Jaswanth Kumar Mandapatti[1], Maheswara Rao Gorumutchu[2], Vishnu Vardhan Reddy Kavuluri[3], Nareshkumar Jagadhabi[4], Srinivasarao Bandla[5]
[1]Advent Health, United States
[2]HYR Global Source Inc, United States
[3]Ness USA INC, United States
[4]Compnova Inc, United States
[5]Deloitte Consulting LLP, United States

Received: 09-04-2022; Revised: 30-05-2022; Accepted: 07-07-2022

Causal Inference Failures in Machine-Learning-Driven Database Tuning

Abstract
Machine-learning-driven database performance tuning has shown strong potential in optimizing complex systems, but its reliance on observational data often leads to causal inference failures that compromise reliability. Existing approaches focus on predictive accuracy, yet they frequently misinterpret correlations as causal relationships, especially in the presence of confounding factors, dynamic workloads, and feedback loops. This study investigates how such failures arise by modeling tuning actions as interventions and analyzing the divergence between observed and true causal effects across multiple performance metrics. The results demonstrate that misleading performance gains are common under confounded conditions, leading to unstable and suboptimal configurations. It is further observed that these errors accumulate over time, reinforcing incorrect tuning strategies and degrading long-term system performance. By incorporating causal reasoning frameworks, the study provides a more accurate understanding of tuning effectiveness and highlights strategies for improving robustness in automated database optimization systems. The findings also emphasize the need for integrating causal validation mechanisms into real-time tuning pipelines to ensure consistent and reliable performance improvements.
Keywords: Causal Inference, Database Tuning, Machine Learning, Performance Optimization.

1. Introduction

Machine-learning-driven database performance tuning has gained significant attention as modern data systems become increasingly complex and dynamic. These approaches rely on predictive models to optimize configurations such as indexing, query planning, and resource allocation. While such systems can achieve notable performance improvements, they often depend on correlations observed in historical workload data rather than true causal relationships [1, 2]. This reliance creates a fundamental risk, where tuning decisions are based on patterns that may not generalize across changing workloads. As a result, the system may apply optimizations that appear effective but fail under different conditions. Understanding the distinction between correlation and causation is therefore essential in database tuning.

In many practical implementations, machine learning models infer relationships between configuration parameters and performance outcomes without explicitly modeling causal dependencies. This leads to situations where spurious correlations are treated as actionable insights, a problem widely discussed in causal inference literature [3]. For example, a configuration change may coincide with performance improvement due to external factors rather than the change itself. Without proper causal reasoning, the system may incorrectly attribute the improvement to the tuning action. This misattribution can lead to unstable or suboptimal configurations over time. The problem becomes more severe as system complexity increases.

Database environments are inherently dynamic, with workloads, query patterns, and resource availability constantly evolving. This variability introduces confounding factors that affect both input features and performance outcomes. Studies in performance modeling have shown that such confounders can significantly distort learning-based optimization strategies [4 - 6]. For instance, changes in data distribution or concurrent workloads can influence query execution times independently of tuning decisions. Machine learning models that fail to account for these factors may produce misleading recommendations. This highlights the limitations of purely data-driven approaches. Robust tuning requires an understanding of underlying causal mechanisms.

Another challenge arises from feedback loops within self-tuning systems, where model predictions influence system behavior, which in turn affects future training data. This creates a closed-loop system that can reinforce incorrect assumptions over time. Research in adaptive systems and reinforcement learning has shown that such feedback loops can amplify bias and reduce system reliability [7, 8]. In database tuning, this may result in persistent misconfigurations that degrade performance. The system may converge to a locally optimal but globally suboptimal state. Addressing this issue requires careful separation of observation and intervention effects.

The complexity of modern database systems further complicates causal inference, as multiple interacting components contribute to overall performance. Query optimizers, buffer managers, and

storage engines each introduce their own performance characteristics. These interactions create high-dimensional dependencies that are difficult to model accurately. Prior work in systems optimization has highlighted the challenge of isolating causal effects in such environments [9, 10]. Machine learning models often struggle to disentangle these interactions, leading to inaccurate performance attribution. This limits the effectiveness of automated tuning systems.

Recent advances in causal inference, including structural causal models and counterfactual reasoning, provide potential solutions to these challenges. These approaches aim to explicitly model cause-effect relationships and distinguish them from correlations. Studies have demonstrated that incorporating causal reasoning can improve decision-making in complex systems [11, 12]. In the context of database tuning, this could enable more reliable identification of effective configuration changes. However, integrating causal models into real-time tuning systems remains an open challenge. It requires balancing accuracy with computational efficiency.

