Proceedings of CONF-SPML 2026 Symposium: The 2nd Neural Computing and Applications Workshop 2025
DOI: 10.54254/2755-2721/2026.TJ29634

A Comprehensive Investigation on Database Knob Tuning Using Machine Learning

Rongpu Shi

Alliance Manchester Business School, University of Manchester, Manchester, United Kingdom
rongpu.shi@student.manchester.ac.uk

Abstract. Knob tuning is an important process in database management, where a knob refers to a combination of configuration parameters, such as the memory allocation strategy, which significantly impact database performance, like query speed. To accelerate this process and maximise its automation, more and more researchers have proposed various machine learning methods. Based on the literature reviewed, this paper categorises these methods into three types, provides detailed descriptions of them, and identifies their key innovations. Firstly, Bayesian Optimisation (BO)-based methods, represented by OtterTune, establish probabilistic models using Gaussian Processes to efficiently search for optimal configurations. Next, Reinforcement Learning (RL)-based approaches, represented by CDBTune, train RL agents to find the optimal strategy through trial and error. The last category is Large Language Model (LLM)-based knob tuning, represented by E2ETune. Through further research and analysis, this paper also identifies some challenges in this field, such as the very common exploration-exploitation dilemma in RL, the instability of performance caused by multiple factors, the feasibility of recommended configurations, and the widely recognised cost issue. In response, some potential future improvement directions are proposed: Using confidence intervals or probability distributions to solve the exploration-exploitation dilemma of RL; Providing more comprehensive information, such as system hardware parameters, to agent to minimize the impact of external factors; Adding constraints or training ML models to verify configuration feasibility, as attempted by OnlineTune; Adopting cloud computing to significantly reduce costs.

Keywords: Database Knob Tuning, Machine Learning, Bayesian Optimisation, Reinforcement Learning, Large Language Models

1. Introduction

Nowadays, with the trend of high digitalisation, no organisation can operate without databases. They store various types of organisational information, such as employee data, which can be quickly retrieved when needed. The term database first appeared in the early 1960s, indicating that information stored in computers can be conceptualised, structured, and manipulated independent [1]. The first Database Management System (DBMS) was born in 1961, which was the Integrated Data Store (IDS) system introduced by Charles Bachman [1]. It is a software system used for managing databases. IDS, along with later DBMSs like IBM's Information Management System (IMS), are all

© 2025 The Authors. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (https://creativecommons.org/licenses/by/4.0/).

DOI: 10.54254/2755-2721/2026.TJ29634

closely integrated with programs and data, resulting in applications relying heavily on data storage order, index structures, and access paths [2].

In 1970, E. F. Codd proposed the relational model to improve this [2]. Then, IBM developed the Structured Query Language (SQL) to query relational databases [3]. Since Oracle launched its first commercial relational database based on SQL in 1979 [4], major companies have begun to develop their own systems. However, as the data volume increased exponentially, traditional databases showed some limitations, including poor scalability, decreased performance, high costs, and limited data processing capacity [5]. In this case, DBMS requires transformation by combining advanced technologies to promote more intelligent development, particularly with the integration of Artificial Intelligence (AI).

Since the introduction of the Transformer architecture in 2017 [6], AI has become extremely popular, and it has made significant progress in database management in different ways. For example, Knob tuning. In a DMBS, knobs refer to a combination of configuration parameters, such as the memory allocation, caching polices, index usage, and log writing strategies [7]. These parameters significantly impact different performance indicators like query speed and resource utilisation. That is why knob tuning is critical in DBMS. This process requires balancing interactions between different knobs, finding the optimal parameter combination, and taking cost into consideration. In the early days, database administrators used heuristic rules for tuning, which were based on human experience and best practices [8].

However, with the increasing size and complexity of databases, manual tuning may not perform well [7]. As a result, a series of machine learning-based tuning methods has been developed. In 2017, OtterTune used historical execution data to train regression models to predict system performance under different parameter settings [7]. In 2018, CDBTune modelled the parameter tuning process as a decision-making process [9], enabling the use of reinforcement learning methods, where an agent could try various parameter combinations to learn the optimal strategy. More recently, in 2024, E2ETune further strengthened this process by using the Large Language Model (LLM) to accelerate the optimiser tuning process [10].

