Large Language Models for Query Optimization

Jonathan Zeismer, Ac Hýbl
School of Computing
Southern Adventist University
Email: {matoush, jziesmer}@southern.edu

***Abstract*—While Relational Database Management Systems (RDBMSs) have seen continuous refinement, artificial intelligence (AI) may be the next technology to propel database advancement. Models have successfully been used for cost estimation and query explanation. Recent generative AI models have been successfully applied to many tasks, suggesting their utility in databases. This research assess the capabilities of base LLMs in the context of query execution plans and optimization with the intent that the results could be compared to database-specific models.**

For our approach we provide the open-source Llama 2 model with queries and database statistics and observe what types of plans it produces and which biases it exhibits. We engineer system prompts and conversations for the model to optimize its execution plan generation. We find that the model is particularly useful for something, but not as useful for other things.

***Index Terms*—Large Language Models, AI4DB, SQL Server, Query Optimization**

I. Introduction

It has been over fifty years since Edgar Codd first defined relational databases [1]. Relational databases and Relational Database Management Systems (RDBMSs) have been fine-tuned and optimized over the decades. Rather than yet another algorithm, the next major advancement for database technology lies in incorporating artificial intelligence (AI).

Li, Zhou and Cao summarized research topic and future milestone as leveraging AI to create a more intelligent database, succinctly abbreviating this paradigm as "AI4DB" [2]. One of the primary implementations of AI4DB involves "learning-based database optimization." This may involve various optimizations including:

1) seeking to use artificial intelligence to help choose the join order of a query
2) estimating the size of the results of a potential operation
3) replacing one query with another more direct query

One potential integration joins large language models (LLMs) and databases. Since Vaswani et al. introduced transformers in 2017, researchers have incrementally trained increasingly more capable models to produce and understand natural language [3]. Additionally, these models contain large amounts of knowledge about the real world that may improve RDBMS performance.

Although domain-specific LLMs can be trained on example or even production databases, it is critical to assess the capabilities of less specific models to provide a baseline for future development. Therefore, the goal of this research is to accurately asses the capabilities of large language models when applied to query execution plans and their optimization.

This research tests LLM capabilities by requiring an LLM to produce query optimization plans given table statistics and a query. The outputs are then qualitatively compared to the an optimized RDBMS execution plan.

Because most LLMs are closed source, we can only use system prompts with open-source models. Thus, we will focus on only evaluating Llama 2 with these methods [4]. Also, our research does not train new models or fine tune current models, so it cannot be used to determine the utility of models trained specifically for databases.

II. Related Work

In their "AI Meets Database: AI4DB and BD4AI" paper, Li, Zhou and Cao identify three primary ways artificial intelligence is being used to optimize database performance [2]: cost estimation, join order, and complete optimization. Cost estimation plays a major role in join order selection, as performing joins and filters on smaller tables before larger ones decreases search times and memory usage. A query's execution plan can vary in many fundamental and nuanced characteristics, yielding millions of execution options. Thus, certain AI-based algorithms have also been used in this area to quickly arrive at an approximately optimal plan.

In 2019, Ji Sun and Guoliang Li implemented an advanced cost estimation model [5]. This model consisted of three layers to handle embedding, high-level query representation, and output. In order to capture the tree-like requirements of most queries and their execution plans, the model's structure was tree-based and allowed nodes to learn their subnodes' execution plans. Once this structure was achieved in the representation-layer, the estimation layer was able to produce more accurate costs than several baselines. Nevertheless, this approach suffers from needing to train a database-specific model for each application. Thus, it exchanges accuracy for generality.

Wang et al. explore another aspect relating to our research in their development of LANTERN, a tool for explaining SQL query execution plans [6]. While achieving an 86% success rate in a related task, this study only describes query execution plans rather than generating them. It is useful for human learners, but not for database management systems (DBMS).

Ali et al. present an approach for optimizing the querying of large-scale heterogeneous models in low-code platforms through compile-time static analysis and specific query optimizers/translators, aiming to improve query execution time and memory footprint [7]. They do not fully address real-time

adaptability in highly dynamic or unpredictable data environments, but it may be fruitful to combine their techniques with LLM optimization in the future. Additionally, a well-trained LLM may be able to provide more robust optimization.

Ambite and Knoblock present a novel approach to query planning in mediators using the Planning by Rewriting (PbR) paradigm, focusing on optimizing plan quality through efficient local search techniques and flexible, scalable system design, demonstrating improved scalability and plan quality over existing methods [8].

Another paper did this. It differs from what we're doing like this. Etc.

III. Methodology

IV. Results

V. Conclusion

References

[1] E. F. Codd, "A relational model of data for large shared data banks," *Communications of the ACM*, vol. 13, no. 6, pp. 377–387, 1970.

[2] G. Li, X. Zhou, and L. Cao, "Ai meets database: Ai4db and db4ai," in *Proceedings of the 2021 International Conference on Management of Data*, 2021, pp. 2859–2866.

[3] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017, available at: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[4] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale *et al.*, "Llama 2: Open foundation and fine-tuned chat models," *arXiv preprint arXiv:2307.09288*, 2023, available at: https://doi.org/10.48550/arXiv.2307.09288.

[5] J. Sun and G. Li, "An end-to-end learning-based cost estimator," *arXiv preprint arXiv:1906.02560*, 2019.

[6] W. Wang, S. S. Bhowmick, H. Li, S. Joty, S. Liu, and P. Chen, "Towards enhancing database education: Natural language generation meets query execution plans," in *Proceedings of the 2021 International Conference on Management of Data*, 2021, pp. 1933–1945.

[7] Q. ul ain Ali, D. Kolovos, and K. Barmpis, "Efficiently querying large-scale heterogeneous models," in *ACM/IEEE 23rd International Conference on Model Driven Engineering Languages and Systems (MODELS '20 Companion)*. Virtual Event, Canada: ACM, 2020.

[8] J. L. Ambite and C. A. Knoblock, "Flexible query planning in mediators," *Journal of Intelligent Information Systems*, vol. 14, pp. 5–28, 2000.