Andrew M. Zhang

(510) 676-4193 | andrewmzhang@berkeley.edu | andrewmzhang.com | github.com/andrewmzhang

EDUCATION

University of California, Berkeley *Aug 2016 – Expected May 2021*

Pursuing Masters of Science in Computer Science, B. A. in Computer Science GPA: 3.907

Relevant Coursework: Machine Learning, Computer Security, Algorithms, Concepts of Probability Theory, Concepts of Statistics, Upper Div. Linear Algebra, Machine Structures, Discrete Math and Probability Theory, Data Structures, Designing Information Devices and Systems I/II, Structure and Interpretation of Computer Programs, Real Analysis, Computational Photography, Deep Reinforcement Learning, Operating Systems, (Graduate) Algorithms, Designing Deep Neural Networks, Abstract Algebra, Optimization Models, Financial Engineering I, Game Theory,

In Progress: Theoretical Statistics, Graduate Operating Systems, Computational Colour

Programming Skills: C/C++ (Nvidia CUDA) ∘ Java (Android) ∘ Python (Cython, Numpy, Sklearn, Tensorflow, Pytorch)

PUBLICATION CONTRIBUTIONS

- Symposium on Cloud Computing 2019: *Centaur - A Practical Serverless Framework for End-to-End ML Workflows*
- NeurIPS Workshop 2018: *A Case For Serverless ML*

EXPERIENCE

Citadel Securities - Market Making: Client Equities *June 2019 - Aug 2019* | *June 2020 - Aug 2020*

Quantitative Research Intern 2020

- Created a trading policy on exhaust orders in a reinforcement learning environment
- Drastically improved trading volume while staying within gross market value and symbol position limits
- Created a fast running trading simulation in Cython that simulates 300M orders in under a minute
- Researched various position and GMV tilt functions to encode opportunity cost as part of the trading utility function

Quantitative Research Intern 2019

- Improved predictions for which trades fall into the bottom ventile in terms of profit
- Ported and serialized my experimental Python model and preprocessor to C++
- Researched various methods of explaining the model's decisions and performance with respect to the data
- Significantly improved basis points (profit per execution dollar) of existing strategy

RISELab, UC Berkeley – Serverless Machine Learning (C++): Cirrus/Centaur/Twill *Aug 2017 - present*

Twill (in progress)

- Low-overhead synchronization primitives framework in OpenWhisk for serverless compute of complex tasks (SQL queries, ML tasks, Map-Reduce), eliminating the need for a high-performance coordinator server
- Implemented the Berkeley Big Data Benchmark SQL queries with various serverless and synchronization backends and identified performance issues with each. Backends: AWS Lambdas, OpenWhisk, Twill, Pywren, AWS S3, Redis, Pocket

Cirrus/Centaur: Project Link: github.com/jcarreira/cirrus. Paper links above

- Low-cost machine learning and hyperparameter optimization framework in C++ that runs serverlessly on AWS Lambdas
- Improved scalability without performance loss using multiple parameter servers (model sharding) for Logistic Regression
- Implemented collaborative filtering model using SGD, helped fix mathematical errors with original implementation
- Made Cirrus converge 70% faster than Spark on datasets, including Criteo Ad-Click Logs and Netflix Recommendations
- Created a Plotly Dash UI to visualize hyperparameter search and kill diverging experiments in Jupyter Notebook
- Implemented logistic regression with map reduce using Pywren (AWS S3, Lambdas) for a comparison baseline

uGSI, UC Berkeley - CS70: Discrete Math and Probability + CS189: Intro to Machine Learning *Jan 2017 - present*

Undergraduate Teaching Assistant. ~750 students

- Prepared mini-lectures, discussion questions, and test questions
- Worked with ~20 other uGSI's to answer students' questions, handle course logistics, and hold 1-on-1 sessions with students
- Ran office hours to help students on homework and theory questions

PERSONAL PROJECTS

Blinn: A C++ Raytracer for Metaballs - github.com/andrewmzhang/blinn - https://youtu.be/dMIecrYXcoE

- Created a raymarching metaball render from native C++
- Used Nvidia CUDA to aid in parallel renders, 1000x speedup over multithreaded CPU solution
- Renders video of several oscillating metaballs interconnected by springs with diffuse shading and Phong speculars