As more and more machine learning frameworks for database knob tuning are developed, a comprehensive review of this area is necessary. This paper will be organised based on the following topics. Firstly, the methodology section will review how researchers have applied different machine learning techniques to automate database knob tuning. Next, the discussion section will discuss some current challenges and limitations faced by AI in knob tuning of the DBMS, as well as potential future development directions. Finally, the conclusion will summarise the findings and provide conclusions drawn from previous sections.

2. Method

This section introduces several machine learning methods for knob tuning proposed by different researchers. Based on the literature reviewed in this paper, these methods can be divided into three categories based on the machine learning approach used: Bayesian Optimisation (BO)-based, Reinforcement Learning (RL)-based, and Large Language Model (LLM)-based knob tuning methods.

2.1. Bayesian Optimisation-based knob tuning

Bayesian Optimisation (BO) seeks the optimal configuration by building a probabilistic model. iTuned is one of the representatives, which uses a Gaussian Process (GP) to model the performance

DOI: 10.54254/2755-2721/2026.TJ29634

function and uses the Expected Improvement (EI) criterion to select the configuration [11]. This allows the balance between exploring the unknown and using previous knowledge.

Following iTuned, more and more researchers further explored BO methods. The most famous one is the OtterTune, proposed by Van Aken et al. in 2017, which consists of a Controller (connecting databases) and a Tuning Manager (building ML models and managing historical data) [7]. Built on the GP, OtterTune's innovation lies in its transferable mechanism. Most of the tuning tools at that time started from scratch every time and could not reuse the past tuning experience [7]. OtterTune changes this by identifying important knobs from historical tuning data and using workload modelling to match current workloads with similar historical ones. This enables the reuse of previous models for new tuning tasks, helping the system reach optimal performance more quickly. Experiments on MySQL have shown that OtterTune can reduce latency by 58%-94% compared to its default settings [12]. Its transferability opened a new path for tuning experience reusing.

Later studies continued to enhance BO methods for high-dimensional tuning problems. For example, ResTune in 2021 used multiple historical tuning models to create an integrated model, which combined multiple GPs for simultaneous optimization [13]. It achieves dimensionality reduction and excellent performance in resource control. Similarly, LlamaTune also uses sharding and random projections to map high-dimensional knob spaces into low-dimensional ones for dimensionality reduction, achieving a throughput improvement of up to 21% [14].

2.2. Reinforcement learning-based knob tuning

A well-known representative of RL-based tuning is CDBTune, proposed by Zhang et al. in 2019. It uses Deep Deterministic Policy Gradient (DDPG) to find the best tuning strategy by trial-and-error with rewards [9]. The process consists of two stages. The first stage is offline training, where standard tools like Sysbench are used to generate simulated SQL workloads. After each execution, query (q), knob configuration (a), state (s), and performance metrics result (r) are recorded. Then, the DDPG model is trained using these generated samples to find the optimal knob combination. The second stage is online tuning, where the system collects the current workload, and uses the model trained in the offline phase to recommend knob configurations. After deploying and testing, the model will be updated based on the feedback. Samples generated during this phase are added to the experience pool for future training.

The innovation of CDBTune is that it is the first end-to-end RL knob tuning system [9]. Previous methods like OtterTune divided the tuning process into multiple stages, where each stage could generate wrong information [15]. In contrast, CDBTune uses deep RL to directly map states to actions, thereby reducing errors. Another advantage of RL is that it does not require a large amount of high-quality historical tuning data, which means the model can achieve good performance with fewer samples. Furthermore, the reward function of CDBTune is more comprehensive, as it considers both the performance improvement from the initial step and the previous one [9], making it more comprehensive to determine whether the model is improving.

Experiments show that CDBTune only takes 25 minutes for a tuning [9], which is one-tenth of the time taken by BestConfig (an earlier tuning tool) [16], and its throughput has also increased by an astonishing 128.66% compared to BestConfig [9]. It also performs well across 6 different cloud databases.

Building on CDBTune, researchers have also made many improvements. For example, in 2020, QTune converts SQL queries into feature vectors, allowing agents to better understand these queries [17]. Compared to CDBTune, QTune achieved a significant throughput improvement of 151.10%,

DOI: 10.54254/2755-2721/2026.TJ29634

with 66.15% less training time [17]. In addition, MMDTune in 2021 further extends RL tuning to multimodal databases and achieves cross-task migration by reusing the policy from a trained agent to initialise another task [18]. This greatly improves efficiency and reduces costs, achieving a 7.35% increase in throughput compared to CDBTune, while using 11.54% fewer resources [18].

