RESEARCH STATEMENT

Quality-Performance Tradeoffs in Analytic and Machine Learning Workloads

Yongjoo Park (https://yongjoopark.com)

My research interest is building systems for fast data analytics and machine learning (ML). These areas of focus inspire much influential research and investment: over the last five years, Walmart has spent millions of dollars on its clusters, and last year, Ford announced a $200 million investment on its data centers. However, the computational burden of performing data analytics and ML is daunting to those at enterprises and startups alike. I have had numerous opportunities to speak with executives and data analysts at Ford, Walmart, Dunnhumby, and several startups; they all expressed a common wish to lower their cluster costs and query latencies. To mitigate the expensive computing costs and long latencies, many data analysts are *willing to accept a slight reduction in result quality in exchange for substantially faster performance.*

However, exploiting these quality-performance tradeoffs is challenging. In most data analytics and ML workloads, the tradeoff relationships are *non-linear* and *data-/workloads-dependent.* To provide proper quality-guarantees on results, we must obtain *mathematical understanding* of these tradeoff relationships. Based on these relationships, we must also build systems that can adaptively adjust their quality-guarantee mechanisms according to the types of workloads and the distribution of data. In addition, these adaptive mechanisms must be highly efficient to minimize overall latencies. Building these systems requires a thorough understanding of both statistical theories and systems.

In my research experience, I have combined **rigorous statistical theories and systems understanding** to build *fast, quality-guaranteed* data analytics and ML systems. As part of my research, I have first-authored five research papers in premier database conferences (e.g., SIGMOD, VLDB). Part of this work was awarded **2018 ACM SIGMOD Jim Gray Dissertation Award runner-up**. Most of my research has been open-sourced. In particular, one of my systems, VerdictDB, has been **adopted by Walmart for its data analytics and by Digital2Go for its cloud platform.** VerdictDB is also tested by several other companies (e.g., Alibaba, Tableau) and is studied by students and researchers at other institutions.

My research can be considered in (not mutually exclusive) three categories:

1. **Data Analytics**: aggregation (VerdictDB) [3, 7], visualization (VAS) [6], searching (NSH) [5]
2. **Machine Learning**: maximum likelihood estimation (BlinkML) [2, 8]
3. **Self-Learning Analytics**: aggregation (Database Learning) [9], selectivity estimation (QuickSel) [10]

Data Analytics

I have applied quality-performance tradeoffs to three types of workloads: aggregation, visualization, and searching.

Aggregation Despite decades of research, *approximate query processing* (AQP)—trading off accuracy for faster query answers—has had little industry adoption. A few available AQP engines are each tied to a specific platform, and thus, require users to abandon their existing systems—an unrealistic assumption given the infancy of those AQP systems.

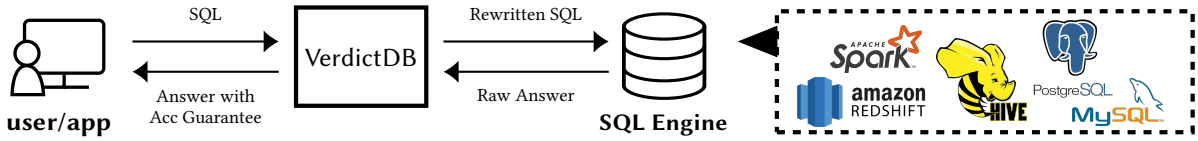


Figure 1: Architecture of VerdictDB. VerdictDB offers a speed-accuracy tradeoff on top of any existing SQL engine.

To close this ever-present gap between the AQP research in academia and the industry-standard database systems, we started an open-source project [1], called VerdictDB, which introduces a completely new architecture. That is, instead of bringing AQP techniques *into* database systems, VerdictDB offers AQP *on top of* existing database systems. Specifically, VerdictDB is a thin layer between the user and any off-the-shelf SQL engine; VerdictDB relies on SQL rewriting to quickly compute accuracy-guaranteed aggregates (see Figure 1). VerdictDB addresses the challenge of *platform-independent accuracy-guarantees* by proposing a highly-efficient, SQL-friendly error-estimation technique, called *variational subsampling*. As a result, VerdictDB could offer 171× speedup (18.45× on average) for a variety of existing engines, such as Amazon Redshift, Apache Spark SQL, and Apache Impala, while incurring less than 2.6% relative error.

Visualization VAS [6] answers the following question: what is the optimal subset of data if we want to maximize the overall quality of visualization? To answer this question, VAS formulates an optimization problem, and solves the problem with an efficient approximation algorithm.

Searching NSH [5] proposes a new algorithm for approximate k-nearest neighbor (kNN) searches. NSH relies on an intuitive idea: for kNN, the distances between close items must be captured more accurately than the distances between other items.

Machine Learning

A sample of data is widely used in practice to reduce ML training time. However, most practitioners are not able to precisely capture the effect of sampling on the quality of their model. Most previous work on accuracy-guaranteed ML either targets a specific type of models or requires data-sensitive pre-processing.

