Database Tuning using Natural Language Processing

Immanuel Trummer
Cornell University, Ithaca (NY), USA
itrummer@cornell.edu

Introduction. We have seen significant advances in the state of the art in natural language processing (NLP) over the past few years [20]. These advances have been driven by new neural network architectures, in particular the Transformer model [19], as well as the successful application of transfer learning approaches to NLP [13]. Typically, training for specific NLP tasks starts from large language models that have been pre-trained on generic tasks (e.g., predicting obfuscated words in text [5]) for which large amounts of training data are available. Using such models as a starting point reduces task-specific training cost as well as the number of required training samples by orders of magnitude [7]. These advances motivate new use cases for NLP methods in the context of databases.

In this context, NLP methods have been mostly used for the user interface. Here, the goal is to make data analysis more user-friendly (e.g., via natural language query interfaces [9, 10, 14]). Instead, I propose to leverage NLP to make data processing more efficient. Specifically, NLP methods can yield useful information for automated tuning tools that solve tasks such as query planning [15], physical design [2], or database system configuration [1].

Use Cases. Performance-relevant information can be extracted from text associated either with the database system itself, the workload, or the data. Text associated with the database system includes the manual as well as blog entries about database performance [12], as well as discussions on forums such as DBA Stack Exchange [4]. In recent work, we have shown that useful and system-specific tuning hints can be automatically extracted from such text [18]. SQL queries contain named elements and comments and can be associated with a text description (e.g., in case of SQL snippets in public repositories [3, 11]). Such information may be useful to classify queries before execution (e.g., to assign an appropriate amount of resources) or to suggest query rewrites to users that cannot be detected by syntactic analysis alone (e.g., proposing to replace part of a query by an SQL snippet from a public repository, if associated text makes it likely that both implement the same function). Finally, the database schema contains useful information that can be unlocked via NLP. For instance, we have recently shown that column names are useful to predict statistical correlation between columns [17] (which influence query plan cost estimates). If such correlations are unknown and need to be discovered [8], NLP-based predictions can help to prioritize analysis efforts to discover correlations faster.

Roadmap. In the simplest case, NLP can be used for a standalone tuning heuristic. We have taken first steps into this direction, showing that extracting suggested system parameter settings from text often leads to improved performance [18]. Other tuning problems (e.g., predicting query execution time from analyzing associated text) may also be amenable to NLP-only approaches. However, while such heuristics can be very efficient (requiring neither trial runs nor expensive data statistics), they are unlikely to produce near-optimal solutions in complex scenarios. Hence, we plan to integrate NLP enhancements with traditional tuning approaches. For instance, information gained via NLP can add features for cost and cardinality estimation models [6, 16, 21]. Alternatively, it may serve as a heuristic, restricting the search space for traditional optimization methods (e.g., excluding join orders whose cost estimates are based on the independence assumption between columns that are likely correlated, based on their names). In all those use cases, the goal is to make tuning tools more robust. Finally, we intend to study whether NLP can expand the optimization scope as well. For instance, text analysis may reveal likely semantic equivalences between data columns or user-defined functions that are not formally specified. This enables query rewrites that traditional optimizers cannot consider (but may have to be validated by users).

REFERENCES

[1] D. V. Aken, A. Pavlo, and G. J. Gordon. Automatic database management system tuning through large-scale machine learning. In *SIGMOD*, pages 1009–1024, 2017.

[2] S. Chaudhuri. Index selection for databases: A hardness study and a principled heuristic solution. *KDE*, 16(11):1313–1323, 2004.

[3] Count - The SQL Notebook. https://github.com/count/sql-snippets, 2021.

[4] DBA Stack Exchange. https://dba.stackexchange.com/, 2021.

[5] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, volume 1, pages 4171–4186, 2019.

[6] B. Hilprecht, A. Schmidt, M. Kulessa, A. Molina, K. Kersting, and C. Binnig. DeepDB: Learn from Data, not from Queries! *PVLDB*, 13(7):992–1005, 2019.

[7] J. Howard and S. Ruder. Universal Language Model Fine-tuning for Text Classification. In *ACL*, pages 328–339, 2018.

[8] I. F. Ilyas, V. Markl, P. Haas, P. Brown, and A. Aboulnaga. CORDS: Automatic discovery of correlations and soft functional dependencies. In *SIGMOD*, pages 647–658, 2004.

[9] F. Li and H. Jagadish. NaLIR: an interactive natural language interface for querying relational databases. *SIGMOD*, pages 709–712, 2014.

[10] X. V. Lin, R. Socher, and C. Xiong. Bridging Textual and Tabular Data for Cross-Domain Text-to-SQL Semantic Parsing. pages 4870–4888, 2020.

[11] Microsoft. https://github.com/microsoft/sql-data-warehouse-samples, 2021.

[12] PERCONA. https://www.percona.com/blog/2018/08/31/tuning-postgresql-database-parameters-to-optimize-performance/, 2018.

[13] S. Ruder, M. E. Peters, S. Swayamdipta, and T. Wolf. Transfer Learning in Natural Language Processing. In *ACL: Tutorials*, pages 15–18, 2019.

[14] D. Saha, A. Floratou, K. Sankaranarayanan, U. F. Minhas, A. R. Mittal, and F. Ozcan. ATHENA: An ontology-driven system for natural language querying over relational data stores. *VLDB*, 9(12):1209–1220, 2016.

[15] P. G. Selinger, M. M. Astrahan, D. D. Chamberlin, R. A. Lorie, and T. G. Price. Access path selection in a relational database management system. In *SIGMOD*, pages 23–34, 1979.

[16] J. Sun and G. Li. An end-to-end learning-based cost estimator. In *VLDBJ*, volume 13, pages 307–319, 2020.

[17] I. Trummer. Can deep neural networks predict data correlations from column names? In *https://arxiv.org/pdf/2107.04553.pdf*, pages 1–12, 2021.

[18] I. Trummer. The case for NLP-enhanced database tuning: towards tuning tools that “read the manual”. In *PVLDB*, 2021.

[19] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017-Decem(Nips):5999–6009, 2017.

[20] 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 *EMNLP*, pages 38–45, 2020.

[21] L. Woltmann, C. Hartmann, M. Thiele, and D. Habich. Cardinality estimation with local deep learning models. In *aiDM*, 2019.