2.3. Large Language Model-based knob tuning

With the rapid development of Large Language Models (LLMs), researchers have also begun exploring their use in DBMS. In 2024, Huang et al. proposed E2ETune, which uses generative language models for knob tuning. This framework treats knob tuning as an end-to-end sequence generation task [10].

There are three stages in total. Firstly, the authors generate some training data to teach the model how to infer the optimal knob combinations from workloads. E2ETune distinguishes between two types of workloads, Online Analytical Processing (OLAP) and Online Transaction Processing (OLTP). OLAP requires low latency because complex queries are involved, which were simulated using GPT-4-Turbo. OLTP mainly involves high-frequency access to a large number of simple transactions, so it has a high demand for throughput. Since these transactions are limited in types, the authors vary the weighting of transactions to increase diversity.

Next, each workload needs to have an optimal configuration for the model training. The authors use the first 130 workloads and their real tuning results to train a cost model for performance prediction. These collected data (workloads and their predicted performance) are then used in the next stage to train the Mistral-7B (a language model), and knob values are normalised and bucketed to reduce learning difficulty. The model predicts the bucket instead of actual values, which increases accuracy and efficiency. After training, the model recommends knob configurations for new workloads by generating 8 configuration candidates, scoring them using the previously trained cost model, and selecting the best one.

Experiments on PostgreSQL show that the tuning time of E2ETune is only 19.8 minutes, which is 98.6% faster than HEBO (Heterogeneous Evolutionary Bayesian Optimisation, an improved BO method) [10].

Another famous example is GPTuner, which uses LLM in a completely different way compared to E2ETune. Instead of generating configurations, GPTuner applies LLMs to parse database manuals and extract expert knowledge such as optimal parameter ranges [19]. GPTuner can also accept relevant domain knowledge to avoid dangerous configurations and accelerate convergence. This is a new trend of combining AI with expert experience. According to the graph provided by the author, GPTuner reduces latency by approximately 34.5% compared to the GP used in OtterTune [19].

3. Discussion

It can be seen that researchers have explored different areas of ML and proposed many novel and feasible solutions to automate knob tuning in databases. The market has also recognised the value of ML-based tuners. For example, OtterTune raised $12 million in Series A funding in 2022 to support the expansion of its automatic database tuning service to new platforms [20]. From OtterTune, which required manual intervention, to the end-to-end deep RL-based CDBTune, researchers have been committed to automating the entire tuning process and improving database performance. However, AI still faces some challenges and limitations in database knob tuning.

DOI: 10.54254/2755-2721/2026.TJ29634

3.1. Challenges

Firstly, the difference in real application scenarios. Different workloads have different features, which means a model may only be applicable to a specific workload. Therefore, the tuner needs to identify the workload features and adjust its strategy accordingly. For example, OtterTune reuses tuning experience by comparing with historical workloads [9]. RL-based methods represented by CDBTune can adapt to the current workload through trial and error, but they still face the challenge of balancing exploration and exploitation. If RL agents excessively explore, they may try some dangerous configurations, resulting in significant fluctuations in system performance. If they stay at a relatively fair position too early, the system may not reach its optimal level.

Secondly, performance is affected by multiple factors. Database performance is not only affected by knob configurations, but also by the system itself and the current request frequency. This means that even with the same knob configuration, there may be very different performances under different conditions, which makes it more difficult to train tuning models and RL agents precisely.

Thirdly, the feasibility of configurations. Most models provide knob combinations that maximise performance, but not all combinations are feasible. For example, blindly increasing memory parameters may lead to system crashes [7], and certain combinations may violate internal constraints.

Fourth, training costs. Each tuning process requires a significant number of resources, especially for RL training, which requires many iterations. There are usually hundreds of parameters for mainstream databases like PostgreSQL and MySQL [16], leading to an exponential number of attempts. For example, the offline training of CDBTune takes 4.7 hours, even with 30 servers computing in parallel [9].

3.2. Future prospects

There are some possible solutions for these common challenges, which can be considered in future studies.

Firstly, regarding the exploration-exploitation dilemma in RL, which has always been a common issue in RL, some researchers have proposed feasible methods such as the Upper Confidence Bound (UCB) approach. It assigns a confidence interval to each knob by considering both average performance and uncertainty [21]. This means that the system tends to explore configurations with fewer attempts as their confidence intervals are larger. Another feasible approach is to model the performance of each configuration as a probability distribution problem, and use random sampling to determine whether to consider the current configuration as optimal or to continue exploring new ones [22]. In addition, adjusting strategies based on the level of workload is also a feasible method. By adding a load-aware function to the agent, it can be trained to explore new knob combinations for better performance when the load is low, while staying at the current optimal configuration under high load to avoid performance decrease.

