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. Theoretical Framework

This section provides a brief explanation of the key concepts used in this research. Figure 1 summarizes the various concepts and their relationships.

A. Natural Language Processing

QAG belongs to a broad field of problems involving machines and human language. These problems can be divided into several categories.

K. R. Chowdhary defines one category, natural language processing (NLP), as "a collection of computational techniques for automatic analysis and representation of human languages, motivated by theory" [9]. NLP ranges from simpler functionality such as spell checking to more challenging problems like the retrieval of important information. Some NLP problems can be further categorized under natural language understanding (NLU). NLU is a more advanced form of NLP in which machines understand language using background information much like humans do. NLU requires semantic information, abstract concepts, and various modules.

Chowdhary separates natural language generation (NLG) from NLP, categorizing both as branches of computational linguistics. While NLP is focused on language analysis, NLG involves machines writing human language rather than just reading and understanding it.

While these categories have different goals, NLG is interconnected with NLP. To generate text, a machine must first achieve a level of NLU. Because QAG requires a machine to generate natural language to achieve the desired output, it is considered part of NLG.

B. AI Models

Stanford University's Human-Centered Artificial Intelligence (HAI) institute cites the original definition of artificial intelligence (AI), "the science and engineering of making intelligent machines," coined by John McCarthy in 1955[1]. This definition relies on the definition of intelligence, which the HAI defines as the ability to learn. Hence, it follows that *artificial* intelligence is identified by a machine's ability to learn.

In *Machine Learning: An Artificial Intelligence Approach* [10], Michalski et al. introduce the necessity of this type of learning by explaining that some tasks are simply too difficult to "laboriously program" into a computer. In other words, AI allows computers to perform tasks that humans perform but cannot fully articulate algorithmically. Rather than explicitly programming billions of decisions with conditional statements (*if* this *then* do that), machine learning allows a computer to "learn by example" and program these "decisions" automatically.

[1] hai.stanford.edu

C. Machine Learning

Machine learning (ML) is a broad term for the process by which a computer constructs an AI model to perform a task. Arthur Samuel, who coined the term [11], defined "machine learning" as the "field of study that gives computers the ability to learn without being explicitly programmed." Machine learning can take many different forms such as logistic regression, Naive Bayes, or clustering. In this thesis, we focus on deep learning.

D. Deep Learning

Michael Nielsen provides an excellent introduction to deep learning and neural networks in *Neural Networks and Deep Learning* [12]. Deep learning is a subset of machine learning used to train neural networks. The fundamental unit of such a network is often called a node, neuron, or perceptron (see Figure 2).

An artificial neuron receives input, performs a mathematical (usually linear) transformation, and then returns some output, usually 0, 1, or some rational number in between. By combining many of these neurons in sequence, output to input, and in parallel, computer scientists construct neural networks (see Figure 3).

The importance, or weight, of the connection between two perceptrons is the "trainable" part of the network. Deep learning uses backpropagation to adjust these connections by constantly taking partial derivatives between a provided example and the current state of each neuron. The neuron weights must be repeatedly tuned until the model can perform well with problems it has never seen before. Thus, to train a neural network using deep learning, we must possess many examples to present to the learning model. These examples are referred to as data.

E. Data

For a model to learn based on empirical evidence, it must be shown many examples representative of the target task. The success of this learning process relies on the format and quality of these examples. For this thesis, we plan to utilize two separate but similar datasets.

F. Transformers

Most QG systems before the 2010s were rule-based, systematically transforming a context from a declarative sentence to a question. The introduction of transformers revolutionized much NLG research including the field of question generation [3]. Because many more thorough explanations can be found

