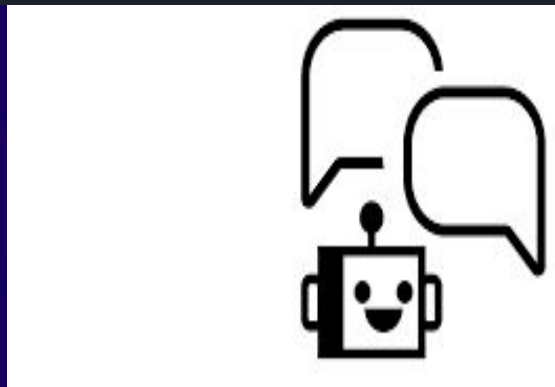
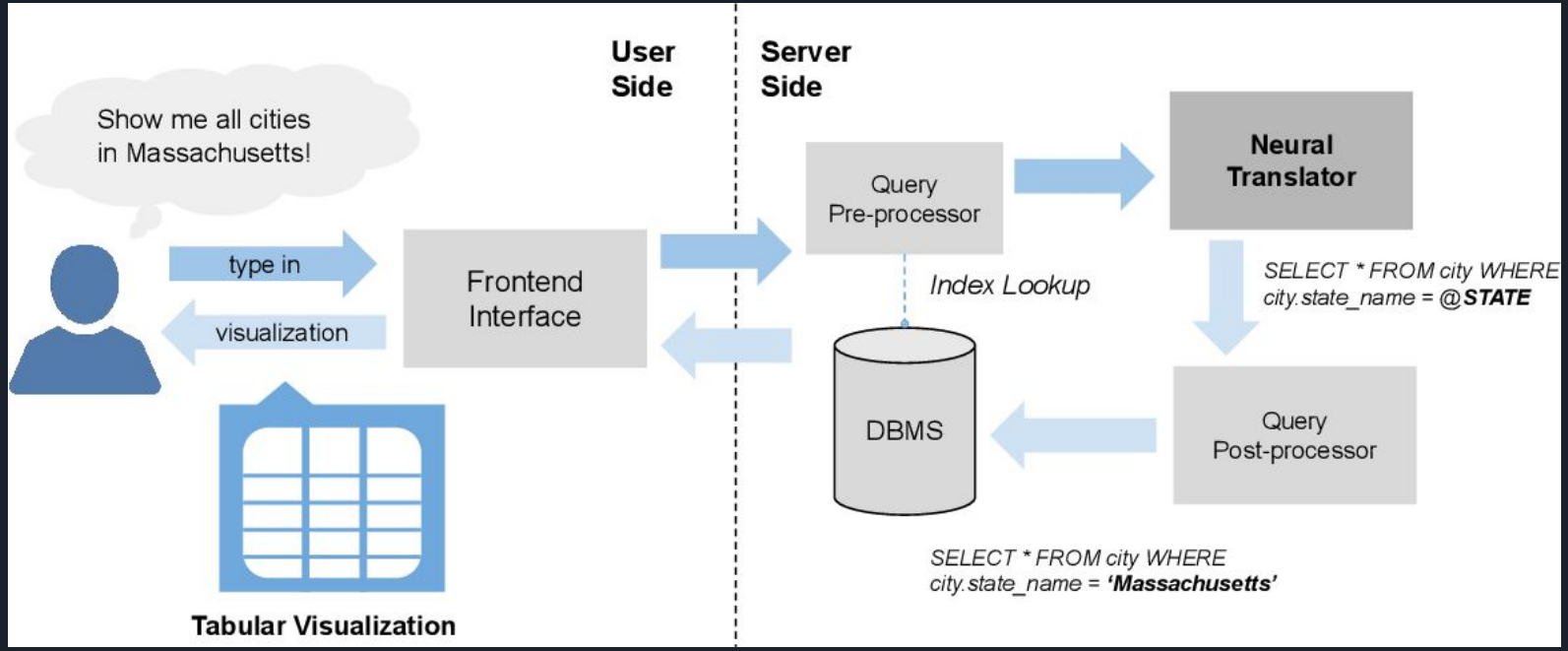
NL2QL

Kevin Allegretti

3/31/22



NL2SQL system (simplified)



Questions are given by the user via speech and is translated into database queries through different processes to give the user the data they ask for from their question.

Early History

1980

- Intermediate logical representation
- Qnl was translated into logical queries regardless of the database
- These queries then are converted into database queries.
- Depends upon hands-on translation so there was still room for massive improvement

2000

- Advanced rule-based methods
- “Off-the-shelf” language dissector to merge the methods in natural language without a specific database needed to train the merger.
- To increase the language threshold for the translator, there was a ranking-based approach. They created multiple candidate mappings and ranked their scores for each candidate on the words that were mapped between the question and the database elements.

Recent history

- Using advanced deep learning technologies, Deep-learning-based methods were introduced to the NLP community
- An interactive learning algorithm and a data augmentation technique using templates and paraphrasing were used by NSP.
- Other organizations also brought in the use of data augmentation techniques and learning algorithms
- published in 2017, WikiSQL is a benchmark to NL2SQL which serves as a dataset.
- Seq2SQL, SQLNet, Coarse2Fine and STAMP all improved upon the deep learning models tailored to WikiSQL.
- The latest learning algorithm called “meta learning” was introduced by PT-MAML
- NaLIR, ATHENA, SQLizer, and Templar all brought unique advancements in the optimization of translation.
- Most recent advancements involve TypeSQL corresponding data types and column names to the words in the question and using them as inputs into a deep learning model.
- The most recent benchmark created and used is Spider

Current Systems and capabilities - what they do

The current NL2SQL systems effectively take in input, translate it into SQL queries and output the desired items from a specific database.

