



Exploring Generative AI in Automated Software Engineering

Miroslaw Staron and Silvia Abrahão 

GENERATIVE AI HAS rapidly become a transformative force in both industry and academia. HuggingFace, one of the premier repositories for sharing AI models, boasts over 160,000 text generation models, with more than 5,000 tailored for generative source code. Alongside these open source models are proprietary commercial models like OpenAI's GPT or Google's Gemini. But how are these models influencing software engineering research? In this edition of the "Practitioner's Digest," we look into which of these 160,000 models are used in the current research presented at the 39th IEEE/ACM International Conference on Automated Software Engineering (ASE 2024) held in Sacramento, CA, USA, in October 2024.

What Models Are Researchers Using?

To uncover which models were used in the papers presented at the conference, we selected those containing the keywords "LLM" or "Large Language Model" in their title or keywords. We read the papers and

extracted the name of the model. If the paper used several sizes of the same model (e.g., 1 billion and 7 billion), we counted the model only once. Our analysis found 20 papers employing 24 unique models. [Figure 1](#) highlights the frequency of model usage at the conference.

The paper showcasing the most diverse set of models was "DataRecipe—How to Cook the Data for CodeLLM?" by Kisub Kim, Jounghoon Kim, Byeongjo Park, Dongsun Kim, Chun Yong Chong, Yuan Wang, Tiezhu Sun, Daniel Tang, Jacques Klein, and Tegawendé F. Bissyandé. This study's primary contribution is demonstrating how curated datasets can improve code generation performance. It evaluates the quality of existing benchmarks for generative models that write source code in Java. By manipulating different properties of these models, the research team demonstrates how these manipulations impact the training quality of the following models: CodeGen, StarCoder, MagiCoder, CodeLlama, Llama2, and DeepSeek-Coder. One key finding is that the DeepSeek-Coder model (1.3 billion and 6.7 billion parameters) outperforms other models on the ODEX dataset (compared with all

models and all datasets). The results also show that results from the same model can vary significantly depending on the dataset. For instance, the DeepSeek-Coder with 1.3 billion parameters has a bilingual evaluation understudy metric that is the best for the ODEX dataset (80.06) and the second-worst for the Alpaca dataset (only 18.62). The paper was presented at ASE 2024. Access it at <https://tinyurl.com/3ec752fx>.

Commercial Versus Open Source Models

OpenAI's GPT models (GPT-3.5 and GPT-4) were the most frequently used commercial models. Researchers cited two main reasons for choosing these over open source alternatives: 1) ease of use, i.e., the Representational State Transfer application programming interface (API) provided by OpenAI simplifies integration, and 2) quality of the results, i.e., GPT models consistently outperform open source models in quality. For instance, in the study "Enabling Cost-Effective UI Automation Testing with Retrieval-Based LLMs: A Case Study in WeChat" by Sidong Feng, Haochuan Lu, Jianqin Jiang, Ting Xiong, Likun Huang, Yinglin Liang, Xiaoqin Li, Yuetang Deng, and Aldeida Aleti, GPT-4

outperformed Llama3-70B (70 billion parameters) with a 90% success rate on the completion task for user interface (UI) automation testing, compared with 71% when used as part of the tool presented in that paper. The response of that model is also typically twice as fast as that of the open source model, although these results can vary greatly depending on the local configuration. The ease of use of the OpenAI API is unmatched, and even frameworks such as Ollama (<https://www.ollama.com/>) are not used as frequently as that API. The paper was presented at ASE 2024. Access it at <https://tinyurl.com/4rewj4na>.

Beyond Code Generation

Generative AI's utility at ASE 2024 extended far beyond traditional code completion. The tasks range from typical programming, such as UI test generation and source code commenting, to vulnerability detection and model checking or safety assurance.

For example, in the paper “CoqPilot, a Plugin for LLM-Based Generation of Proofs” by Andrei Kozyrev, Gleb Solovev, Nikita Khramov, and Anton Podkopaev, large language models (LLMs) are used to generate formal proofs in Coq. The goal of using the generative AI model is twofold. First, it enables novice users to enter the world of proof-writing much more easily than without this technology, and second, it provides a platform for experimenting with proof generation. This paper introduces a new Visual Studio Code extension, CoqPilot, which seamlessly integrates this technology with existing programming tools. An interesting fact is that this paper, the only one of those reviewed, uses the newest GPT-4o model, which had just been released at the time of the

ASE 2024 submission deadline. Even in this paper, the commercial GPT models outperform the open source (and older) Llama2-13B model (50% versus 2%) for short proofs. The GPT-4o with the CoqPilot approach was able to prove 39% of the theorems, 51% of them on the first try. The paper demonstrates that generative AI and theorem provers can significantly increase the effectiveness and efficiency of writing and proving formal proofs. The paper was presented at ASE 2024. Access it at <https://tinyurl.com/mry7xwvv>.