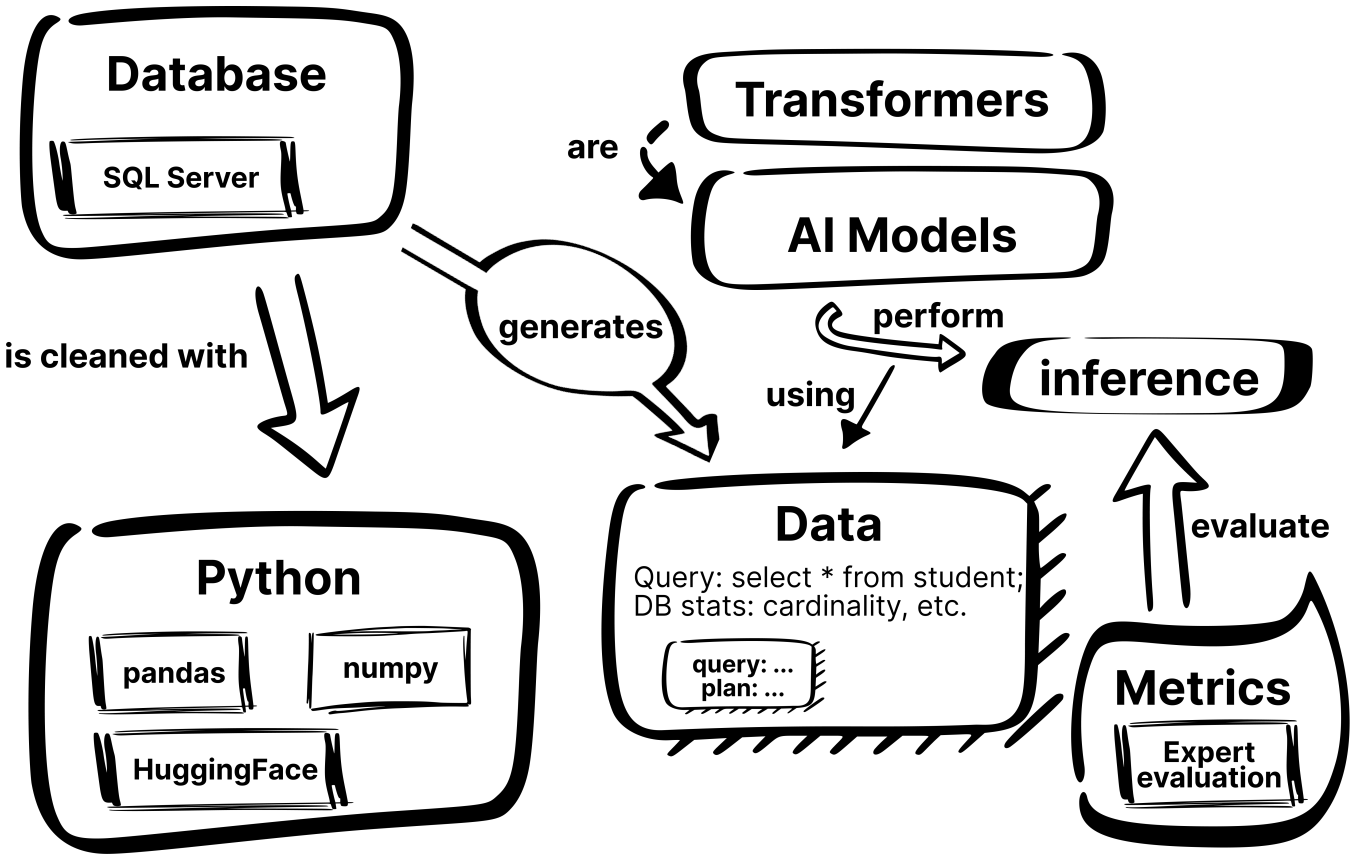

Fig. 1. Concept map

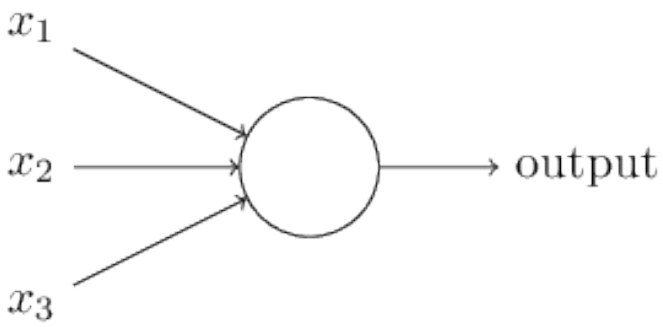

Fig. 2. A perceptron with three inputs x_1, x_2, and x_3 [12]

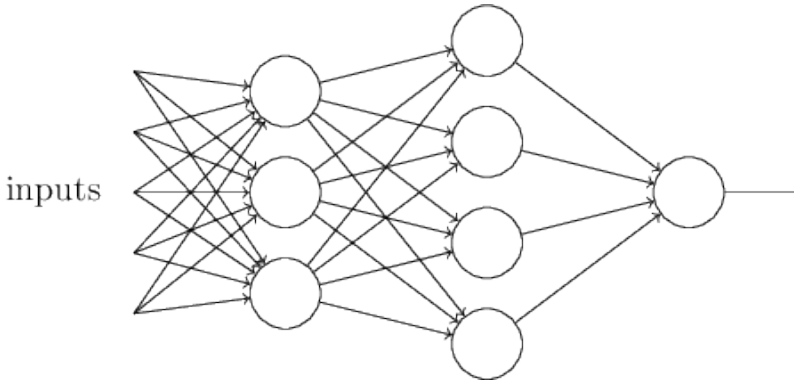

Fig. 3. Connecting many perceptrons forms an artificial neural network [12]

online and in published research, we will only briefly explain transformers at a high level here. As the title of Vaswani et al.'s "Attention is All You Need" paper suggests, rather than using long short-term memory (LSTM), transformers leverage attention mechanisms and to allow a model to selectively retain pieces of important information from past input and output sequences. This provides the model with a selective but theoretically infinite memory. Self-attention allows words near each other to adjust each other's meanings. Moreover, unlike Recurrent Neural Networks (RNNs) which generate words in sequence by feeding output back into the model, transformers utilize positional encoding, allowing for faster inference and training. This architecture not only performs well in NLG tasks, but is also parallelizable, making transformer training on vast datasets much easier.

