Shiyu Zhao

☎ +1 (650)422-8429 ◇ ✉ shiyuz@stanford.edu◇ https://shiyu-zhao.netlify.app/

EDUCATIONAL BACKGROUND

Stanford University, School of Engineering — Stanford, United States
Computer Science (Master's degree) — Sep 2023 - Mar 2025(expected)

Tsinghua University, Yao Class — Beijing, China
Computer Science and Technology (Bachelor's degree) — Sep 2019 - Jun 2023

- **GPA:** 3.84/4.0 **Coursework:** Algorithm design, Fundamentals of object-oriented programming, Machine learning, Artificial intelligence, Mathematics for computer science, Operating system, Distributed system, Data structure
- **Honors and Awards:** Rank 1/70,000+ students in Chinese College Entrance Examination in Ningxia (2019), Dean's List (2020, 2021, 2022), Outstanding graduate of Yao Class (Highest honor in the department, 2023)

SKILLS

- **Programming Languages:** Python, C/C++, SQL, bash, Java, JavaScript, MATLAB, LaTeX, Verilog, Go, VB
- **Tools:** PyTorch, Pandas, Linux, Git, NumPy, Unix, MySQL, Azure Data Studio, Jetbrains, TensorFlow, Redis
- **Language Skills:** Mandarin(Native), English(Fluent), TOEFL: 108(S25), GRE:338(V169+Q169+W4.0)

EXPERIENCES

Software Engineer Intern — Beijing, China
miHoYo — Jul 2023 - Aug 2023

- Collaborated on the development of foundation model in a five-people agile team from scratch with PyTorch.
- Designed and implemented the SFT (supervised fine-tuning) of the foundation model, including the game data and roleplay data collection/processing, prompt designing/engineering, error analysis and schema design.
- Developed an API where virtual idols/game NPCs can interact with user with actions and mood given different characteristic descriptions, supported by the finetuned LLM as backend.
- Created a fully automated toolkit with bash and C++ to evaluate the performance of virtual idols and game NPCs, added into the company's toolkits to expedite the future development.

Machine Learning Intern — Montreal, Canada
Montreal Institute for Learning Alogorithms (MILA) — Mar 2022 - Aug 2022

- Developed an end-to-end differentiable model that can dynamically adjust itself according to the user's input molecule and user's need to retrieve relevant molecules and make molecule property prediction.
- Engineered a pipeline to update molecules in a large database with asynchronize update and momentum update, achieving low cost and high efficiency, and designed a task-oriented training framework with EM algorithm.
- Published as part of TorchDrug platform for drug discovery, achieved on average 2-3% improvement for all models on HIV, ClinTox and PCQM datasets.

Research Assistant — Beijing, China
Tsinghua University, ZhipuAI — Jan 2021 - Nov 2021

- Introduced a well-functioning pretrain-finetune large model into knowledge graph area with great generalizability.
- Designed a knowledge graph triple transformation method to apply transformer on knowledge graph with ease and a mechanism to unify different downstream tasks of knowledge graph problems.
- Achieves SOTA on both in-domain and out-of-domain reasoning task, significantly outperforms previous SOTA CQD(ICLR 2021 best paper) by over 12.1% relatively on FB15k-237 and over 6.4% relatively on NELL995.
- Accepted by SIGKDD 2022: *Mask and Reason: Pre-Training Knowledge Graph Transformers for Complex Logical Queries* as the first co-author.

PROJECTS

Logic Message Passing Graph Neural Network: Used Python to model one-step reasoning in NLP as triangle update, and used C to build modules to improve model efficiency. Outperformed SOTA by 9.25%, paper under review.
Random Matrix Factorization of Large-scale Network Embedding: Used C++ to improve NetMF embedding by single-view SVD and speeded up the factorization by freigs algorithm. Expanded the network scale and boosted speed.
Distributed Wallets based on Blockchain: Used Golang to build a blockchain system for distributed wallets and transcations. Implemented merkle trees and digital signature to encrypt blockchain more effectively and securely.
Database Storage, Execution and Optimization: Used C++ to implement tuple storage in a B+ tree structure for efficient searching; a logical optimizer to achieve an optimized plan from the naive plan before execution; a volcano style executor including filter, seqscan, project, hashjoin, hash-aggregate, limit, distinct and sort.