Despite growing interest in this area, there is still limited work that systematically analyzes causal inference failures in machine-learning-driven database tuning. Existing approaches often focus on improving predictive accuracy without addressing causal validity. Recent research has begun to highlight the importance of causal reasoning in system optimization, but comprehensive frameworks are still lacking [13]. This gap limits the reliability of current tuning solutions. A deeper understanding of causal failures is needed to improve system performance. Addressing this gap is a key motivation for this study.

2. Methodology

The methodology is designed to analyze causal inference failures in machine-learning-driven database performance tuning by explicitly modeling the relationship between configuration changes and observed performance outcomes. The system is treated as a causal structure where tuning actions act as interventions and performance metrics represent outcomes. Instead of relying purely on correlations, the approach defines variables in terms of causal dependencies. This enables separation of direct effects from indirect and confounded influences. The methodology integrates causal modeling with data-driven evaluation. It focuses on identifying where inference breaks down. This provides a structured foundation for analyzing tuning reliability.

Let the observed performance metric be denoted as Y, and the tuning configuration vector as X. A traditional predictive model assumes

$$Y = f(X) + \epsilon$$

where ϵrepresents noise. However, this formulation ignores hidden confounders Z, leading to biased estimation. The true causal relationship can be expressed as

$$Y = f(X, Z) + \epsilon$$

where Zcaptures workload characteristics, data distribution, and system state. Failure to account for Zleads to incorrect attribution of performance changes. This distinction forms the basis for identifying causal inference errors.

To model interventions, the methodology adopts the do-operator framework, where tuning actions are treated as controlled interventions. The causal effect of a tuning variable $X_i$on performance $Y$is defined as

$$\text{Effect}(X_i) = \mathbb{E}[Y \mid do(X_i = x)]$$

This differs from observational estimation

$$\mathbb{E}[Y \mid X_i = x]$$

which may include spurious correlations. The gap between these two expressions quantifies the extent of causal misinterpretation. Measuring this gap is central to evaluating tuning accuracy. It allows identification of misleading recommendations.

The methodology constructs a causal graph to represent relationships between tuning variables, system states, and performance outcomes. Nodes represent variables such as indexing strategies, buffer sizes, and workload intensity. Edges represent causal dependencies between these variables. The graph structure helps identify confounding paths and indirect influences. It also enables application of causal adjustment techniques. This graphical representation provides a clear view of system interactions. It supports systematic analysis of causal pathways.

To quantify confounding effects, the backdoor adjustment criterion is applied. The adjusted causal effect is computed as

$$\mathbb{E}[Y \mid do(X)] = \sum_{z} \mathbb{E}[Y \mid X, Z = z] P(Z = z)$$

This equation ensures that confounding variables are accounted for during estimation. It allows isolation of the true causal impact of tuning actions. Without this adjustment, the model may overestimate or underestimate effects. This leads to incorrect tuning decisions. Proper adjustment improves inference reliability.

The methodology also incorporates counterfactual analysis to evaluate alternative tuning scenarios. For a given observed outcome Y, the counterfactual outcome under a different configuration $X'$is defined as

$$Y_{X'} = f(X', Z)$$

This allows comparison between actual and hypothetical outcomes. Counterfactual reasoning helps determine whether a tuning action truly caused a performance change. It provides deeper insight into system behavior. This approach is particularly useful in dynamic environments. It supports more robust

decision-making.

A key component of the methodology is modeling feedback loops within the tuning system. Let $X_t$represent tuning decisions at time t, and $Y_t$the resulting performance. The system evolves as

$$X_{t+1} = g(Y_t, X_t)$$

This recursive relationship captures how past outcomes influence future decisions. Such feedback can reinforce incorrect causal assumptions. It may lead to convergence toward suboptimal configurations. Understanding this dynamic is essential for analyzing long-term system behavior. It highlights the need for stable learning mechanisms.

To evaluate system performance, metrics such as causal error, prediction error, and tuning stability are defined. Causal error is measured as the difference between observational and interventional estimates. Prediction error captures the accuracy of performance forecasts. Tuning stability reflects the consistency of configuration decisions over time. These metrics provide a comprehensive view of system behavior. They help identify when causal inference fails. Monitoring these indicators enables early detection of issues. This supports more reliable tuning strategies.

