EDUHints 2.0: AI-Assisted Hint Generation with Local Inference (NSF IUSE)

Taylor Wolff, The Evergreen State College

Taylor Wolff is an undergraduate computer science student and research assistant at The Evergreen State College.

Jack Cook, The Evergreen State College

Jack Cook is a Cybersecurity consultant and lead maintainer of the EDURange Cybersecurity Training Platform

Richard Weiss, The Evergreen State College

Richard Weiss is currently a Member of the Faculty at The Evergreen State College and has been teaching security and information assurance since 2003. He received an A.B. in mathematics from Brandeis University and a Ph.D. in mathematics from Harvard University. His research has included cybersecurity education, computer vision and robotics, applications of machine learning, and computer architecture.

Jens Mache, Lewis & Clark College

Jens Mache is an educator and researcher at Lewis & Clark College in Portland, Oregon.

EDUHints 2.0: AI-Assisted Hint Generation via Local Inference (NSF IUSE)

The problems that motivate our work are: 1) it is not easy for instructors to tell which students need help and to give them the right hints, and 2) it is not feasible for instructors or TAs to be available 24/7 to answer questions when students are working on assignments. In addition, it is now tempting for students to use large language models to obtain solutions without advancing their learning. Thus, our goal is to create a semi-automated hint system to help the student learn when they are doing their assignments, and provide a human-in-the-loop interface to instructors for facilitating these exercises. This work-in-progress paper presents our successor to the hint generation system *EDUHints*. While the *EDUHints* system's first iteration was closely integrated into a specific cybersecurity framework, this new version was redesigned as an open-source library for seamless integration with other applications that can capture student actions and deliver hints.

1 Introduction

EDUHints [1] is a tool for generating hints which the instructor can send either as is, with modification, or not at all. [2]. Our goal was to research how emerging AI technologies could assist students while preserving quality student-instructor interaction [3]. While any program can interface with the library, thus making the library agnostic to specific methods of hint generation/delivery systems, the system was designed with a special focus on human-in-the-loop hint generation. With this design, the instructor serves as the intermediary between the AI and the student (see Figure 1). This has three key benefits: we need not worry about the AI giving an ineffective hint to the student or the hint containing inappropriate information/hallucinating, and lastly we can classify the hints as acceptable or not.

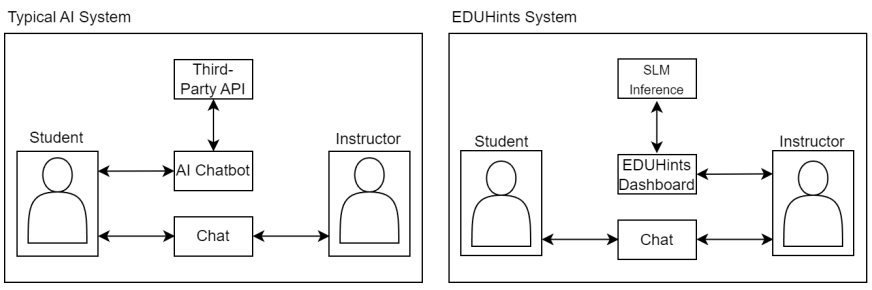

Figure 1: Typical AI system usage versus EDUHints system usage.

2 Related Work

Previous research efforts regarding the development and testing of AI systems that assist students have primarily focused on the integration of large language models or LLMs [4–7]. Yet, there are many drawbacks to LLMs. They can use excessive amounts of computing power, are costly for the student or school, may require the sharing of student data with a third-party AI provider, and can hallucinate answers. Yu et al [8] propose a possible alternative for educational software, the small language model (SLM). SLMs are language models with fewer parameters than their larger counterparts. Lu et al [9] note that

> The vision behind SLMs is to democratize access to machine intelligence, making it both accessible and affordable to people everywhere. This approach seeks to make intelligence ubiquitous and practical, available to anyone, anywhere, at any time – much like the human brain, which everyone possesses.

While SLMs typically perform worse than LLMs, methods such as retrieval-augmented generation (RAG) can sometimes enhance them such that they outperform larger models [10]. Because of their reduced computational requirements, many SLMs can be locally deployed and run on consumer CPUs [11], potentially providing a low-cost, private, and reasonably performant alternative to third-party LLM APIs.