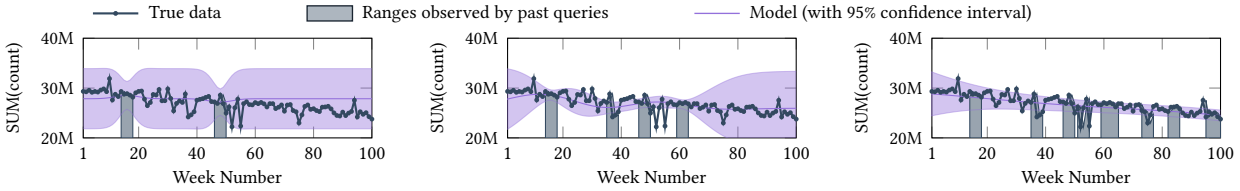
To offer formal guarantees to these practical operations, we developed BlinkML [8], a system for *fast* ML training with *probabilistic accuracy guarantees*. BlinkML supports any models that rely on maximum likelihood estimation for its training, including linear regression, logistic regression, max entropy classifier, and Probabilistic Principal Component Analysis. BlinkML's great efficiency stems from its novel mechanism that estimates the accuracy of a sample-trained model *without having to train it*. Thus, BlinkML automatically (and efficiently) determines the minimum sample size that is large enough to satisfy a user-requested accuracy guarantee. A BlinkML-trained model produces the same predictions as the full model (trained on the entire data) with high probability (say 95%). In addition, BlinkML relies on uniform random sampling; thus, no data-sensitive pre-processing is needed.

BlinkML achieves an impressive speed-accuracy tradeoff. BlinkML could train ML models 13.2×–629× faster while guaranteeing that its model produces the same predictions as the full model with 95% probability. Moreover, in feature engineering tasks [2], BlinkML could examine 320× more models compared to full model training.

Self-Learning Analytics

I have developed self-learning algorithms for approximate query processing (AQP) and selectivity estimation.

Approximate Query Processing Database learning (DBL) [4, 9] is the first AQP system that can produce increasingly more accurate answers as it processes more queries. DBL builds a non-parametric probabilistic model of the underlying data relying on the answers to past queries (see Figure 2). This probabilistic model is then used to improve the accuracy of new query answers. Processing more queries leads to a more accurate probabilistic model, which then leads to further accuracy improvements.



(a) After 2 Queries (b) After 4 Queries (c) After 8 Queries

Figure 2: An example of database learning's probabilistic model construction.

The key challenge in improving this accuracy is its computational overhead. Too large of an overhead defeats the purpose of AQP. DBL shows that an efficient, analytic inference is possible by using the probabilistic model built from second-order statistics (i.e., pairwise covariances between query answers). These statistics can be cheaply computed by inspecting the queries, not their answers. With this efficient mechanism, DBL could speed up AQP by 23.0×.

Selectivity Estimation The selectivity estimation is a key step in almost any cost-based query optimizer. Recently, we developed QuickSel [10], an ML-based selectivity estimation algorithm. QuickSel gradually refines the model of the underlying data by relying on the selectivities of past queries. The model can then be used for estimating the selectivity of a new query. Unlike database learning, QuickSel does not provide accuracy guarantees for its estimates; instead, QuickSel offers faster training and inference by employing a different model.

In our experiments, QuickSel delivered impressive performance. Compared to the previous work (i.e., adaptive histograms), QuickSel was 46.8%–75.3% more accurate. Compared to periodically updating histograms and samples, QuickSel was 57.3% and 91.1% more accurate, respectively.

Future Research

My future research will further explore this exciting opportunity—the combination of statistical theories and systems—for fast data analytics and ML workloads. This research can build systems that will quickly and automatically discover interesting patterns from data. The users of these systems will be able to gain insights almost instantly with minimal effort. To achieve this goal, I plan to pursue two specific topics in the near future: (1) automatic insights discovery and (2) automatic optimization for the cloud.

Automatic Insights Discovery from Big Data Discovering interesting patterns is computationally challenging. For the discovery, the following procedure is typically used. Many candidate statistics or ML models are first computed or trained. Among these candidates, a few top candidates are presented to users. Unfortunately, the cost of this process increases with the size of data. However, small errors in statistics computations or ML training may have little impact on the list of those top candidates. If small, fixed errors are permitted, the processing cost can be fixed regardless of the data size. Thus, even for extremely big data, we can quickly discover interesting patterns. Relying on this observation, I plan to initiate the AutoInsight project, which will study the following concrete problems:

