Machine Learning for Databases

Guoliang Li
Tsinghua University, China
liguoliang@tsinghua.edu.cn

Xuanhe Zhou
Tsinghua University, China
zhouxuan19@mails.tsinghua.edu.cn

Lei Cao
MIT, USA
lcao@csail.mit.edu

ABSTRACT

Machine learning techniques have been proposed to optimize the databases. For example, traditional empirical database optimization techniques (e.g., cost estimation, join order selection, knob tuning) cannot meet the high-performance requirement for large-scale database instances, various applications and diversified users, especially on the cloud. Fortunately, machine learning based techniques can alleviate this problem by judiciously learning the optimization strategy from historical data or explorations. In this tutorial, we categorize database tasks into three typical problems that can be optimized by different machine learning models, including (i) NP-hard problems (e.g., knob space exploration, index/view selection, partition-key recommendation for offline optimization; query rewrite, join order selection for online optimization), (ii) regression problems (e.g., cost/cardinality estimation, index/view benefit estimation, query latency prediction), and (iii) prediction problems (e.g., transaction scheduling, trend prediction). We review existing machine learning based techniques to address these problems and provide research challenges.

ACM Reference Format:
Guoliang Li, Xuanhe Zhou, and Lei Cao. 2021. Machine Learning for Databases. In *The First International Conference on AI-ML-Systems (AIMLSystems '21), October 21–23, 2021, Bangalore, India.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3486001.3486248

1 MOTIVATION

Traditional database optimization techniques are based on empirical methodologies and specifications, and require heavy human involvement to tune and maintain the databases. Thus existing empirical techniques cannot meet the high-performance requirement for growing applications, large-scale database instances, and diversified users, especially on the cloud. Fortunately, learning-based techniques can alleviate this problem. For instance, deep learning can improve the quality of cost estimation [4, 19, 20, 23], reinforcement learning can be used to optimize join order selection [13, 16, 22, 25], and deep reinforcement learning can be used to tune database knobs [2, 12, 28, 30].

However, there are several common challenges in applying machine learning techniques in classic database problems. First, database is a complex system with various workload and runtime characters (e.g., read/write ratio, buffer hit rate). Moreover, some characters are not available in real online scenarios (e.g., query logs). So the first challenge is how to select and characterize effective features for the machine learning models to make decisions. Second, there are various ML models like supervised/unsupervised learning models, which can be further divided into classic machine learning and deep learning. These models fit different problems. For example, deep learning can work well to regress for high-dimension table data, while RL is good at decision-making. So the second challenge is how to select proper machine learning models to solve different database problems. Third, there are various rigorous requirements for different database scenarios. For example, query rewrite requires extremely low rewrite overhead (e.g., milliseconds) and it is unaffordable to take hours applying reinforcement learning to select rewrite rules for single queries. Moreover, other problems like performance robustness and scenario migration are also challenging. In this tutorial, we will separately discuss existing learning-based works and how they solve one or some of above challenges, and provide some future research directions.

2 TUTORIAL OVERVIEW

(1) Background and Motivation (10min). We first introduce the background and motivation of learning based techniques.

(2) Machine learning for Optimizing NP Problems (30min). Many database optimization problems can be modeled as exploring the optimal solutions for NP-hard problems. Machine learning based techniques can efficiently learn exploration models to replace heuristic algorithms and utilize the learned models to optimize the NP-hard problems [12, 25, 26, 28, 29]. We classify them into two cases based on exploring the solutions online or offline.

(2.1) Offline Optimizing NP-hard Problems [1, 7–9, 12, 15, 26, 28, 29]. We first train a model and then use the model to optimize the NP-hard problems. The model can be fine-tuned during optimizing the NP-hard problems, and thus may be heavy.

(2.1.1) *Knob space exploration* [1, 2, 9, 12, 28, 29]. Recently learned tuning methods are proposed to improve the tuning performance or resource utilization. There are three types of models. (i) Gradient-based models [1, 9, 29] like Gaussian Process are widely used to explore local-optimal knob settings based on gradient descent; (ii) Similarly, deep-learning models [21] estimate the performance of selected knob settings. They take selected knobs and internal metrics as input and output the predicted response time; (iii) RL-based methods [12, 28] take knob tuning as a trail-and-error procedure, where the agent inputs tuning factors (e.g., system metrics, queries), outputs proper knobs, and learns from the tuning results.

(2.1.2) *Index/View selection* [3, 5, 8, 10, 26]. Database indexes and views are fairly crucial to achieve high performance. But it is expensive to recommend and build appropriate indexes/views with a large number of columns/tables and queries/subqueries. Hence, there are some reinforcement-learning models that recommend indexes [8, 15] and materialized views [5, 26].

(2.1.3) *Database Partition* [7]. Traditional methods heuristically select columns as partition keys (single column mostly) and cannot balance between load balancing and access efficiency. Some

AIMLSystems '21, October 21–23, 2021, Bangalore, India
© 2021 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-8594-7/21/10.
https://doi.org/10.1145/3486001.3486248

work [7] also utilizes reinforcement learning model to explore different partition keys.