3 Technical Specifications

EDUHints 2.0 was developed with the Python programming language and features a modular architecture, allowing for easy expansion and customization. For inference the system leverages the open-source *Phi-4-mini-instruct* model from Microsoft via Hugging Face's Transformers library. Unlike most AI applications today, the system uses no third-party API for computation, instead inference is computed on the host machine (either physically local to the user or local to a third-party cloud instance for example). The capability to locally deploy the system presents two key advantages, firstly this prevents sensitive student data from being sent to third-party AI services, giving peace-of-mind to instructors and academic institutions and secondly, the system is effectively free-of-cost to operate assuming the instructor has access to reasonable computer hardware (i.e, a 4-core CPU, 8GB of RAM, 100GB of disk space).

4 User Experience

The *EDURange* framework and standalone *EDUHints* library are available for download at github.com/edurange/edurange3 and github.com/edurange/eduhints respectively. While *EDUHints 1.0* featured a dedicated page and dashboard, when integrating the new library, we decided to reduce mental context switching for the instructor by encapsulating the *EDUHints* system's interface within the platform's integrated chat messaging system. Here the instructor can view the recent logs of each students, then after identifying a student who may need assistance they can generate them a hint. The program will then inference the local SLM with relevant context, including the selected student's recent bash, chat and answer logs and other relevant context data that the user can upload via text files (the process is documented via the project's

README.md). Once generated the hint will appear in the corresponding text box and the instructor can either send the hint or regenerate it or close the sub-dashboard.

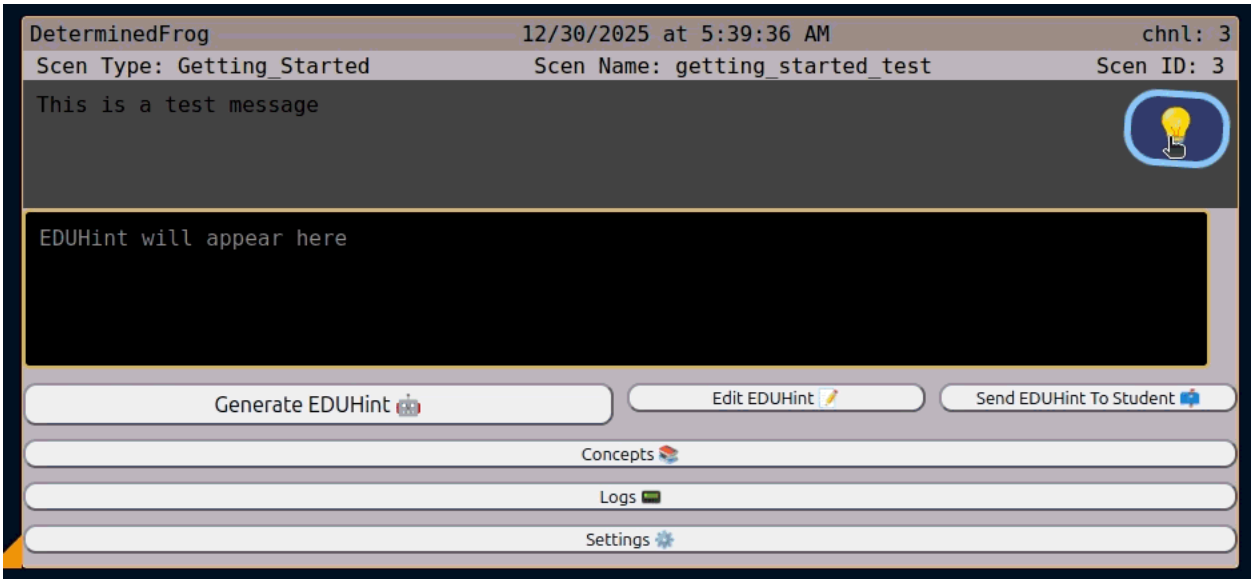

Figure 2. Screenshot of *EDUHints 2.0*'s sub-dashboard within EDURange's integrated chat messaging system.

5 Conclusions and Future Work

Following the library's public release in January 2026, *EDUHints 2.0* was been tested across several classes. Notably, the SLM had trouble tracking the progress of individual students throughout exercises and sometimes continued to focus on a question that the student asked at the beginning. This indicates that we need to provide better context to the SLM with information about the current state of the student, i.e. what they are currently working on. We plan to repeat the survey used in [1] to measure students' subjective feedback, compare offline performance metrics between *EDUHints 1.0* and *EDUHints 2.0*, such as hint-generation times, and collect additional feedback from instructors. For long-term plans, we'd like to find collaborators who wish to customize, expand, and integrate the library into their own educational platforms, thereby demonstrating the library's broad applicability. The system's expandability is well attested by recent work [12] demonstrated substantial improvements to the quality of system-generated hints by integrating *EDUHints 1.0* into their own GraphRAG hint generation system.

