Research Statement

Yuhao Zhang
yuz870@eng.ucsd.edu
https://yhzhang.info

Advancements in ML/AI have brought workloads that put existing data analytics infrastructures and systems to the test. They ushered in an era of huge workloads that are increasingly computation- and data-intensive. Recent breakthroughs in Large Language Models (LLM) and chatbots such as ChatGPT indicate a new wave of innovation. However, many of these workloads run with severe inefficiency and face huge scalability challenges due to suboptimal scheduling, poor resource/memory management, or the lack of proper software system support. Today, accelerators such as GPUs crucial for the computation of AI/ML are expensive assets, with an Nvidia A100 costing tens of thousands and a state-of-art AI machine from Cerebras having a price tag of several million. Furthermore, with the stagnated Moore's law, large-scale and multiple computational nodes, and distributed computation are almost inevitable, making all the system issues mentioned more complex and the cost prohibitively large. My overall research interest is to **speed up AI/ML and reduce the costs**. I aim for faster and more affordable AI and want to work from ML algorithmic and systems angles. I want to make significant progress in ML systems and contribute to advancing the state-of-the-art. In the past five years of my Ph.D. study, advised by Prof. Arun Kumar at UCSD, I have built systems for various ML/AI workloads and gained the expertise to explore further and extend the frontier of ML systems research.

My research approach. The very core of ML systems research is still systems research. The various efficiency, cost, and scalability challenges are often the re-imagining or essentially extending of years-old OS, compiler, scheduling, and data system problems. What sets ML/AI workloads apart is the vast amount of dense data, huge computational costs, heavy communications, different access patterns, and more complex mathematical/statistical behaviors. The main challenge is correctly identifying the challenges and bottlenecks of ML/AI workloads, then repurposing and innovating upon well-established system techniques. The core mission is: given the finite set of methods, how do we synergize and innovate upon them to accommodate the ever-changing and seemly infinite variations of ML/AI workloads?

My research goal and impact. My primary research goal is to devise new software systems and abstractions for diversified ML/AI workloads to increase throughput, scalability, and usability, rooted in my in-depth understanding of both ML/AI algorithms and systems. In my past research, I captured the core challenges of various workloads on different data modalities, ranging from: model selection and training workloads on IID (e.g., images and tabular) and sequence (e.g., time series) data, both on data lake [6, 5] and within data management systems [9]; unbounded vocabulary querying workloads on video and image data [8]; and Graph Neural Network model selection and training workloads on graph data [7]. I proposed novel techniques drawing upon lessons from the worlds of database systems, distributed systems, and ML/AI and built novel systems, and built systems to boost the throughput and increase scalability and usability. All of my previous work has been released as open-source software. My past research on Cerebro [6, 5] and Cerebro-DS [9] have been incorporated into the Apache MADlib open-source project [1] and offered in Greenplum Database by VMware. The same projects have also been integrated with Spark [2], and Databricks is also reviewing the same project to provide to their customers. A prominent graph DBMS vendor is also interested in my work of Lotan [7].

Past Research

Resource-efficient DL model selection and training system. A significant bottleneck to the broader adoption of DL exists: the pain and resource intensiveness of model selection. An empirical process involves exploring deep net architectures and hyper-parameters, often requiring hundreds of trials. However, most DL systems focus on training one model at a time, reducing throughput and raising overall resource costs; some also sacrifice reproducibility. Towards higher throughput and resource utilization and as a part of a grander vision [3, 4] of DL model selection systems, I built Cerebro [6]. Cerebro is a data system to raise deep net model selection throughput at scale without raising resource costs or sacrificing reproducibility or accuracy. Cerebro uses a new parallel deep learning training strategy called model hopper parallelism. It hybridizes task- and data parallelism to mitigate the cons of these prior paradigms and offer the best of both worlds. Experiments on large ML benchmark datasets showed that Cerebro provides 3x to 10x runtime savings relative to data-parallel systems like Horovod and Parameter Server and up to 8x memory/storage savings or up to 100x network savings relative to task-parallel systems. Cerebro also supports heterogeneous resources and fault tolerance.