Secondly, to address the issue of multiple influencing factors of performance, a future approach could be to input system parameters (hardware information, current database workload, date, time, etc.) as part of the RL agent's state. This could allow the agent to comprehensively learn how the same configuration works under different conditions, thereby further improving optimisation effectiveness.

Thirdly, regarding the feasibility of configurations, constraints can be added to the future training process. For example, if the CPU usage exceeds a certain threshold, the agent will get a penalty. Another way to consider is to train an ML model to predict the feasibility of configurations before

DOI: 10.54254/2755-2721/2026.TJ29634

applying them, which provides an additional layer of safety. Similar solutions have already appeared on OnlineTune, which evaluates configuration security using black-box and white-box knowledge before applying, successfully reducing unsafe configuration recommendations by 91.0% [23]. Moreover, systems should include an automatic rollback mechanism. Once the configuration is not feasible, they can quickly revert to the last stable configuration, as iBTune does [24].

Finally, for the cost issue, many frameworks already support transfer learning to reduce training costs. In the future, cloud computing can be used to train RL models with its powerful and large-scale computing ability, and then deploy them back locally. A report from the US federal government indicated that migrating traditional IT data centres to the cloud could reduce total costs by 66.6% over 13 years [25]. At the same time, for cloud computing vendors, this is also a great opportunity, where they can build a knowledge base specifically for database knob tuning with a subscription-based service model. New users could reuse existing models instead of training from scratch. This not only saves resources and costs, but also brings long-term benefits to cloud computing vendors.

4. Conclusion

This paper provides a comprehensive overview of database knob tuning. The mentioned literature can be categorised into three types: BO-based methods represented by OtterTune, RL-based methods represented by CDBTune, and LLM-based methods represented by E2ETune, which has great future development prospects. Through analysis of this field, this paper also identifies some challenges, such as the exploration-exploitation dilemma of RL, multiple factors affecting performance, configuration feasibility, and cost issues. At the same time, some future possible improvement directions have also been discussed, such as using confidence intervals, inputting more information to RL agents, training ML models to verify configuration feasibility, and using cloud computing for model training.

References

[1] National Research Council, Funding a Revolution: Government Support for Computing Research. National Academies Press, 1999.
[2] E. F. Codd, 'A relational model of data for large shared data banks', Commun. ACM, vol. 13, no. 6, pp. 377–387, 1970, doi: 10.1145/362384.362685.
[3] D. D. Chamberlin and R. F. Boyce, 'SEQUEL: A structured English query language', in Proceedings of the 1974 ACM SIGFIDET (now SIGMOD) workshop on Data description, access and control, in SIGFIDET '74. New York, NY, USA: Association for Computing Machinery, 1974, pp. 249–264. doi: 10.1145/800296.811515.
[4] R. Greenwald, R. Stackowiak, and J. Stern, Oracle Essentials: Oracle Database 12c. O'Reilly Media, Inc., 2013.
[5] D. Kunda and H. Phiri, 'Comparative Study of NoSQL and Relational Database', zictjournal, vol. 1, no. 1, pp. 1–4, Dec. 2017, doi: 10.33260/zictjournal.v1i1.8.
[6] A. Vaswani et al., 'Attention Is All You Need', June 12, 2017, arXiv: arXiv: 1706.03762. doi: 10.48550/arXiv.1706.03762.
[7] D. Van Aken, A. Pavlo, G. J. Gordon, and B. Zhang, 'Automatic Database Management System Tuning Through Large-scale Machine Learning', in Proceedings of the 2017 ACM International Conference on Management of Data, Chicago Illinois USA: ACM, May 2017, pp. 1009–1024. doi: 10.1145/3035918.3064029.
[8] X. Zhao, X. Zhou, and G. Li, 'Automatic Database Knob Tuning: A Survey', IEEE Transactions on Knowledge and Data Engineering, vol. 35, no. 12, pp. 12470–12490, Dec. 2023, doi: 10.1109/TKDE.2023.3266893.
[9] J. Zhang et al., 'An End-to-End Automatic Cloud Database Tuning System Using Deep Reinforcement Learning', in Proceedings of the 2019 International Conference on Management of Data, in SIGMOD '19. New York, NY, USA: Association for Computing Machinery, 2019, pp. 415–432. doi: 10.1145/3299869.3300085.
[10] X. Huang et al., 'E2ETune: End-to-End Knob Tuning via Fine-tuned Generative Language Model', Mar. 19, 2025, arXiv: arXiv: 2404.11581. doi: 10.48550/arXiv.2404.11581.

