Yuxiang Qiu

yuxqiu@proton.me | github.com/yuxqiu | yuxqiu.github.io Last updated in 2025-05-20

Education

University College London 09/2021 - 06/2025

MEng Computer Science

- **Grades**: 1st class (87%, 1st year, rank: 1/150), 1st class (86%, 2nd year)
- **Coursework**: Algorithms for Computer Systems, Computer Architecture & Concurrency, Decentralized Finance, Intro to Cryptography, Logic, Malware, Networked Systems, Security, Supervised Learning, Theory of Computation

Georgia Institute of Technology 08/2023 – 05/2024

BS Computer Science (Exchange Student)

- **GPA**: 4.0/4.0
- **Coursework**: Blockchain & Cryptocurrency, Compiler & Interpreter, Computer Graphics, Deep Learning, Design & Analysis of Algorithm, Processor Design, Quantum Computing

Experience

Thesis: Trustless Efficient Light Clients 10/2024 - 04/2025

- Background: Light clients enable resource-constrained devices to interact with blockchains but face scalability challenges due to linear-time verification. Existing solutions, designed for specific blockchains, lack generalizability, restricting their applicability to diverse blockchain ecosystems.
- Research: Proposed **Mim**, a general-purpose light client protocol using folding-based SNARKs for **constant-sized proofs** and **sub-linear verification** of committee rotations; introduced the **Levelled Merkle Forest (LMF)**, a novel accumulator reducing proof generation time and enabling variable-length proofs for enhanced scalability.
- Implementation: Authored **8,000+ lines of Rust** using `arkworks` and `sonobe`; delivered the **first generic R1CS hash-to-curve implementation** for BLS12 curves and **resolved a critical bug** in `arkworks` related to emulated field variables that caused unsatisfiable constraints.
- Experiment: Evaluated Mim on a blockchain modeled after Sui; showed that committee rotation proofs can be generated in **<12 hours** without LMF and **<7 days** with LMF for one year's data (with sufficient memory), while maintaining **~1-3s verification** for all chain lengths; demonstrated that LMF matches Merkle trees in proving/verification performance and achieved **reduction in proof size** for all checkpoints.
- Advisor: Prof. Philipp Jovanovic and Dr. Alberto Sonnino

Research Assistant 06/2024 – 09/2024

UCL Software Optimisation, Learning and Analytics Research Lab *London, UK*

- Background: Recent advances in LLM show the promise of using it to judge text quality. However, current methods lack interpretability and are vulnerable to adversarial attacks. To address this, we explored methods to score a software patch explanation by measuring how well an LLM can accomplish tasks with this explanation.
- Research: Reviewed 10+ available datasets; proposed LLM-as-a-judge as the baseline; evaluated and **enhanced 4 text perturbation methods**; designed ways to **improve and measure the diversity** of generated text.
- Implementation: **Integrated SWE-bench** into the eval framework; implemented fault localization and differential testing as tasks; designed the transitivity-based comparison algorithm used in surveys.
- Experiment: Designed and conducted experiments to **analyze the performance** (in terms of agreement, Kendall's Tau, and Spearman's correlation) in different settings (with CoT, different perturbations, etc.).
- Advisor: Prof. Federica Sarro and Prof. Sergey Mechtaev

Software Development Engineer Intern 06/2023 – 08/2023

Amazon *London, UK*

- Researched cross-platform portability of Java apps running on Windows, resulting in a ~10-page research report.
- Delved into the Java SE Specifications (JVMS and JLS), the JAR file specifications, and the OpenJDK source code.
- Developed a Java application and library that performs **incompatibility detection at the bytecode** level (checking for 7 different types of cross-platform issues) with **~80% accuracy and 90%+ recall**.
- Optimized libraries by profiling hot spots and bringing parallelism to CPU-bound tasks, resulting in a **3x speedup**.

Teaching Assistant
UCL
- 2024-2025: COMP0002 Principles of Programming, COMP0004 Object-Oriented Programming
- 2022-2023 Programming Tutor: Tutored 12 students in 6 programming languages (C, C++, Rust, Haskell, Java, Python) and familiarized them with shell scripting, computer networking, and frontend/backend development.

Open Source Contributions
- **AI**: `pytorch/torcheval` (#195), `princeton-nlp/SWE-bench` (#186, #189, #212)
- **Crypto**: `arkworks-rs/r1cs-std` (#157, #161)
- **PL**: `rust-lang/rust-clippy` (#11865, #12084, #12094), `typst/biblatex` (#34)

Projects

Probabilistic Data Structures and Algorithms 02/2025 - 04/2025
- Accelerated the Morris Counter by implementing **geometric sampling**, achieving **up to 8x speedup** while preserving error guarantees, validated through rigorous benchmarking on varied stream sizes and error parameters.
- Designed an **adaptive sketch selection mechanism** that dynamically chooses between Count Min Sketch and Count Sketch queries by estimating the L2 norm, **reducing MSE** across diverse Zipfian distributions.
- Developed **two novel Deletable Learned Bloom Filter designs**, a Sandwiched Learned Bloom Filter (SLBF) with two Counting Bloom Filters and a Learned Bloom Filter (LBF) with three Standard Bloom Filters, optimizing tradeoffs between FPR, FNR, and deletability, with simulations showing **superior FPR for SLBF** and **guaranteed deletability for LBF** with a controlled FNR.

TrueLearn 01/2023 - 08/2023
- Led a team of 4 students to **implement a Python machine-learning library** with a family of baseline and Bayesian classifiers for building learner models to predict their engagement with educational resources.
- **Created 9 static and interactive visualizations** to present the learner representations in humanly-intuitive ways.
- Conducted **hyperparameter tuning** via grid search and evaluated library scalability by analyzing wall-clock time.
- Augmented the PEEKC dataset (with 30000+ Wikipedia data) to provide richer info during the entity linking process.
- Advisor: Dr. Sahan Bulathwela

Logic Parser 10/2022 - 12/2022
- Devised a **one-pass iterative parser** and a **tableau-based SAT solver** for propositional and predicate logics.
- Built efficient iterative algorithms for AST operations that support processing logic formulas of arbitrary size in a scalable way, with **performance comparable to the SOTA z3 solver** for propositional logic.

Awards

UCL Studentship for Research 2024
UCL Faculty Undergraduate Scholarships for Excellence (1 student per faculty, 1 out of 1000+ students) 2022

Publications

A Toolbox for Modelling Engagement with Educational Videos
Yuxiang Qiu, Karim Djemili, Denis Elezi, Aaneel Shalman Srazali, Mar'ia P'erez-Ortiz, Emine Yilmaz, John Shawe-Taylor and Sahan Bulathwela
Proceedings of the AAAI Conference on Artificial Intelligence, 2024

TrueLearn: A Python Library for Personalised Informational Recommendations with (Implicit) Feedback
Yuxiang Qiu, Karim Djemili, Denis Elezi, Aaneel Shalman, María Pérez Ortiz and Sahan Bulathwela
6th Workshop on Online Recommender Systems and User Modeling, ACM RecSys 2023

Skills

Languages: C++, C, Python, Rust, Java, Verilog, Solidity, HTML, CSS, JavaScript, Haskell, x86 Assembly, GLSL
Libraries: ANTLR, arkworks, Bootstrap, Flask, Koa, openai, OpenCV, OpenGL, PyTorch, scikit-learn, Vue.js