G. Python and Libraries

Due to the various state-of-the-art algorithms and resources included in LLM fine-tuning, research works commonly use Python for configuring their training experiments [13]–[18]. Python simplifies the implementation of these algorithms and the access to these resources through its vast amount of useful

AI and data management libraries. For AI tasks, we plan to utilize a family of libraries released by HuggingFace[2] centered around the transformers library [19]. To clean and format data we will use the Python libraries pandas and numpy.

IV. Methodology

V. Results

VI. 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.

[9] K. Chowdhary and K. Chowdhary, "Natural language processing," *Fundamentals of artificial intelligence*, pp. 603–649, 2020, available at: https://doi.org/10.1007/978-81-322-3972-7_19.

[10] R. S. Michalski, J. G. Carbonell, and T. M. Mitchell, *Machine Learning: An Artificial Intelligence Approach (Volume I)*. Elsevier, 2014, vol. 1, available at: https://books.google.com/books?hl=en&lr=&id=Aw2jBQAAQBAJ&oi=fnd&pg=PP1&dq=artificial+intelligence+learning+by+experience&ots=_HblQpUHqD&sig=-o-DLIL-AhA_uXQEKx3oh0z3eZY.

[11] Z.-H. Zhou, *Machine learning*. Springer Nature, 2021, available at: https://books.google.com/books?hl=en&lr=&id=ctM-EAAAQBAJ&oi=fnd&pg=PR6&dq=machine+learning&ots=oZRmW8Xr_u&sig=AEotOqjqV_3MF4Gp1cPrW_m3hUE.

[12] M. A. Nielsen, *Neural networks and deep learning*. Determination press San Francisco, CA, USA, 2015, vol. 25, available at: https://www.ise.ncsu.edu/fuzzy-neural/wp-content/uploads/sites/9/2022/08/neuralnetworksanddeeplearning.pdf.

[13] I. V. Serban, A. García-Durán, C. Gulcehre, S. Ahn, S. Chandar, A. Courville, and Y. Bengio, "Generating factoid questions with recurrent neural networks: The 30m factoid question-answer corpus," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany: Association for Computational Linguistics, Aug. 2016, pp. 588–598, available at: http://dx.doi.org/10.18653/v1/P16-1056. [Online]. Available: https://aclanthology.org/P16-1056

[14] A. Kumar, D. Singh, A. Kharadi, and M. Kumari, "Automation of question-answer generation," in *2021 Fourth International Conference on Computational Intelligence and Communication Technologies (CCICT)*, 2021, pp. 175–180, available at: https://doi.org/10.1109/CCICT53244.2021.00043.

[15] A. Ushio, F. Alva-Manchego, and J. Camacho-Collados, "Generative language models for paragraph-level question generation," *arXiv preprint arXiv:2210.03992*, 2022, available at: https://www.doi.org/10.48550/arXiv.2210.03992.

[16] R. Goyal, P. Kumar, and V. Singh, "Automated question and answer generation from texts using text-to-text transformers," *Arabian Journal for Science and Engineering*, pp. 1–15, 2023, available at: https://doi.org/10.1007/s13369-023-07840-7.

[17] A. Ushio, F. Alva-Manchego, and J. Camacho-Collados, "A practical toolkit for multilingual question and answer generation," *arXiv preprint arXiv:2305.17416*, 2023, available at: https://doi.org/10.48550/arXiv.2305.17416.

[18] M.-H. Hwang, J. Shin, H. Seo, J.-S. Im, H. Cho, and C.-K. Lee, "Ensemble-nqg-t5: Ensemble neural question generation model based on text-to-text transfer transformer," *Applied Sciences*, vol. 13, no. 2, p. 903, 2023, available at: https://doi.org/10.3390/app13020903.

[19] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. Le Scao, S. Gugger, M. Drame, Q. Lhoest, and A. Rush, "Transformers: State-of-the-art natural language processing," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Online: Association for Computational Linguistics, Oct. 2020, pp. 38–45, available at: http://dx.doi.org/10.18653/v1/2020.emnlp-demos.6. [Online]. Available: https://aclanthology.org/2020.emnlp-demos.6

[2]https://huggingface.co/