DOI: 10.54254/2755-2721/2026.TJ29634

[11] S. Duan, V. Thummala, and S. Babu, 'Tuning database configuration parameters with iTuned', Proc. VLDB Endow., vol. 2, no. 1, pp. 1246–1257, Aug. 2009, doi: 10.14778/1687627.1687767.
[12] B. Zhang et al., 'A demonstration of the ottertune automatic database management system tuning service', Proc. VLDB Endow., vol. 11, no. 12, pp. 1910–1913, Aug. 2018, doi: 10.14778/3229863.3236222.
[13] X. Zhang et al., 'ResTune: Resource Oriented Tuning Boosted by Meta-Learning for Cloud Databases', in Proceedings of the 2021 International Conference on Management of Data, Virtual Event China: ACM, June 2021, pp. 2102–2114. doi: 10.1145/3448016.3457291.
[14] K. Kanellis, C. Ding, B. Kroth, A. Müller, C. Curino, and S. Venkataraman, 'LlamaTune: Sample-Efficient DBMS Configuration Tuning', Aug. 23, 2022, arXiv: arXiv: 2203.05128. doi: 10.48550/arXiv.2203.05128.
[15] B. Milicevic and Z. Babovic, 'A systematic review of deep learning applications in database query execution', Journal of Big Data, vol. 11, no. 1, p. 173, Dec. 2024, doi: 10.1186/s40537-024-01025-1.
[16] Y. Zhu et al., 'BestConfig: Tapping the Performance Potential of Systems via Automatic Configuration Tuning', in Proceedings of the 2017 Symposium on Cloud Computing, Sept. 2017, pp. 338–350. doi: 10.1145/3127479.3128605.
[17] G. Li, X. Zhou, S. Li, and B. Gao, 'QTune: a query-aware database tuning system with deep reinforcement learning', Proc. VLDB Endow., vol. 12, no. 12, pp. 2118–2130, Aug. 2019, doi: 10.14778/3352063.3352129.
[18] F. Ye, Y. Li, X. Wang, N. Nedjah, P. Zhang, and H. Shi, 'Parameters tuning of multi-model database based on deep reinforcement learning', J Intell Inf Syst, vol. 61, no. 1, pp. 167–190, Aug. 2023, doi: 10.1007/s10844-022-00762-0.
[19] J. Lao et al., 'GPTuner: A Manual-Reading Database Tuning System via GPT-Guided Bayesian Optimization', Proc. VLDB Endow., vol. 17, no. 8, pp. 1939–1952, Apr. 2024, doi: 10.14778/3659437.3659449.
[20] K. Wiggers, 'OtterTune, which taps AI to optimize databases, raises $12M', TechCrunch. Accessed: Oct. 01, 2025. [Online]. Available: https: //techcrunch.com/2022/05/10/2309852/
[21] K. Jamieson, M. Malloy, R. Nowak, and S. Bubeck, 'lil' UCB : An Optimal Exploration Algorithm for Multi-Armed Bandits', Dec. 27, 2013, arXiv: arXiv: 1312.7308. doi: 10.48550/arXiv.1312.7308.
[22] D. Russo, B. V. Roy, A. Kazerouni, I. Osband, and Z. Wen, 'A Tutorial on Thompson Sampling', July 14, 2020, arXiv: arXiv: 1707.02038. doi: 10.48550/arXiv.1707.02038.
[23] X. Zhang, H. Wu, Y. Li, J. Tan, F. Li, and B. Cui, 'Towards Dynamic and Safe Configuration Tuning for Cloud Databases', in Proceedings of the 2022 International Conference on Management of Data, June 2022, pp. 631–645. doi: 10.1145/3514221.3526176.
[24] J. Tan et al., 'iBTune: individualized buffer tuning for large-scale cloud databases', Proc. VLDB Endow., vol. 12, no. 10, pp. 1221–1234, June 2019, doi: 10.14778/3339490.3339503.
[25] T. Alford, G. Morton, and A. Repczynski, 'Addressing the Benefits of Infrastructure in the Cloud', 2010, [Online]. Available: https: //www.iceaaonline.com/wp-content/uploads/2017/09/SOF09A-Repczynski.pdf