Huanting Wang Curriculum Vitae

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

Education

10/2021 – 04/2025 (Expected date)	University of Leeds, United Kingdom	*Ph.D. in computer science* • **5 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** • **3x First Class Scholarships (top 5% students), during 2018, 2019 and 2020**
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*	*Research Intern in LLM group*
07/2019 – 12/2019	Ant Group *Software Engineer*	*Software Engineer Intern in Security group*

Selected Publications

[1] *Enhancing Predictive Model Robustness During Deployment: A Case Study for Code Analysis and Optimization,*

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

The International Symposium on Code Generation and Optimization (CGO), 2025

Premier ACM conference in Compiler Optimization (CORE A)

[2] *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*)

[3] *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)

[4] *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, ranking #2 of the top publication list in the "Computer Security & Cryptography" category according to Google Scholar's metric.

ESI top 1% highly cited paper and was featured on several online media.

Programming Skills

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

Awards

2024	AI Super Connector **Award** (£ 20K)
2023	MITACS Globalink **Research Award** ($ 6K)
2021	School of Computing **Full Scholarship** of University of Leeds (2 places)

Research Experience

My research experience in the area of **vulnerability detection** and **compiler optimization** through the use of **machine learning techniques**. I have participated in the following projects during my MSc and Ph.D. study:

- **Robust Machine Learning for Program Modeling** **April 2023 to now**

In recent years, machine learning has emerged as a powerful tool for assisting code analysis and optimization tasks. Machine learning 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 a statistical analysis based framework, PROM, to enhance the robustness of predictive models against such changes during deployment. We applied our system to 13 machine learning models, covering heterogeneous device mapping, GPU thread coarsening, loop vectorization, source-code level bug detection and Tensor tuning. Our system successfully identifies 96% (up to 100%) of mispredictions and enhances prediction performance in operational environments through incremental learning.

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

- **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. By implementing a focused symbolic execution solution, we bring the benefits of static and dynamic code features while reducing the expensive symbolic execution overhead. We have successfully uncovered more than 70 unique vulnerabilities and yielded 36 new, unique CVE IDs and also outperformed 14 prior methods by providing higher accuracy and lower false positive rates.

*This work has led to one paper published in **ACM ICSE 2024 [2]**.*

- **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, Supersonic, 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 super-optimization. Experimental results show improved performance and accelerated deployment-stage search by an average of 1.75x (up to 100x).

*This work has led to one paper published in **ACM CC 2022 [3]**. Collaboration with **Facebook AI Research**.*

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

Deep learning 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 introduced a novel Graph Neural Networks approach, Funded, to learn valuable code representations from program graphs, distinguishing diverse code relationships, including data and control flow in source code and LLVM IR, critical for downstream tasks. We apply our approach to four tasks: device mapping, thread coarsening, loop vectorization, and vulnerability detection, across various programming languages. Experimental results consistently show its superiority over competing methods. Funded is better than six state-of-the-art code vulnerability detection models.

*This work has led to three papers published in **IEEE TIFS 2021 [4], etc**. Collaboration with **Ant group**.*

- **Large Language Model** **August 2021 - November 2021**

My work involves creating and implementing machine learning models, tools, and infrastructure. I ensure their seamless integration into Alibaba's pre-trained model ecosystem. *Collaboration with **DAMO Academic**.*