Active Database Learning

Yongjoo Park
University of Michigan, Ann Arbor
pyongjoo@umich.edu

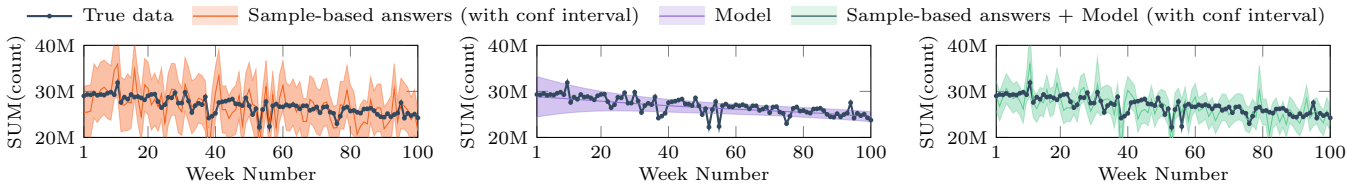

Figure 1: Benefits of using a statistical data model in approximate query processing context. Combining sample-based approximate answers (red on the left) and a statistical model (purple in the middle) produces more accurate answers (green on the right) than solely relying on samples.

Learning from Past Query Processing— In today's databases, the answers to past queries barely benefit processing future queries. The query answers and the work performed for processing queries—such as I/O and computations—are no longer used after returning query answers.[1]

Database Learning (DBL) [4] proposes to change this paradigm in an approximate query processing (AQP) context. DBL uses its knowledge acquired from past queries and their answers for speeding up future query processing. The more queries are processed, the smarter a system becomes; thus, the faster it can process new queries.

DBL is based on the observation that the answers to queries (even if they access different subsets of rows and columns) stem from the same underlying distribution (which produced the entire data). Therefore, an answer to each query reveals a piece of information about this underlying distribution, which can be used to construct a concise statistical model of the distribution.

Using a statistical model could bring a significant performance benefits. In an ideal case where the model precisely captures the underlying distribution, we could answer queries by analytically evaluating the model, instead of reading and processing terabytes of raw data. Even an imprecise model can still be beneficial; for instance, one can use a small sample of the entire data to quickly produce a sample-based answer [1], which can then be combined with the model to produce a more accurate approximate answer.

Our prototype [4]—that uses the *maximum entropy principle* for model construction and SparkSQL for distributed computations—showed 73.7% support of real-world query workloads and 23.0 times of possible speedup compared to existing AQP systems (at the same accuracy level).

[1] A few works, such as view selection [2] and adaptive indexing [3], use past *queries*, but they still do not exploit query *answers*. See [4] for more discussion.

A Step Further: Active Learning— DBL's weakness is its limited source of knowledge, i.e., past queries and their answers. That is, it misses the possible chance of model refinement through the active examination of the data.

Active Database Learning (ADL) is our new initiative aiming to address the limitation of DBL. It extends the *passive* behavior of DBL by adding the capability of *actively* analyzing the data; thus, improving a statistical model even without newly processed queries. Based on past query workloads and possible hints from users, ADL seeks best columns/rows such that, if a statistical model is built and improved for, can bring the largest performance benefits for future query processing. ADL's active model refinement may be performed offline or between query processing. Similar to DBL, ADL's model is useful for improving the quality of AQP, as illustrated in Figure 1.

For small and simple datasets, ADL might boil down to materialized views (MV), since (most of) aggregations can be exactly precomputed. However, ADL's model has advantages compared to MV for big and complex data. First, ADL's analytic expression can succinctly model the tables with many columns and many unique attribute values. However, MV must store a value for every unique combination, easily suffering from the *curse of dimensionality*. Second, a statistical model can be built using (already-available-for-AQP) samples of the entire data, avoiding hours of offline processing. Constructing MV for big data may require hours of downtime even for distributed database systems.

Challenges for Active Database Learning— We need to overcome the following challenges for practical ADL. First, we need to find a balance between (i) the accuracy stemming from a more expressive and accurate model and (ii) the speed stemming from a simple and less accurate model. Second, we need a solid theory for determining the most beneficial columns/rows for model construction, possibly based on past query workloads, hints from users, correlations between columns, etc. Third, ADL should evolve its model according to dynamically changing query workload. This is more desirable when space limit is placed.

We believe ADL can be a primary OLAP technique in AQP context, as an advance over the popular materialized views for (more traditional) existing database systems.

REFERENCES

[1] S. Agarwal, B. Mozafari, A. Panda, H. Milner, S. Madden, and I. Stoica. Blinkdb: queries with bounded errors and bounded response times on very large data. In *EuroSys*, 2013.

[2] S. Agrawal, S. Chaudhuri, and V. R. Narasayya. Automated selection of materialized views and indexes in sql databases. In *VLDB*, 2000.

[3] S. Idreos, M. L. Kersten, S. Manegold, et al. Database cracking. In *CIDR*, 2007.

[4] Y. Park, A. S. Tajik, M. Cafarella, and B. Mozafari. Database learning: Toward a database that becomes smarter every time. http://yongjoopark.com/resources/dblt-vldb17.pdf, 2016.