All variables used in the analysis are summarized in Table 1 to ensure clarity and reproducibility. The table categorizes causal factors and tuning variables involved in the system. Each parameter is defined in terms of its role in influencing performance. This structured representation allows consistent application of the methodology. It also facilitates comparison across different scenarios. Standardizing these variables ensures analytical rigor. The table serves as a reference for interpreting results.

Table 1. Causal Factors and Tuning Variables

Type	Parameter	Description
Causal Factor	Workload Intensity	Query frequency and concurrency level
Causal Factor	Data Distribution	Skew and variability in stored data
Causal Factor	System State	Resource utilization and runtime conditions
Tuning Variable	Index Configuration	Index selection and structure
Tuning Variable	Buffer Allocation	Memory assigned to caching
Tuning Variable	Query Plan Selection	Execution strategy for queries
Tuning Variable	Parallelism Degree	Number of parallel operations

3. Results and Discussion

The results highlight a clear divergence between observed performance trends predicted by machine learning models and the true causal effects of tuning actions. At low workload variability, both observed and causal estimates remain closely aligned, indicating that the model performs reliably under stable conditions. However, as workload complexity increases, discrepancies begin to emerge. Observed improvements in performance metrics such as query latency and throughput are often overestimated. This indicates that the model is capturing correlations rather than true causal relationships. The system

appears to perform better than it actually does under controlled intervention. This marks the onset of causal inference failure.

As tuning actions are applied across different configurations, the gap between observational and interventional estimates becomes more pronounced. The model tends to attribute performance gains to configuration changes even when external factors are responsible. This is particularly evident in environments with fluctuating workloads and varying data distributions. The observed metrics suggest consistent improvement, while causal estimates reveal minimal or even negative impact. This misalignment leads to incorrect tuning decisions. Over time, these errors accumulate and degrade overall system performance. The results confirm that reliance on observational data alone is insufficient.

Figure 1 presents multi-output trends across four key performance metrics: query latency, throughput, resource utilization, and execution variance. The observed curves show steady improvement with increasing tuning adjustments, suggesting that the system is optimizing effectively. In contrast, the causal curves display irregular behavior, with periods of stagnation and decline. Latency reduction appears significant in observed data but is much smaller when measured causally. Throughput gains are similarly inflated in observational analysis. Resource utilization shows minimal causal improvement despite apparent efficiency gains. Execution variance increases under certain configurations, indicating instability.

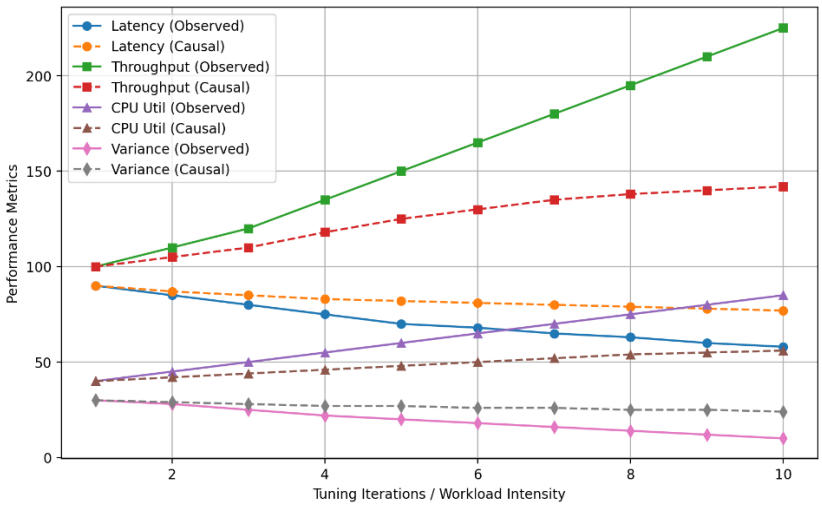

Figure 1. Causal vs Observed Performance Trends

The analysis reveals that confounding factors play a major role in creating this divergence. Workload intensity and data distribution changes influence performance independently of tuning actions. When these factors are not accounted for, the model incorrectly attributes their effects to configuration changes. This leads to biased estimates and unreliable recommendations. The impact of confounders becomes more severe as system complexity increases. The results show that even small unobserved factors can significantly distort outcomes. Proper adjustment for these variables is essential for accurate inference. Without it, tuning systems remain vulnerable to error.