Bringing DL to data system-resident data. Deep learning's popularity is not limited to ML researchers; many enterprises and businesses also consider adopting deep learning for their data analytics applications. Large business-critical datasets in such settings typically reside in RDBMSs or other parallel data systems. In the work of Cerebro above, I explored the landscape of standalone deep learning model selection systems and proposed a new parallelism strategy. However, it was unclear if the parallelism and scheduling could be incorporated into existing infrastructures of data management systems. In Cerebro-DS [9], I characterized the particular suitability of Cerebro on data systems. To bring the novel model hopper parallelism approach to DB resident data, I showed that there was no single "best" approach and that an exciting tradeoff space of approaches exists. I explained four canonical approaches and built prototypes upon Greenplum Database. I compared them analytically on multiple criteria (e.g., runtime efficiency and ease of governance) with real large-scale deep learning workloads. The experiments and analyses showed that it was non-trivial to meet all practical desiderata well, and there was a Pareto frontier; for instance, some approaches are 3x-6x faster but fare worse on governance and portability. These results and insights can help DBMS and cloud vendors design better deep learning support for DB users.

Bridging the gap between graph analytical systems and GNN workloads. Moving on from the common IID data, I looked at the rapidly growing area of Graph Neural Networks (GNN) on graph data. The complexity of GNN training and scalability challenges has also sparked interest from the ML systems community, with efforts to build systems that provide higher efficiency, better memory management, and schemes to reduce costs. However, many of these systems reinvent the wheel by rediscovering years of research and development on advanced graph-parallel data systems. Further, they often couple the scalability challenges of graph data processing with those of GNN training, resulting in entangled complex problems and systems that need help to handle either scalability challenge. Lotan [7] is a novel and highly scalable data system for full-batch GNN training with a clean decoupling of graph and neural network at its core. With this decoupling, Lotan can achieve high scalability and bridge existing graph data systems and deep learning frameworks. Lotan offers a series of technical innovations, including execution plan rewriting, highly-efficient data movement between systems, a GNN-centric graph partitioning method and the corresponding gradients backpropagation scheme, and GNN model batching. Using real large-scale GNN workloads, I demonstrated the system's capability to train large GNN models that prior art, even from industry labs, crashed on. The system can surpass the training throughput of state-of-art systems such as DistDGL and AliGraph by over 40x and beat a naively implemented in-data-system GNN training framework by 76x. Lotan can increase efficiency for existing workloads and open new possibilities for future GNN algorithmic research.

System for unbounded vocabulary querying workloads on video and image data Video data is sometimes dubbed as "fast data", characterized by their sheer volume and the requirement of real-time responses for many applications such as video monitoring. Using deep learning methods for these applications incurs high computational costs and inference latency. The prior art has studied how to improve system efficiency. Nevertheless, they primarily focus on small "closed world" prediction vocabularies, even though many surveillance security and traffic analytics applications have an ever-growing set of target entities. I call this the "unbounded vocabulary" issue, which is a crucial bottleneck for emerging video monitoring applications. I presented the first data system for tackling this problem for video querying, Panorama [8]. The design philosophy is to build a unified and domain-agnostic system that lets application users generalize to unbounded vocabularies in an out-of-the-box manner without tedious manual re-training. To this end, I synthesized and innovated upon an array of techniques from the literature of ML, vision, databases, and multimedia systems to devise a new system architecture. I also presented designs to ensure Panorama had high inference efficiency. Experiments with multiple real-world datasets showed that Panorama could achieve between 2x to 20x higher efficiency than baseline approaches on in-vocabulary queries while still yielding comparable accuracy and also generalizing well to unbounded vocabularies.

Future Research

ML Systems research is a relatively new domain, and many opportunities exist. ML-based applications will be ubiquitous eventually, and there are destined to be new challenges. We are transitioning from segmented and often ad-hoc solutions to fully-fledged systems forming the backbones of the next wave of technological innovation. I choose to conduct future research aligned with and extrapolating from my current experiences and remain at the frontline during this postdoc.

From homogenous to heterogenous ML systems. As ML/AI research grows in complexity and because the world is heterogeneous and multimodal, highly heterogenous environments will only become more common, where distinct ML model architectures, multiple data storages, diversified data modalities, and various computational resources are all involved. Such complexities must be abstracted away from the user for easy adoption. I imagine a poly-ML system like the polystore systems in data system research. There are many system problems to expect: first, logical plan optimization of multiple correlated

workloads such as transfer learning, neural architecture search, and AutoML. Second, scheduling and coordination of heterogenous computational graph operators. Third, resource management of heterogenous physical clusters. Can we build a platform to facilitate multimodal ML research? Can we develop a set of abstractions and logical operators to express most of the said research? Can we allow the user to easily manipulate the data and experiment with their models without worrying about the underlying heterogeneous parallelism, data storage, and physical computational resources?