The Role of Data

When using generative AI in research, the application is almost as important as the data used to train and evaluate the models. Most of the papers presented at the conference used existing datasets complemented with newly generated ones. The paper “ContractTinker: LLM-Empowered Vulnerability Repair for Real-World Smart Contracts” by Che Wang, Jiashuo Zhang, Jianbo Gao, Libin Xia, Zhi Guan, and Zhong Chen is one such

example. The paper presents a ContractTinker tool based on LLMs such as GPT and Llama. The tool utilizes the models with the chain-of-thought technique to analyze vulnerabilities in smart contracts. The authors extract new data from an existing platform, Code4Rena (<https://code4rena.com/>). The tool can generate strategies to fix the vulnerability. The Top3 success rate for the best model is 91.6% (Top3 means that one of the three best strategies generated by the tool corresponds to the actual strategy used to fix the vulnerability). At the same time, the tool was able to correctly generate 48% of the patches and an additional 21% that required minor modifications. The key to the success of this tool was, as in the previous examples, the combination of LLM technology with high-level static and semantic analysis. The paper was presented at ASE 2024. Access it at <https://tinyurl.com/42f997vr>.

The Emergence of AI Agents

Jensen Huang, CEO of Nvidia, highlighted generative AI-backed

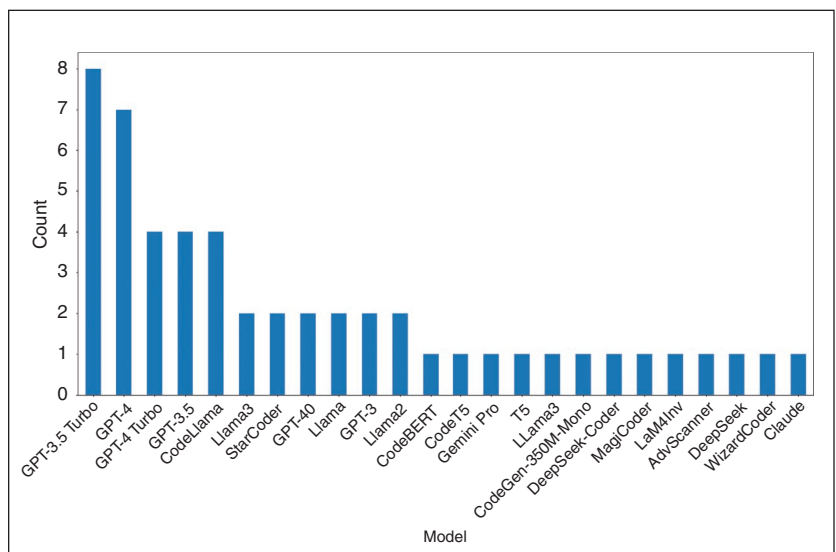


FIGURE 1. The frequency of generative AI models used in papers presented at ASE 2024.

Table 1. Takeaways from the use of generative AI models at the ASE 2024 conference.