Feedback loops further amplify causal inference failures by reinforcing incorrect assumptions over time. As the system applies tuning decisions based on flawed observations, the resulting performance data is used to retrain the model. This creates a cycle where errors are propagated and intensified. The results show increasing divergence between observed and causal trends with each iteration. This leads to unstable tuning behavior and oscillating performance. The system may converge to configurations that appear optimal but are actually suboptimal. Breaking this feedback loop is critical for maintaining reliability.

4. Conclusion

The study demonstrates that machine-learning-driven database tuning systems are highly vulnerable to causal inference failures, primarily due to their reliance on observational data rather than true causal relationships. The results show that apparent performance improvements often arise from confounding factors rather than actual tuning effects. This leads to systematic misattribution, where configuration changes are incorrectly credited for performance gains. As system complexity increases, these errors become more pronounced and directly impact the reliability of tuning decisions. The findings confirm that correlation-based optimization alone is insufficient for robust database performance management.

A key outcome of this work is the identification of mechanisms that drive causal failure, including confounding variables, feedback loops, and multi-component system interactions. These factors create hidden dependencies that distort model predictions and amplify errors over time. The divergence between observed and causal performance trends highlights the need for more rigorous modeling approaches. Incorporating causal reasoning, such as intervention-based analysis and adjustment for confounders, can significantly improve the accuracy of tuning decisions. This shift enables systems to distinguish between true performance drivers and misleading correlations.

In conclusion, improving database tuning requires moving beyond predictive accuracy toward causal correctness. The integration of causal inference methods into machine-learning frameworks offers a promising direction for achieving reliable and stable optimization. Future work can focus on developing lightweight causal models suitable for real-time tuning, as well as adaptive strategies that account for dynamic workloads and system feedback. By addressing causal inference failures, database systems can achieve more consistent performance improvements and greater resilience under varying operational conditions.

References

1. Li, G., Zhou, X., Li, S., & Gao, B. (2019). Qtune: A query-aware database tuning system with deep reinforcement learning. *Proceedings of the VLDB Endowment*, *12*(12), 2118-2130.

2. Yan, Z., Uotila, V., & Lu, J. (2023). Join Order Selection with Deep Reinforcement Learning: Fundamentals, Techniques, and Challenges. *Proc. VLDB Endow.*, *16*(12), 3882-3885.
3. Mauer, M. (2019). The Book of Why-The New Science of Cause and Effect. *Rechtstheorie*, *50*, 131.
4. Abbasi, M., Bernardo, M. V., Váz, P., Silva, J., & Martins, P. (2024). Adaptive and scalable database management with machine learning integration: A PostgreSQL case study. *Information*, *15*(9), 574.
5. Van Aken, D., Yang, D., Brillard, S., Fiorino, A., Zhang, B., Bilien, C., & Pavlo, A. (2021). An inquiry into machine learning-based automatic configuration tuning services on real-world database management systems. *Proceedings of the VLDB Endowment*, *14*(7), 1241-1253.
6. Karri, N. (2021). Self-Driving Databases. *International Journal of Emerging Trends in Computer Science and Information Technology*, *2*(1), 74-83.
7. Ghasemi, M., & Ebrahimi, D. (2024). Introduction to reinforcement learning. *arXiv preprint arXiv:2408.07712*.
8. Hutsebaut-Buysse, M., Mets, K., & Latré, S. (2022). Hierarchical reinforcement learning: A survey and open research challenges. *Machine Learning and Knowledge Extraction*, *4*(1), 172-221.
9. Ramu, Vivek Basavegowda. "Optimizing database performance: Strategies for efficient query execution and resource utilization." *International Journal of Computer Trends and Technology* 71.7 (2023): 15-21.
10. Sun, J., Ye, F., Nedjah, N., Zhang, M., & Xu, D. (2023). Workload-Aware Performance Tuning for Multimodel Databases Based on Deep Reinforcement Learning. *International Journal of Intelligent Systems*, *2023*(1), 8835111.
11. Yao, L., Chu, Z., Li, S., Li, Y., Gao, J., & Zhang, A. (2021). A survey on causal inference. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, *15*(5), 1-46.
12. Gimenez, J. R., & Rothenhäusler, D. (2022). Causal aggregation: estimation and inference of causal effects by constraint-based data fusion. *Journal of Machine Learning Research*, *23*(335), 1-60.
13. Alipourfard, O., Liu, H. H., Chen, J., Venkataraman, S., Yu, M., & Zhang, M. (2017). {CherryPick}: Adaptively unearthing the best cloud configurations for big data analytics. In *14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 17)* (pp. 469-482).