From ML frameworks to distributed ML data systems. The future of ML systems is distributed, as the performance of a single machine has physical limitations. However, today's practitioners still predominantly rely on single machines, although they are offered data-, model-, and other more advanced parallelisms. The adoption of these parallel techniques is not ideal, despite the tremendous amount of effort made. One of the major culprits behind the under-utilization of distributed processing is the systems' common lack of support for distributed storage, querying, and data computation. In the world of SQL and MapReduce, the users do not write a script and execute it on every machine; they code directly declaratively so that their program is readily parallelized. However, a common pattern for ML systems is that they try to parallelize and rewrite the users' single-node script. Instead of writing a single-node ML program with single-node operators and then trying to compile and distributedly execute it, can we offer distributed counterparts of the ML operators directly to the user?

From optimizing costs to enabling innovations. So far, ML Systems research has been primarily developed in parallel with ML algorithms research. The trend is shifting to some extent, but in general, it was the system researchers trying to catch up with the latest innovations from the ML community and building systems for their emerging use cases. This pattern usually results in a lag behind the frontline of ML research. In an era of fast-paced innovations, it is hard to predict what comes next, and the present challenges may be left behind by the next generation of models and workloads. To get ahead of this curve, I aspire to focus on the applications of my future ML Systems to facilitate ML/AI research. Examples include applications from economics with satellite images, time-sequence, and video data-aided public health/behavior studies and various GNN applications from computational neuroscience, material engineering, and chemistry. I look forward to collaborating with ML researchers and enabling model architectures that were impossible with limited computational resources. On the other hand, there are opportunities to design ML system-aware models. Can we build model architectures designed so that their data dependencies, data access, and communication patterns are easily parallelized and well-optimized by the underlying system? The model can even assume data locality, cache, and sparsity. Can we abstract and summarize the principles of scalable model architecture design? Can we build model architecture search systems that, besides accuracy, also optimize for runtime efficiency and scalability?

References

[1] MADlib Model Selection. https://madlib.apache.org/docs/latest/group__grp__mdl.html.

[2] Resource Efficient Deep Learning Model Selection on Apache Spark. https://www.databricks.com/session_na20/resource-efficient-deep-learning-model-selection-on-apache-spark.

[3] A. Kumar, S. Nakandala, **Yuhao Zhang**, S. Li, A. Gemawat, and K. Nagrecha. (Vision paper) Cerebro: A Layered Data Platform for Scalable Deep Learning. In *CIDR*. www.cidrdb.org, 2021.

[4] A. K. S. Nakandala and **Yuhao Zhang**. Some damaging delusions of deep learning practice (and how to avoid them). In *KDD Deep Learning Day 2021*, 2021.

[5] S. Nakandala, **Yuhao Zhang**, and A. Kumar. Cerebro: Efficient and reproducible model selection on deep learning systems. In *DEEM@SIGMOD*, pages 6:1–6:4. ACM, 2019.

[6] S. Nakandala, **Yuhao Zhang**, and A. Kumar. Cerebro: A Data System for Optimized Deep Learning Model Selection. *Proc. VLDB Endow.*, 13(11):2159–2173, 2020.

[7] **Yuhao Zhang** and A. Kumar. Lotan: A Highly Scalable System for GNN Training via the Separation of Graph and Neural Network. *Under Minor Revision at VLDB 2023*.

[8] **Yuhao Zhang** and A. Kumar. Panorama: A data system for unbounded vocabulary querying over video. *Proc. VLDB Endow.*, 13(4):477–491, 2019.

[9] **Yuhao Zhang**, F. Mcquillan, N. Jayaram, N. Kak, E. Khanna, O. Kislal, D. Valdano, and A. Kumar. Distributed Deep Learning on Data Systems: A Comparative Analysis of Approaches. *Proc. VLDB Endow.*, 14(10):1769–1782, 2021.