIV), but do not highlight opportunities newly available in the context of modern LLMs.

[1] `http://www.winlp.org/big-directory/`

References

[1] L. A. Agrawal, A. Kanade, N. Goyal, S. Lahiri, and S. Rajamani. Monitor-guided decoding of code lms with static analysis of repository context. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 32270–32298. Curran Associates, Inc., 2023.

[2] N. Alshahwan, J. Chheda, A. Finogenova, B. Gokkaya, M. Harman, I. Harper, A. Marginean, S. Sengupta, and E. Wang. Automated unit test improvement using large language models at meta. In *Companion Proceedings of the 32nd ACM International Conference on the Foundations of Software Engineering*, FSE 2024, page 185–196, New York, NY, USA, 2024. Association for Computing Machinery.

[3] Z. Azerbayev, H. Schoelkopf, K. Paster, M. D. Santos, S. M. McAleer, A. Q. Jiang, J. Deng, S. Biderman, and S. Welleck. Llemma: An open language model for mathematics. In *The Twelfth International Conference on Learning Representations*, 2024.

[4] J. Gehring, K. Zheng, J. Copet, V. Mella, T. Cohen, and G. Synnaeve. Rlef: Grounding code llms in execution feedback with reinforcement learning, 2024.

[5] A. Q. Jiang, S. Welleck, J. P. Zhou, T. Lacroix, J. Liu, W. Li, M. Jamnik, G. Lample, and Y. Wu. Draft, sketch, and prove: Guiding formal theorem provers with informal proofs. In *The Eleventh International Conference on Learning Representations*, 2023.

[6] C. Lee, A. Mahmoud, M. Kurek, S. Campanoni, D. Brooks, S. Chong, G.-Y. Wei, and A. M. Rush. Guess & sketch: Language model guided transpilation. In *The Twelfth International Conference on Learning Representations*, 2024.

[7] F. Mora, J. Wong, H. Lepe, S. Bhatia, K. Elmaaroufi, G. Varghese, J. E. Gonzalez, E. Polgreen, and S. A. Seshia. Synthetic programming elicitation and repair for text-to-code in very low-resource programming languages, 2024.

[8] T. X. Olausson*, A. Gu*, B. Lipkin*, C. E. Zhang*, A. Solar-Lezama, J. B. Tenenbaum, and R. P. Levy. Linc: A neuro-symbolic approach for logical reasoning by combining language models with first-order logic provers. 2023.

[9] K. Pei, D. Bieber, K. Shi, C. Sutton, and P. Yin. Can large language models reason about program invariants? In *Proceedings of the 40th International Conference on Machine Learning*, 2023.

[10] G. Poesia, K. Gandhi, E. Zelikman, and N. D. Goodman. Certified reasoning with language models. 2023.

[11] M. L. Siddiq, J. C. Da Silva Santos, R. H. Tanvir, N. Ulfat, F. Al Rifat, and V. Carvalho Lopes. Using large language models to generate junit tests: An empirical study. In *Proceedings of the 28th International Conference on Evaluation and Assessment in Software Engineering*, EASE '24, page 313–322, New York, NY, USA, 2024. Association for Computing Machinery.

[12] B. Wang, Z. Wang, X. Wang, Y. Cao, R. A. Saurous, and Y. Kim. Grammar prompting for domain-specific language generation with large language models. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 65030–65055. Curran Associates, Inc., 2023.

[13] S. Welleck, J. Liu, X. Lu, H. Hajishirzi, and Y. Choi. Naturalprover: Grounded mathematical proof generation with language models. In A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[14] S. Welleck and R. Saha. Llmstep: Llm proofstep suggestions in lean. *arXiv preprint arXiv:2310.18457*, 2023.

[15] C. Yang, X. Li, M. R. H. Misu, J. Yao, W. Cui, Y. Gong, C. Hawblitzel, S. Lahiri, J. R. Lorch, S. Lu, F. Yang, Z. Zhou, and S. Lu. Autoverus: Automated proof generation for rust code, 2024.

[16] H. Zhang, P.-N. Kung, M. Yoshida, G. V. den Broeck, and N. Peng. Adaptable logical control for large language models, 2024.

[17] L. Zheng, W.-L. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. Xing, H. Zhang, J. E. Gonzalez, and I. Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 46595–46623. Curran Associates, Inc., 2023.