(2.2) Online Optimizing NP-hard problems [16, 17, 22, 25, 30]. Some database problems (e.g., query rewrite, online plan optimization) have instant feedback requirements (e.g., optimizing within milliseconds). Hence, it cannot tolerate a long time to update a model. Hence, we want to select machine learning models that can (i) adaptively learn the policy during optimization and (ii) balance between performance and efficiency.

(2.2.1) *SQL rewriter* [13, 31]. Rule-based query rewriting methods may not find high-quality rules and appropriate rule order. Instead, learned tree search algorithms (e.g., MCTS) can be used to select the appropriate rules and apply rules in a good order.

(2.2.2) *Join order selection* [16, 17, 22, 25]. A SQL query may have millions, even billions of possible plans and it is very important to efficiently find a good plan. Traditional heuristics methods cannot find optimal plans for dozens of tables and dynamic programming is costly to explore the huge plan space. Thus there are some deep reinforcement learning based methods [16, 17, 22, 25] that automatically select good plans.

(3) Machine learning for Regression Problems (30min). Many database problems can be modeled as a regression problem. Cardinality estimation aims to estimate the cardinality of a query and a regression model (e.g., deep learning model) can be used [4, 6, 19, 20, 23, 24]. Index/view benefit estimation aims to estimate the benefit of creating an index (or a view), and a regression model can be used to estimate the benefit [3, 10]. Latency prediction aims to estimate the execution time of executing a query and a regression model can be used to estimate the performance based on query and concurrency features [18, 32].

(4) Machine learning for Prediction Problems (10min). It is also vital to proactively optimize the database by predicting incoming queries. These prediction problems identify the temporal workload patterns and rearrange query execution to maximize the query performance or resource usage. And there are some machine learning methods for such prediction problems, e.g., cluster-based algorithms for trend prediction [14], reinforcement learning for workload scheduling [27].

(5) Challenges and Opportunities (10min). Finally, we provide research challenges and opportunities.

Remark. This is an invited tutorial based on our tutorial at VLDB [11]. We will provide more deep analysis on existing works.

3 BIOGRAPHY

Guoliang Li is a professor in the Department of Computer Science, Tsinghua University. His research interests mainly include data cleaning and integration and machine learning for databases.

Xuanhe Zhou is currently a PhD student in the Department of Computer Science, Tsinghua University. His research interests lie in AI/ML for data management.

Lei Cao is a Postdoc Associate at MIT CSAIL, working with Prof. Samuel Madden and Prof. Michael Stonebraker.

ACKNOWLEDGMENTS

This paper was supported by NSF of China (61925205, 61632016), Huawei, TAL education, and Beijing National Research Center for Information Science and Technology (BNRist).

REFERENCES

[1] Dana Van Aken, Andrew Pavlo, Geoffrey J. Gordon, and Bohan Zhang. 2017. Automatic Database Management System Tuning Through Large-scale Machine Learning. In *SIGMOD 2017*. 1009–1024. https://doi.org/10.1145/3035918.3064029

[2] Dana Van Aken, Dongsheng Yang, Sebastien Brillard, Ari Fiorino, Bohan Zhang, Christian Billian, and Andrew Pavlo. 2021. An Inquiry into Machine Learning-based Automatic Configuration Tuning Services on Real-World Database Management Systems. *VLDB* 14, 7 (2021), 1241–1253.

[3] Bailu Ding, Sudipto Das, Ryan Marcus, Wentao Wu, Surajit Chaudhuri, and Vivek R. Narasayya. 2019. AI Meets AI: Leveraging Query Executions to Improve Index Recommendations. In *SIGMOD*. 1241–1258.

[4] Anshuman Dutt, Chi Wang, Azade Nazi, and et al. 2019. Selectivity Estimation for Range Predicates using Lightweight Models. *VLDB* 12, 9 (2019), 1044–1057.

[5] Yue Han, Guoliang Li, Haitao Yuan, and Ji Sun. 2021. An Autonomous Materialized View Management System with Deep Reinforcement Learning. In *ICDE*.

[6] Shohedul Hasan, Saravanan Thirumuruganathan, Jees Augustine, Nick Koudas, and Gautam Das. 2020. Deep Learning Models for Selectivity Estimation of Multi-Attribute Queries. In *SIGMOD*. 1035–1050.

[7] Benjamin Hilprecht, Carsten Binnig, and Uwe Röhm. 2020. Learning a Partitioning Advisor for Cloud Databases. In *SIGMOD*. 143–157.

[8] Jan Kossmann, Stefan Halfpap, Marcel Jankrift, and Rainer Schlosser. 2020. Magic mirror in my hand, which is the best in the land? An Experimental Evaluation of Index Selection Algorithms. *VLDB* 13, 11 (2020), 2382–2395.

[9] Mayuresh Kunjir and Shivnath Babu. 2020. Black or White? How to Develop an AutoTuner for Memory-based Analytics. In *SIGMOD*. 1667–1683.