Acknowledgements

This work was partially supported by the National Science Foundation under IUSE Awards 2216485 and 2216492.

References

[1] T. Wolff, R. Weiss, J. Cook, J. Granville, and J. Mache, "EDUHints: A human-in-the-loop small language model hint generation system for cybersecurity education," in *24th European Conference on Cyber Warfare and Security*, 2025.

[2] C. Zhai, S. Wibowo, and L. D. Li, "The effects of over-reliance on AI dialogue systems on students' cognitive abilities: a systematic review," *Smart Learning Environments*, vol. 11, no. 1, p. 28, Jun 2024. [Online]. Available: https://doi.org/10.1186/s40561-024-00316-7

[3] C. K. Y. Chan and L. H. Tsi, "Will generative AI replace teachers in higher education? a study of teacher and student perceptions," *Studies in Educational Evaluation*, vol. 83, p. 101395, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0191491X24000749

[4] A. Hellas, J. Leinonen, and L. Leppänen, "Experiences from integrating large language model chatbots into the classroom," in *Proceedings of the 2024 on ACM Virtual Global Computing Education Conference V. 1*, ser. SIGCSE Virtual 2024. New York, NY, USA: Association for Computing Machinery, 2024, p. 46–52. [Online]. Available: https://doi.org/10.1145/3649165.3690101

[5] J. Salminen, S.-g. Jung, J. Medina, K. Aldous, J. Azem, W. Akhtar, and B. J. Jansen, "Using cipherbot: An exploratory analysis of student interaction with an llm-based educational chatbot," in *Proceedings of the Eleventh ACM Conference on Learning @ Scale*, ser. L@S '24. New York, NY, USA: Association for Computing Machinery, 2024, p. 279–283. [Online]. Available: https://doi.org/10.1145/3657604.3664690

[6] A. Lieb and T. Goel, "Student interaction with newtbot: An LLM-as-tutor chatbot for secondary physics education," in *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*, ser. CHI EA '24. New York, NY, USA: Association for Computing Machinery, 2024. [Online]. Available: https://doi.org/10.1145/3613905.3647957

[7] A. T. Neumann, Y. Yin, S. Sowe, S. Decker, and M. Jarke, "An LLM-driven chatbot in higher education for databases and information systems," *IEEE Transactions on Education*, vol. 68, no. 1, pp. 103–116, 2025.

[8] Z. Yu, S. Liu, P. Denny, A. Bergen, and M. Liut, "Integrating small language models with retrieval-augmented generation in computing education: Key takeaways, setup, and practical insights," in *Proceedings of the 56th ACM Technical Symposium on Computer Science Education V. 1*, ser. SIGCSETS 2025. New York, NY, USA: Association for Computing Machinery, 2025, p. 1302–1308. [Online]. Available: https://doi.org/10.1145/3641554.3701844

[9] Z. Lu, X. Li, D. Cai, R. Yi, F. Liu, X. Zhang, N. D. Lane, and M. Xu, "Small language models: Survey, measurements, and insights," 2025. [Online]. Available: https://arxiv.org/abs/2409.15790

[10] S. Liu, Z. Yu, F. Huang, Y. Bulbulia, A. Bergen, and M. Liut, "Can small language models with retrieval-augmented generation replace large language models when learning computer science?" in *Proceedings of the 2024 on Innovation and Technology in Computer Science Education V. 1*, ser. ITiCSE 2024. New York, NY, USA: Association for Computing Machinery, 2024, p. 388–393. [Online]. Available: https://doi.org/10.1145/3649217.3653554

[11] "GitHub - ggml-org/llama.cpp: LLM inference in C/C++," https://github.com/ggml-org/llama.cpp, accessed: 2026-01-21.

[12] I. Abraham, J. Mache, T. Wolff, J. Cook, R. Weiss, and H.-A. Wang, "SLMs meet GraphRAG: A structured approach to context-aware cybersecurity hint generation," in *International Conference on Cyber Warfare and Security*, 2026.