1. Quality-Guaranteed Analytics for Almost All: Unlike previous work that mostly focuses on traditional aggregations or specific ML models, BlinkML unleashes the power of uniform random sampling for a wide-class of ML models. I want to extend this capability beyond that offered by VerdictDB or BlinkML. We can first target commonly used machine learning models (e.g., decision tree, Gaussian Process regression, support vector machines) and statistical tools (e.g., ANOVA, ARMA model, hypothesis testing), then further extend this capability to less-commonly used techniques.
2. Common Programming Patterns for Speed-Accuracy Tradeoff: Programming abstractions help us to organize our thought processes, reduce programming mistakes, and increase software maintainability. Moreover, analytics systems can exploit the abstraction for effectively scheduling its tasks, efficiently distributing jobs across multiple machines, and reducing communication overhead. However, designing good abstractions requires experiences and case studies. My experience in *quality-guaranteed systems* will be of benefit in defining simple but effective abstractions.
3. Low-Latency Distributed Engine: Distributed computational frameworks, such as Apache Spark, excel in handling long-running, deterministic jobs. However, this design choice may lead to sub-optimal efficiency for short-lived, stochastic workloads, which are common when speed-accuracy tradeoff is exploited. With speed-accuracy tradeoff, processing just a few million records often suffices. Processing a few million records takes only a few seconds using today's distributed engines. Second, stochastic workloads are common when we rely on random samples. The job may finish early if the intermediate answers are considered to be accurate enough. Alternative systems must support short-lived, stochastic workloads efficiently, respecting the common programming patterns for speed-accuracy tradeoff.

Automatic Optimization for the Cloud More data and analytics systems are moving to the cloud. Unlike on-premises clusters, the cloud allows greater opportunities for automatic optimizations. Specifically, the data and analytics systems in the cloud (e.g., Amazon EC2, Google Dataproc) can be assessed via commonly used API and access credentials. Given user authorizations, third-party tools can automatically manage those data for future ad-hoc analytics. Moreover, the automatic management can also exploit stored workload patterns. To fully exploit this exciting opportunity, I plan to initiate a new project called AutoCloud. AutoCloud's primary goal is the following:

1. Automatic Sample Provisioning: For speed-accuracy tradeoff, random samples are essential. The current VerdictDB design relies on pre-built samples, constructed upon user requests. However, AutoCloud will actively examine various data statistics and create appropriate samples in advance. These automatically built samples can serve both traditional aggregations (e.g., sum, count, avg) and ML workloads (e.g., regression and classification). Under-utilized samples should be automatically removed to save storage space.
2. Automatic Data Formats: Data formats have significant impact on performance. For instance, Parquet or ORC format excels for traditional analytics workloads, while Compressed Sparse Row Format excels for training ML models on high-dimensional data. By examining well-known datasets and common operations performed on them, we can learn the relationship and apply it for choosing the right format for arbitrary data.

Finally, my future research will continue to contribute to the community with open-source projects, online and offline discussions, and meetings. For example, VerdictDB is already open-sourced under Apache License, and is published to the Maven Central Repository and the Python Package Index. VerdictDB has received/reflected about 200 user requests through GitHub, maintains a Gitter channel, and provides online tutorials and example code.

References

[1] VerdictDB website. http://verdictdb.org/.

[2] M. R. Anderson, D. Antenucci, V. Bittorf, M. Burgess, M. J. Cafarella, A. Kumar, F. Niu, **Yongjoo Park**, C. Ré, and C. Zhang. Brainwash: A data system for feature engineering. *CIDR'13: The 8th biennial Conference on Innovative Data Systems Research*, Santa Cruz, USA, 2017. Full Research.

[3] W. He, **Yongjoo Park**, I. Hanafi, J. Yatvitskiy, and B. Mozafari. Demonstration of verdictdb, the platform-independent aqp system. *SIGMOD'18: ACM SIGMOD International Conference on the Management of Data*, Houston, USA, 2018. Demo.

[4] **Yongjoo Park**. Active database learning. *CIDR'17: The 8th biennial Conference on Innovative Data Systems Research*, Santa Cruz, USA, 2017. Abstract.

[5] **Yongjoo Park**, M. Cafarella, and B. Mozafari. Neighbor-sensitive hashing. *VLDB'16: 42nd International Conference on Very Large Data Bases*, New Delhi, India, 2016. Full Research.

[6] **Yongjoo Park**, M. Cafarella, and B. Mozafari. Visualization-aware sampling for very large databases. *ICDE'16: 32nd IEEE International Conference on Data Engineering*, Helsinki, Finland, 2016. Full Research.

[7] **Yongjoo Park**, B. Mozafari, J. Sorenson, and J. Wang. VerdictDB: universalizing approximate query processing. *SIGMOD'18: ACM SIGMOD International Conference on the Management of Data*, Houston, USA, 2018. Full Research.

[8] **Yongjoo Park**, J. Qing, X. Shen, and B. Mozafari. BlinkML: Efficient maximum likelihood estimation with probabilistic guarantees. *SIGMOD'19: ACM SIGMOD International Conference on the Management of Data*, Amsterdam, The Netherlands, 2019. Full Research.

[9] **Yongjoo Park**, A. S. Tajik, M. Cafarella, and B. Mozafari. Database Learning: Toward a database that becomes smarter every time. *SIGMOD'17: ACM SIGMOD International Conference on the Management of Data*, Chicago, USA, 2017. Full Research.

[10] **Yongjoo Park**, S. Zhong, and B. Mozafari. QuickSel: Quick selectivity learning with mixture models. *In Submission.* Full Research.