Automated Paraphrasing for Pure Natural Language Interaction with Databases

Christopher Baik
cjbaik@umich.edu
University of Michigan
Ann Arbor, MI

Retrieving data from a relational database is a challenge for non-technical users. While SQL is a formidable swiss army knife for database tasks, even its core functionality, such as the ability to join tables, remains opaque for many users. This challenge is exacerbated by the fact that many production databases have complex schemas. Consequently, several lines of research [3, 8, 9, 11] have focused on helping less-technical users access databases via a natural language interface to database (NLIDB).

Several NLIDBs [3, 8, 10] follow a common interaction model:

N1. The user issues a free-form natural language query (NLQ) describing their request.
N2. The system generates candidate SQL queries from the NLQ.
N3. The user selects one of the SQL queries (or an intermediate representation of the SQL [3]), or returns to **N1** if none of the SQL queries match the request.
N4. The system executes the SQL query and returns an answer.

One pitfall of this interaction model is in **N3**, where the user is expected to select the correct SQL query from a list of candidates. Given that one of our goals in developing an NLIDB is to assist non-technical users without knowledge of SQL, it is contradictory to expect the user to understand SQL queries during the interaction.

Alternatively, an ideal interaction model would allow users to interact with the system ***completely in natural language***:

I1. The user issues a free-form NLQ describing their request.
I2. The system uses the NLQ to generate candidate paraphrases which each unambiguously refer to a unique SQL query.
I3. The user selects one of the paraphrases.
I4. The system executes the SQL query and returns an answer.

However, there are three challenges in accomplishing the ideal model. First, each system-generated paraphrase in **I2** must be *comprehensible* by humans familiar with the relevant domain, which requires correct use of terminology and grammatical correctness. Second, each paraphrase should be *unambiguous*. This entails that for a given database, a paraphrase should refer to exactly one SQL query, and when multiple paraphrases are presented to the user, they should be clearly differentiable by a human. Finally, each paraphrase should be *valid input* to the system—i.e., if the user copies and pastes the paraphrase into the system in **I1**, the system should return the SQL referred to by the paraphrase.

To solve these challenges, we propose a fully-automated system named Clement (**C**ontinuously **L**earning **E**ngine with **M**utating **E**xplanations for **N**atural Language Ques**T**ions). Clement leverages an existing NLQ-to-SQL engine, treating it as a replaceable black box which takes an NLQ as input and returns the highest confidence SQL query and a confidence score in the range $[0, 1]$ as output. Clement explores *mutations* on the original NLQ, using the output of the NLQ-to-SQL engine as a guide, to generate a set of candidate paraphrases for the user's request. Clement *continuously learns* from online user feedback on the quality of the paraphrases.

To ensure comprehensibility, mutations are guided by a publicly-available paraphrase[1] database as well as previously-issued NLQs. To keep paraphrases unambiguous, the semantic distance between candidate paraphrases are maximized, and paraphrases are adorned with clarifying phrases to distinguish them from closest-neighbor paraphrases. Clement guarantees that generated paraphrases are valid input to the system by maintaining a cache which also acts as training data for the internal NLQ-to-SQL engine.

Clement is an improvement over previous paraphrasing approaches [1, 4–7], which require users or administrators with schema knowledge to manually create a translation table or knowledge base. [2] generates paraphrases that flesh out the user's original NLQ with values from the database, but fails to achieve the unambiguous or valid input criteria.

REFERENCES

[1] H. Amano and Y. Kambayashi. Translation of sql queries containing nested predicated into pseudonatural language. In *DASFAA*, pages 116–125. World Scientific, 1991.
[2] D. Deutch, N. Frost, and A. Gilad. Provenance for natural language queries. *Proceedings of the VLDB Endowment*, 10(5):577–588, 2017.
[3] F. Li and H. Jagadish. Constructing an interactive natural language interface for relational databases. *Proceedings of the VLDB Endowment*, 8(1):73–84, 2014.
[4] J. Ljungberg. Paraphrasing sql to natural language. In *Intelligent Text and Image Handling-Volume 2*, pages 790–808. Le Centre de Hautes Etudes Internationales d'Informatique Documentaire, 1991.
[5] B. G. Lowden and A. N. De Roeck. The remit system for paraphrasing relational query expressions into natural language. In *VLDB*, pages 365–371, 1986.
[6] W. Luk and S. Kloster. Elfs: English language from sql. *ACM Transactions on Database Systems (TODS)*, 11(4):447–472, 1986.
[7] E. M. Mueckstein and G. D. Moerdler. Semantic interpretation of a database query language. *Data & Knowledge Engineering*, 1(2):123–138, 1985.
[8] A.-M. Popescu, O. Etzioni, and H. Kautz. Towards a theory of natural language interfaces to databases. In *Proceedings of the 8th international conference on Intelligent user interfaces*, pages 149–157. ACM, 2003.
[9] D. Saha, A. Floratou, K. Sankaranarayanan, U. F. Minhas, A. R. Mittal, and F. Özcan. Athena: an ontology-driven system for natural language querying over relational data stores. *Proceedings of the VLDB Endowment*, 9(12):1209–1220, 2016.
[10] N. Yaghmazadeh, Y. Wang, I. Dillig, and T. Dillig. Sqlizer: query synthesis from natural language. *Proceedings of the ACM on Programming Languages*, 1(OOPSLA):63, 2017.
[11] T. Yu, M. Yasunaga, K. Yang, R. Zhang, D. Wang, Z. Li, and D. Radev. Syntaxsqlnet: Syntax tree networks for complex and cross-domain text-to-sql task. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1653–1663, 2018.

[1]"Paraphrase" here refers to synonymous natural language phrases, not to the alternative query interpretations in our context.