Huanting Wang — Curriculum Vitae

Email: schwa@leeds.ac.uk Personal Website: https://huantwang.github.io/

Education

10/2021 – 08/2025 (Expected date)	University of Leeds, United Kingdom	*Ph.D. in computer science* • **6 Publications** • **School of Computing Full Scholarship (2 places)**
09/2018 – 07/2021	Northwest University, China (Tier 1A)	*MSc in Software Engineering* • **4 Publications + 5 patents** • **First Class Scholarships (top 5% students)**
09/2014 – 07/2018	Chang'an University, China (Tier 1A)	*BSc in Software Engineering*

Professional Experience

12/2024 - Present	University of Leeds	*Research Fellow*
08/2021 – 11/2021	Alibaba DAMO Academic	*Research Intern in LLM Group*
07/2019 – 12/2019	Ant Group	*Software Engineer Intern in Security Group*

Selected Publications

[1] SecureMind: A Framework for Benchmarking Large Language Models in Bug Detection and Repair,

H. Wang, D. Jacob, D. Kelly, Y. Elkhatib, J. Singer, Z. Wang,

Proceedings of the International Symposium on Memory Management (ISMM), 2025

[2] Enhancing Deployment-Time Predictive Model Robustness for Code Analysis and Optimization,

H. Wang, P. Lenihan, Z. Wang,

Proceedings of the International Symposium on Code Generation and Optimization (CGO), 2025

Premier ACM conference in Compiler Optimization (CORE A)

Distinguished Paper Award!

[3] *Combining Structured Static Code Information and Dynamic Symbolic Traces for Software Vulnerability Prediction,*

H. Wang, Z. Tang, S. Chen, Jie. Wang, Y. Liu, H. Fang, C. Xia, Z. Wang,

Proceedings of the International Conference on Software Engineering (ICSE), 2024

Premier ACM conference in Software Engineering (CORE A*)

[4] *Automating reinforcement learning architecture design for code optimization,*

H. Wang, Z. Tang, C. Zhang, J. Zhao, C. Cummins, H. Leather, Z. Wang,

Proceedings of the 31st ACM SIGPLAN International Conference on Compiler Construction (CC), 2022

Premier ACM conference in parallel computing (CORE A)

[5] *Combining Graph-based Learning with Automated Data Collection for Code Vulnerability Detection,*

H. Wang, G. Ye, Z. Tang, S.H. Tan, S. Huang, D. Fang, Y. Feng, L. Bian, Z. Wang**,**

IEEE Transactions on Information Forensics and Security (TIFS), 2021

Flagship journal in Computer Security.

ESI Top 1% Highly Cited Paper!

Programming Skills

AI; Deep/Reinforcement Learning; Software Security; Code Optimization; Python; C++; Pytorch/Tensorflow;

Awards

2025	CGO **Distinguished Paper Award**; CGO **Travel Grant**
2024	AI **SuperConnector Award**
2023	MITACS Globalink **Research Award**;

Research Experience

My research experience lies in the **code analysis** (Software Security) and **code optimization** using **machine learning (ML)** techniques. I have participated in the following projects during my MSc and PhD studies:

- **Large Language Model for Software Security** **January 2025 - now**

Large language models (LLMs) hold great promise for automating software vulnerability detection and repair, but ensuring their correctness remains a challenge. We introduce "SecureMind", an open-source framework for evaluating LLMs in vulnerability detection and repair, with a focus on memory-related vulnerabilities. We evaluate 10 representative LLMs on 16,000 test samples covering 8 types of vulnerabilities.

*This work has led to one paper published in **ISMM 2025 [1]**.*

- **Robust Machine Learning during Deployment Time** **April 2022 to Jan 2025**

In recent years, ML has emerged as a powerful tool for assisting code analysis and optimization tasks. ML models can be vulnerable to changes in the deployment environment. Even slight alterations in hardware or application workloads can severely impact their accuracy.

We introduce "Prom" to enhance the robustness and performance of predictive models against such changes during deployment. We applied Prom to 12 representative ML models, covering heterogeneous device mapping, GPU thread coarsening, loop vectorization, source-code level bug detection and Tensor tuning. Prom successfully identifies 90% (up to 100%) of mispredictions and enhances prediction performance in operational environments through incremental learning.

*This work has led to one paper published in **CGO 2025 [2]**.*

- **Hybrid Learning-based Software Vulnerability Prediction** **December 2021 to Aug 2023**

Deep Learning (DL) is increasingly employed for software bug and vulnerability detection, extracting program representations from static code sources such as code texts. DL may face challenges from complex code structures, redundant statements, and extensive execution paths, potentially reducing the performance.

We proposed using DL to learn program presentations by combining static source code information and dynamic program execution traces. We have successfully uncovered more than 100 unique vulnerabilities and yielded 36 new, unique CVE IDs and outperform 14 prior methods by providing higher accuracy.

*This work has led to one paper published in **ICSE 2024 [3]**.*

- **Automatic Reinforcement Learning Model Architecture Design** **July 2020 to Apr 2022**

While programmers apply reinforcement learning (RL) to their domain, the first step is to design the RL architecture for their tasks. However, expertise creates a barrier between programmers and RL.

We proposed an open-source framework for automating RL architecture search, simplifying RL integration into compilers. We applied it to four optimization problems: image pipelines, neural network code generation, code size reduction, and superoptimization. Experimental results demonstrate its superiority, improving performance and accelerating deployment-stage search by an average of 1.75x (up to 100x).

*This work has led to one paper published in **CC 2022 [4]**. Collaboration with **Meta AI research lab**.*

- **Deep Program Structure Modeling using Graph Neural Networks** **June 2019 to January 2021**

DL is promising for code-related tasks like compiler optimization. An important factor is having the right representation to characterize the model input for the given task. Existing approaches in the area typically treat the program structure as a sequential sequence but fail to capitalize on the rich semantics of data and control flow information, for which graphs are a proven representation structure.

We introduce a novel Graph Neural Network (GNN) approach to learn rich code representations from program graphs, effectively capturing diverse code relationships—including data flow and control flow—that are critical for downstream tasks. We apply our method to four classification tasks, and experimental results consistently demonstrate its superiority over competing approaches.

*This work has led to 3 papers published in **IEEE TIFS[5], PACT 2020 and JISA**.*