[10] Hai Lan, Zhifeng Bao, and Yuwei Peng. 2020. An Index Advisor Using Deep Reinforcement Learning. In *CIKM*. 2105–2108.

[11] Guoliang Li, Xuanhe Zhou, and Lei Cao. 2021. Machine Learning for Databases. *Proc. VLDB Endow.* 14, 12 (2021), 3190–3193.

[12] Guoliang Li, Xuanhe Zhou, and et al. 2019. QTune: A Query-Aware Database Tuning System with Deep Reinforcement Learning. *VLDB* (2019), 2118–2130.

[13] Guoliang Li, Xuanhe Zhou, Sun Ji, Xiang Yu, Yue Han, Lianyuan Jin, Wenbo Li, and et al. 2021. openGauss: An Autonomous Database System. *VLDB* (2021).

[14] Lin Ma, Dana Van Aken, Ahmed Hefny, Gustavo Mezerhane, Andrew Pavlo, and Geoffrey J. Gordon. 2018. Query-based Workload Forecasting for Self-Driving Database Management Systems. In *SIGMOD*. 631–645.

[15] Lin Ma, Bailu Ding, Sudipto Das, and et al. 2020. Active Learning for ML Enhanced Database Systems. In *SIGMOD*. 175–191.

[16] Ryan Marcus and Olga Papaemmanouil. 2018. Deep Reinforcement Learning for Join Order Enumeration. In *SIGMOD 2018*. 3:1–3:4. https://doi.org/10.1145/3211954.3211957

[17] Ryan C. Marcus, Parimarjan Negi, Hongzi Mao, Chi Zhang, Mohammad Alizadeh, Tim Kraska, Olga Papaemmanouil, and Nesime Tatbul. 2019. Neo: A Learned Query Optimizer. *PVLDB* (2019), 1705–1718.

[18] Ryan C. Marcus and Olga Papaemmanouil. 2019. Plan-Structured Deep Neural Network Models for Query Performance Prediction. *VLDB* (2019), 1733–1746.

[19] Ji Sun and Guoliang Li. 2019. An End-to-End Learning-based Cost Estimator. *PVLDB* 13, 3 (2019), 307–319. https://doi.org/10.14778/3368289.3368296

[20] Ji Sun, Guoliang Li, and Nan Tang. 2021. Learned Cardinality Estimation for Similarity Queries. In *SIGMOD*. 1745–1757.

[21] Jian Tan, Tieying Zhang, Feifei Li, Jie Chen, Qixing Zheng, Ping Zhang, Honglin Qiao, Yue Shi, Wei Cao, and Rui Zhang. 2019. iBTune: Individualized Buffer Tuning for Large-scale Cloud Databases. *PVLDB* (2019), 1221–1234.

[22] Immanuel Trummer, Junxiong Wang, Deepak Maram, Samuel Moseley, Saehan Jo, and Joseph Antonakakis. 2019. SkinnerDB: Regret-Bounded Query Evaluation via Reinforcement Learning. In *SIGMOD 2019*. 1153–1170.

[23] Peizhi Wu and Gao Cong. 2021. A Unified Deep Model of Learning from both Data and Queries for Cardinality Estimation. In *SIGMOD*. 2009–2022.

[24] Zongheng Yang, Eric Liang, Amog Kamsetty, Chenggang Wu, and et al. 2019. Deep Unsupervised Cardinality Estimation. *VLDB* 13, 3 (2019), 279–292.

[25] Xiang Yu, Guoliang Li, Chengliang chai, and Nan Tang. 2019. Reinforcement Learning with Tree-LSTM for Join Order Selection. In *ICDE 2020*. 196–207.

[26] Haitao Yuan, Guoliang Li, Ling Feng, and et al. 2020. Automatic View Generation with Deep Learning and Reinforcement Learning. In *ICDE*. 1501–1512.

[27] Chi Zhang, Ryan Marcus, Anat Kleiman, and Olga Papaemmanouil. 2020. Buffer Pool Aware Query Scheduling via Deep Reinforcement Learning. *CoRR* abs/2007.10568 (2020). arXiv:2007.10568

[28] Ji Zhang, Yu Liu, Ke Zhou, Guoliang Li, and et al. 2019. An End-to-End Automatic Cloud Database Tuning System Using Deep Reinforcement Learning. In *SIGMOD*.

[29] Xinyi Zhang, Hong Wu, Zhuo Chang, and et al. 2021. ResTune: Resource Oriented Tuning Boosted by Meta-Learning for Cloud Databases. In *SIGMOD*. 2102–2114.

[30] Xuanhe Zhou, Lianyuan Jin, Sun Ji, and et al. 2021. DBMind: A Self-Driving Platform in openGauss. *VLDB* (2021).

[31] Xuanhe Zhou, Guoliang Li, Chengliang Chai, and Jianhua Feng. 2022. A Learned Query Rewrite System using Monte Carlo Tree Search. *PVLDB* 1, 15 (2022).

[32] Xuanhe Zhou, Ji Sun, Guoliang Li, and et al. 2020. Query Performance Prediction for Concurrent Queries using Graph Embedding. *VLDB* (2020), 1416–1428.