Liyun Zhang

✉ zhang_ly@sjtu.edu.cn · ☎ (+86) 19959999549 · ⌂ liyun-zhang.github.io

Research Interests

My research interests mainly lie in **database systems**, **distributed systems** and **storage systems**. My previous research also includes work in the field of **deep learning**. I look forward to integrating this knowledge with various systems and designing systems for deep learning tasks, more specifically **ML4Sys** and **Sys4ML**.

Education

SEIEE, Shanghai Jiao Tong University (SJTU), Shanghai, China 2022.09 – Present

Master student in Computer Science (CS), GPA 3.73/4.0, advised by *Prof.* Guangtao Xue
Expected 2025.03

School of Computer Science, Hangzhou Dianzi University (HDU), Zhejiang, China 2018.09 – 2022.06

B.S. of Engineering in Computer Science (CS), GPA 3.98/5.0

Research Experience

VCEMO: Multi-Modal Emotion Recognition for Chinese Voiceprints (*Under review*) 2023.04 – 2023.09

This task aims to recognize the speaker's emotions based on provided acoustic features and transcriptions.

- Analyzed misclassified predictions made by existing models, pointing out the reasons for the inefficiency.
- **Leverage pre-trained models to generate a new modality incorporating external knowledge**. Alleviate the problem that some samples require external knowledge to understand.
- **Utilize a regularization based on contrastive learning**. Mitigate the disruption caused by label noise during training by reducing the constraints imposed by labels on the model training.
- Designed and implemented multiple sets of controlled experiments. Achieved results that surpass current SOTA models.

Project Experience

TinyKV Distributed System (*Golang*) 2023.11 – 2024.01

- It is a simplification of industrial implementation, TiKV, based on the design of Google Spanner.
- **Sharded Multi-Raft**. Implemented automatic resizing shards by splitting and peer-level configuration adjustments (including peer leaving, joining Raft group, and moving between groups).
- **Placement driver**. Implemented a scheduler to generate operators to balance each store's total size of regions.
- **Transaction concurrency control based on MVCC**. Implemented MVCC layer managing locks and writing records. Implemented transaction API for clients to perform transactional operations.

BusTub CMU 15-445/645 Database System (*C++*) 2023.07 – 2023.08

- Practiced commonly used C++ features and learned implementation for single-machine multi-threaded databases.
- **Index structure based on B+ trees**. Implemented the basic B+ tree data structures and algorithms, including search, insert, delete, and leaf scan. Optimized concurrency insertion and deletion with the optimistic locking protocol.
- **SQL execution operators and multiple optimizers**. Implemented Top-N optimization, optimized join operator complexity, enhanced data transmission between operators, and utilized index-based searches.
- **2-phase locking transaction concurrency control**. Implemented lock manager and deadlock detection. Support isolation levels of read uncommitted, read committed, and repeated read. Modified SQL executors to support concurrency execution.

MIT 6.824 Distributed System (*Golang*) 2022.08 – 2023.03

- Gained experience implementing and debugging distributed systems and analyzing system logs.
- Reproduced the **MapReduce** interface and implemented a **Key/Value storage engine based on the Raft algorithm**.
- **Read-only optimization**. Implemented read-index to allow followers to handle read-only requests. Achieved speedup of around 29× without snapshots and 8.6× with snapshots when executing 1000 reads consecutively after 1000 appends.
- **Pre-vote optimization**. Implemented pre-vote to prevent partitioned followers from interrupting the current term upon reconnecting to the Raft group.
- **Sharded Key/Value storage engine**. Support static sharding. Implemented group-level configuration modification (including Raft group leaving, joining, and shard moving) and a garbage collection mechanism.
- Configuration modification is proposed by administrators to the controller group. The installation process is designed to be an asynchronous operation that allows the Raft to serve a shard once it has received the necessary state.

As requested by these courses, I should not publish my codes in public GitHub repositories. If you are interested, I uploaded them to Google Drive. You can download them through this link.

Skills

- Familiar with C++, Golang, Pyhton. Proficient in PyTorch framework. Experienced with CUDA programming.
- Familiar with Git and LaTeX.