Model Use	Applications
Leverage proven commercial APIs for immediate gains	Commercial models like GPT-4 and GPT-4o consistently outperform open source alternatives in terms of both ease of use and output quality. Practitioners can take advantage of the robust APIs provided by companies like OpenAI to integrate these capabilities into existing workflows with minimal overhead. This is especially relevant for tasks requiring quick deployment or consistent performance, such as UI automation and test case generation.
Experiment with open source models for customization	Despite their performance gap, open source models like Llama2 and DeepSeek-Coder provide unique opportunities for customization and cost efficiency. For teams with domain-specific requirements or budget constraints, these models can be fine-tuned to specific tasks, offering competitive performance at a fraction of the cost of commercial solutions.
Combine AI with domain-specific techniques	The most impactful results from ASE 2024 stemmed from combining generative AI with domain-specific methods. For instance, integrating LLMs with theorem provers (as in CoqPilot) or static and semantic analysis (as in ContractTinker) significantly enhanced their utility. Practitioners should explore how such hybrid approaches can enhance their own tools and processes.
Focus on data quality and domain relevance	The success of generative AI applications is often as much about the quality and relevance of the data as it is about the model itself. Tools like ContractTinker demonstrate the importance of curating high-quality datasets and employing techniques like chain-of-thought reasoning to improve results. Practitioners should prioritize creating or acquiring domain-specific datasets to maximize the effectiveness of generative AI in their projects.
Prepare for AI-driven collaboration and automation	The emergence of agent-based systems, such as PairCoder, indicates a shift toward more autonomous and collaborative AI tools. These systems promise to augment human expertise by providing high-level planning and detailed execution capabilities. Practitioners should explore how these emerging tools could support software engineering tasks such as pair programming, code review, and other collaborative tasks.
Monitor emerging benchmarks and research	Tools and methods showcased at ASE 2024 highlight the importance of staying informed about the latest research and benchmarks. Practitioners can benefit from adopting proven approaches, such as the use of HumanEval and CodeContest for performance evaluation, to assess the suitability of generative AI models for their specific needs.

agents as the future during his Consumer Electronics and Services keynote. We can see traces of it at this conference. Although there are only a handful of papers using agents, the one that caught our attention the most is titled “A Pair Programming Framework for Code Generation via Multi-Plan Exploration and Feedback-Driven Refinement” by Huan Zhang, Wei Cheng, Yuhan Wu, and Wei Hu. This paper proposes a new language model-based code generation tool, much like Github CoPilot. However, this tool uses two models instead of one. One of the models is responsible for the high-level planning of the solution

(called Navigator), and the other is used for the detailed implementation (called Driver). This tool improves performance on top of the existing approaches, such as GPT-3.5 or DeepSeek-Coder, by adding at least 5% to the pass@1 metric (meaning that the first solution is acceptable/correct). The performance of this new tool, PairCoder, reaches 87.8% for the HumanEval benchmark, which is very good. In addition, on the challenging CodeContest benchmark, the tool achieves 17.95% pass@1 with GPT-3.5-turbo, representing the best performance at the time under comparable computational costs. The paper was presented at ASE

2024. Access it at <https://tinyurl.com/36wyyhs7>.

Conclusions and Takeaways

Generative AI is reshaping software engineering research, as evident at ASE 2024. The breadth of its applications—spanning from code generation to vulnerability repair, formal proofs, and even advanced agent-based programming frameworks—shows its transformative potential for the field. While LLMs dominate, their impact is amplified when paired with complementary techniques such as static analysis, semantic analysis, and theorem proving. These hybrid approaches not only unlock new possibilities but also

align with commercial advancements seen in tools like ChatGPT-4o1. Table 1 summarizes the key insights derived from the reviewed papers, highlighting several takeaways for practitioners and suggesting some applications in their own contexts.

As generative AI continues to evolve, its integration into software engineering promises to redefine productivity and innovation for researchers and practitioners alike. By embracing both the opportunities and challenges presented by this technology, software engineers can position themselves at the forefront of the next wave of innovation in this field. 🌐

ABOUT THE AUTHORS



MIROSLAW STARON is a full professor in the Interaction Design and Software Engineering Division in the Computer Science and Engineering Department shared by Chalmers University of Technology and the University of Gothenburg, Gothenburg 412 96, Sweden. Contact him at <https://www.staron.nu> or mirosław.staron@cse.gu.se.



SILVIA ABRAHÃO is a full professor at the Department of Computer Science, Universitat Politècnica de València, Valencia 46022, Spain. Contact her at <https://sabrahao.wixsite.com/dsic-upv> or sabrahao@dsic.upv.es.

IT Professional

TECHNOLOGY SOLUTIONS FOR THE ENTERPRISE

CALL FOR ARTICLES

IT Professional seeks original submissions on technology solutions for the enterprise. Topics include

- emerging technologies,
- cloud computing,
- Web 2.0 and services,
- cybersecurity,
- mobile computing,
- green IT,
- RFID,
- social software,
- data management and mining,
- systems integration,
- communication networks,
- datacenter operations,
- IT asset management, and
- health information technology.

We welcome articles accompanied by web-based demos. For more information, see our author guidelines at www.computer.org/itpro/author.htm.

WWW.COMPUTER.ORG/ITPRO

Digital Object Identifier 10.1109/MS.2025.3551381



IEEE
COMPUTER
SOCIETY

