Hyunjae Suh

Ph.D. Student in Software Engineering — University of California, Irvine

hyunjas@uci.edu • hyunjas.github.io • LinkedIn • Google Scholar

Irvine, CA

Summary

Ph.D. student in Software Engineering specializing in Generative AI, Large Language Models (LLMs), and Machine Learning for software development. Experienced in building scalable machine learning pipelines, fine-tuning LLMs, and evaluating AI-generated code for reliability, accessibility, and software quality assurance.

Research Interests: Large Language Models (LLMs), Generative AI, Machine Learning, Software Engineering, Code Intelligence, Foundation Models, Fine-tuning, Prompt Engineering, Multi-Agent Systems, ReAct Frameworks

Education

University of California, Irvine Sep. 2023 – Present (Expected Jun. 2027)
Ph.D. in Software Engineering *Irvine, CA*

Kookmin University Mar. 2017 – Aug. 2023
B.S. in Computer Science *Seoul, South Korea*

Industry Experience

Applied Scientist Intern Jun. 2025 – Sep. 2025
Amazon *Austin, TX*

- Developed and deployed ML-based change risk models on large-scale Prime Video codebases (10M+ LOC) to proactively flag risky commits and improve deployment reliability.
- Engineered diff-aware representations using LLMs and evaluated ensemble classifiers (Random Forest, XGBoost), reaching 84.60 F1-score and demonstrating the effectiveness of feature design in risk prediction tasks.
- Led large-scale experiments analyzing predictive features of risky changes, integrating risk scores into release pipelines to reduce failed deployments.

Graduate Research Assistant Aug. 2023 – Dec. 2023
eBay *Remote*

- Fine-tuned and evaluated open-source Large Language Models (LLMs) (e.g., Vicuna, GPT-Neo) for automated commit message generation, reducing manual documentation effort.
- Applied advanced prompt engineering and instruction tuning to standardize commit messages across 50+ repositories, enhancing clarity, stylistic consistency, and developer productivity.
- Benchmarked multiple LLMs on domain-specific datasets, analyzing trade-offs between accuracy, fluency, and token efficiency to inform model selection and deployment.

Research Experience

Ph.D. Researcher – AI & Software Engineering Sep. 2023 – Present
University of California, Irvine *Irvine, CA*

- Conduct empirical studies on LLMs for software engineering tasks, with focus on accessibility, quality, and attribution.
- Built a detection pipeline combining fine-tuned LLMs, code embeddings, and feature-based classifiers, achieving state-of-the-art results (82.55 F1-score) on benchmark datasets — published at **ICSE 2025**.
- Designed and tested prompting strategies (Few-Shot, Self-Criticism, Multi-Agent Debate, ReAct) to reduce accessibility violations of AI-generated applications.
- Developed *FeedA11y*, a ReAct-based pipeline integrating accessibility evaluation feedback (IBM Equal Access, AChecker) into refinement loops for accessible code generation.
- Conducted large-scale experiments on fine-tuning order in LLMs for software engineering tasks, analyzing cross-task interference and transfer dynamics.

Publications

An Empirical Study on Automatically Detecting AI-Generated Source Code: How Far Are We?
***Hyunjae Suh**, Mahan Tafreshipour, Jiawei Li, Adithya Bhattiprolu, Iftekhar Ahmed.*
ICSE 2025 (47th IEEE/ACM International Conference on Software Engineering). [DOI]

Human or LLM? A Comparative Study on Accessible Code Generation Capability
***Hyunjae Suh**, Mahan Tafreshipour, Sam Malek, Iftekhar Ahmed.*
arXiv preprint, 2025.

Does the Order of Fine-tuning Matter and Why?
*Qihong Chen, Jiawei Li, **Hyunjae Suh**, Lianghao Jiang, Zheng Zhou, Jingze Chen, Jiri Gesi, Iftekhar Ahmed.*
arXiv preprint, 2024.

Projects

Detection of LLM-generated Source Code ICSE 2025

- Developed a **state-of-the-art detection pipeline** combining fine-tuned LLMs, code embeddings, and feature-based classifiers for automatic code attribution.
- Achieved **82.55 F1-score** on benchmark datasets, setting a new standard for detecting LLM-generated source code.

Accessible Code Generation via Prompting in LLMs

- Designed and implemented *FeedA11y*, a **ReAct-based pipeline** integrating formal accessibility evaluation tools (IBM Equal Access, AChecker) into LLM refinement loops.
- Evaluated advanced prompting strategies (Self-Criticism, Multi-Agent Debate) to improve WCAG 2.1 compliance in AI-generated UI code.

The Impact of Fine-tuning Order on LLMs for SE

- Conducted large-scale empirical experiments analyzing the effect of fine-tuning task order on cross-task interference and knowledge transfer across software engineering tasks (e.g., Code Translation, Code Summarization, Defect Prediction, Clone Detection).

Service

Reviewer May 2025 – Present
ACM Transactions on Software Engineering and Methodology (TOSEM)

Reviewer Jun 2025 – Present
IEEE Transactions on Reliability (TR)

Teaching Experience

Teaching Assistant Sep. 2023 – Present
University of California, Irvine *Irvine, CA*

- IN4MATX 115 - Software Testing, Analysis, and Quality Assurance
- ICS 10 - How Computers Work
- ICS 32 - Programming with Software Libraries
- CS 121 - Information Retrieval

Technical Skills

Languages: Python, Java, R, JavaScript, C++, SQL
Machine Learning & AI: Deep Learning, Large Language Models (LLMs), Generative AI, Transformer Models, Model Fine-tuning, Prompt Engineering, NLP (Natural Language Processing)
Frameworks: PyTorch, Hugging Face, scikit-learn, Keras, Pandas, NumPy, Matplotlib, XGBoost, LangChain
Tools: AWS SageMaker, AWS Bedrock, AWS S3, Git, LaTeX
Code Analysis: Checkstyle, Detekt, ESLint, Soot, Understand
Research Methods: Empirical Studies, Quantitative Analysis, Data Visualization