# Xuanyi Li

◇ 614-397-8198 ◇ shanelxy@outlook.com ◇ linkedin.com/in/wirybeaver ◇ github.com/wirybeaver ◇ shanelxy.top

## EDUCATION

**The Ohio State University** **Columbus, United States**
Master of Science in Computer Science; GPA (3.97/4.0) 08/2017 – 05/2019

**Zhejiang University** **Hangzhou, China**
Bachelor of Engineering in Automation; GPA (3.97/4.0) 09/2013 – 07/2017

## TECHNICAL SKILLS

**Programming Languages:** Java, C++, C, Bash, Go, SQL, Assembly, Python
**Frameworks:** Spring Boot, Apache Druid, Tensorflow, MapReduce, QFS, Kafka, PostgreSQL, MyBatis
**Operations:** Terraform, Jenkins, DataDog, Linux, Gradle, JUnit, Mockito, Git, Docker, Kubernetes
**AWS:** EC2, RDS, Lambda, S3, Route53, Route53 Health Check, IAM, EMR

## WORK EXPERIENCE

**Software Engineer** **LinkedIn, Sunnyvale**
*Deep Learning Authoring Infrastructure* 03/29/2021 – Present

- Calibrated Tensorflow's Multi Worker Mirrored Strategy (MWMS) run against Keras model and custom training loop model. Drove the win that model training speed can linearly be scaled in terms of total number of devices, i.e. GPUs, meanwhile the model is as accurate as it is trained on a single device.
- Introduced singleton pattern with double checked locking to ensure thread safe and bypass a bug caused by Tensorflow, which requires that collective ops must be initialized after creating MWMS.
- Accelerated the distributed training speeding of a LinkedIn's open source NLP model to 2.5X faster by specifying the sharding and batching logic to avoid the slow fall back behaviour caused by Tensorflow's awful auto sharding against AvroReader.

**Software Engineer** **Quantcast, Seattle**
*Audience Platform* 07/15/2019 – 03/26/2021

Daily Work:

- Develop an in house analytical engine called Kamke in Go that performs efficient set operations over distinguished 100 billion roaring bitmaps. It allows to deliver flexible planning / profiling / sequence insights on composite audience at interactive speed for Ads Planning.
- Architect and optimize the realtime analytics database Apache Druid hosted on AWS which handles insight calculations against billions of objects. The AWS services comprised prevailing RDS, Zookeeper, S3, EC2 and ELB and are coded via Terraform and Bash.
- Ingest terabytes of batch and Kafka streaming Ads Reporting data into Druid through proprietary MapReduce and customized Druid indexing extension respectively.
- Build gPRC API with Spring Boot that deliver insights backed up by Druid to brands and publishers.
- Leverage Gradle, Jenkins, JFrog, Docker to foster an agile CI/CD environment. Utilize AWS Lambda and DataDog to constantly probe and ensure the compliance of service health metrics.

Achievement Highlights:

- Introduced efficient event sequence insight query into in house analytical engine Kamke. Designed the data structure of innovative tiered roaring bitmap at byte level and the implementation of distributed query execution. Partnered with teammates on query parser, query generation and the data pipeline of sequence segments.
- Major contributor of Quantcast first real-time insights product to enable self-served advertising. This product consumes 1 billion streaming events per day with P95 data freshness of 30s.
  - Fixed a Druid internal bug that IntermediateRowParser erroneously truncated the tail of batch records after the first row not containing targeting labels.
  - Leveraged Having clause and Druid bitmap mechanism to speed up the worst case latency of the term prefix search query run against Druid by a factor of 4 times.
- Druid stakeholder of the company wide project to adopt Lambda Architecture for Ads Reporting.

- Ingested Kafka streaming data derived from different source into a single Druid dataset with minutely granularity. Regularly reindexed minutely segments into hourly bucket within the same dataset.
- Built reharvest pipeline via Spark and in house MapReduce flowing through Snowflake, QFS and S3.
- Stored timestamps from different sources as Druid Metrics instead of Columns to accelerate the data freshness query against different type of source by 10 times.

- Built the back-end of Pacing Visualization where daily budget goal is plotted alongside actual spend. This feature empowers customers to diagnose when they are having under or over delivery.
  - Incrementally synchronized from up-stream's Snowflake to PostgreSQL by exploiting JDBC drivers.
  - Used Mybatis Dynamic SQL and Type-Handler to deal with variant SQL queries and SQL Array type.
  - Accelerated time series aggregation by roughly 3.5 times through fine tuned Java Thread-Pool.
- Reduced the query concurrency by half by inserting new hierarchy aggregation dimensions via internal MapReduce. Sped up the re-harvesting by 16 times faster with Broadcast Join.
- Made extensive efforts to improve the maintainability and scalability of Druid cluster.
  - Designed and implemented color deployment policy to dramatically shrinks the cluster-wide deployment time from 1 workday to 1 hour mean-while guarantees zero query down time.
  - Tiered Druid cluster to prevent queries from different tenants intervening each other.
  - Integrated Fluentd to emit cluster logs to AWS ElasticSearch.
  - Migrated Terraform back end with multi workspaces from S3 to more secured Terraform Cloud.
- Wrote extendable probes, hosted on AWS Lambda, to emit data freshness into a centralized Datadog monitor. It compacts services having different SLO, segment granularity and data source type (batch or streaming) so as to reduce redundant code and maintenance workload, both current and anticipated.

## PROJECTS

**Raft based Fault-Tolerant Key/Value Storage in Go**

- Implemented the distributed consensus protocol Raft with Go channel, including leader election, heartbeats, log replication and persistence determination.
- Optimized log backtracking by add a conflictIndex in RPC reply to bring stale follower up to date quickly.
- Created a key/value service on top of Raft to cope with concurrent and duplicated client requests.
- Implemented snapshotting to avoid log grows without bound.

**Disk Oriented Storage Manager for the SQLite DBMS in C++**

- Developed thread-safe buffer pool manager, encompassing extendible hash table and LRU policy to move physical pages back and forth from main memory to disk.
- Built B+Tree index to support insertion, deletion, point search and iterator.
- Implemented latch crabbing protocol to allow multiple threads access and modify the B+Tree index.

**Other Side Projects**

- **Lisp Interpreter:** Implemented Lisp interpreter in Java to parse the expression into binary tree, evaluate arguments, bind them to associated formal parameters and recursively evaluate function body.
- **Seckill Shopping:** Used Redis to reject overselling requests. Combined RocketMQ with local message table to implement the distributed transaction per the choreography-based saga pattern.

## PUBLICATION

Xuanyi Li, Weimin Wu, Hongye Su. Convolutional Neural Networks Based Multi-Task Deep Learning for Movie Review Classification. In proceedings of the 4th IEEE DSAA. Tokyo, Japan, 10/2017.