Phases from current systems involve input, input enrichment, translation, post translation, training, and output.

Most models follow a similar process to effectively encode and decode natural language into SQl.

This is helpful for users who are not familiar with SQL or how to query for results in databases. The natural language that is turned into SQL code is fit for those to speak simple questions for what they are looking for.

Current Systems and capabilities - how they do it

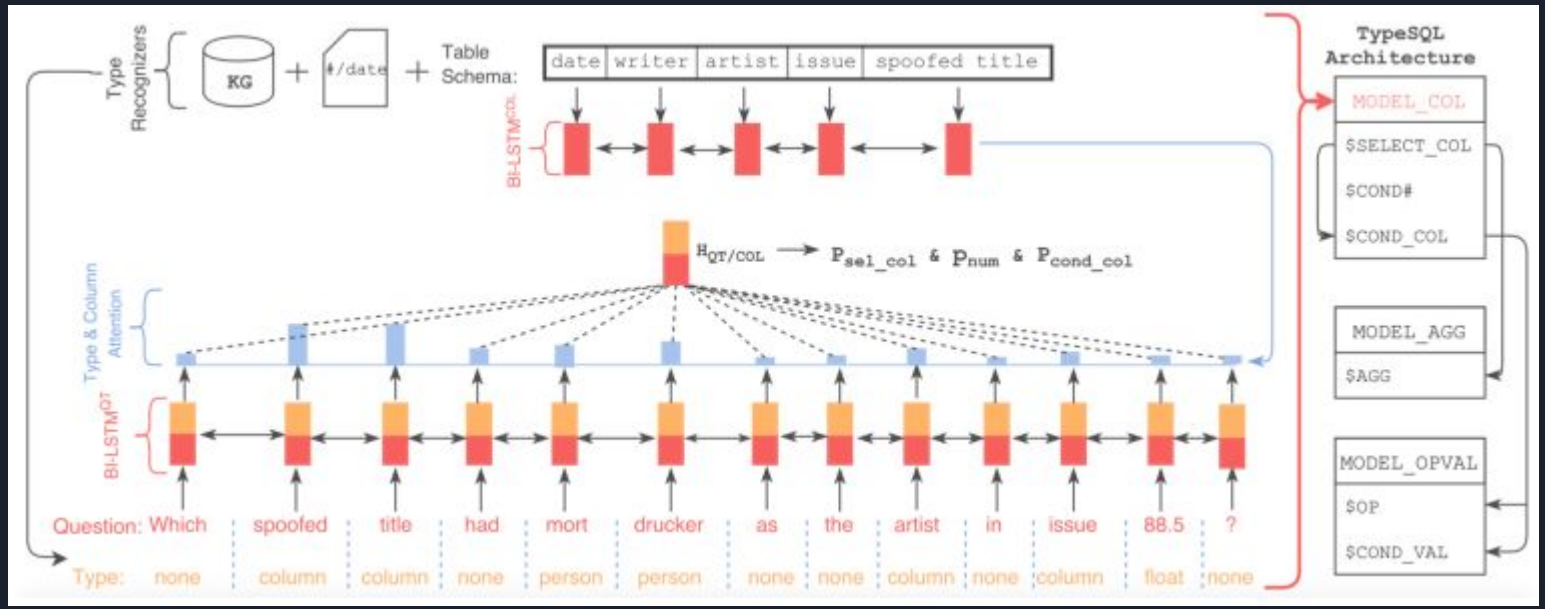
The latest NL2SQL model breaks down the different phases into subdimensions. For input enrichment, tagging, linking, and anonymizing are the processes. Translation is either rule-based, or deep-learning based and goes through the integration on different databases with generating the SQL queries. This is an extensive process with a multitude of methods.

- In deep-learning based, it uses NSP and DBPAI for sequence to sequence with database schema as part of output vocabulary, Pt-MAML and STAMP for sequence to sequence with the database schema as an input, IRNet and GNN for sequence to tree, Seq2SQL, SQLNet, TypeSQL and DialSQL for slot filling.
- In Rule-based, it uses ATHENA for sequence to tree, NaLIR for tree to tree, and SQLizer for tree to sequence mapping.

Post translation will be used after this to fill in any missing information or SQL code. Then the systems are trained with learning algorithms to compare the SQL with the natural language and develop patterns. Finally, the output of the SQL code will show data tables of the specific database that is used.

Current Systems and capabilities - how they do it

To examine one of these methods further, slot filling with TypeSQL will use semantic parsing for more accuracy in translation.



What’s next? (Features coming soon)

- Increasing model sizes
- Increasing computational efficiency
- Switching models between domains
- Fine tuning and sample efficiency
- Models producings consistent constant values
- Inductive biases

Research opportunities

Supporting linguistic diversity and handling long, multiple sentences are two areas that have relatively poor results in most methods used. Although it is incredibly complex, these notions will be a significant catalyst for NL2SQL if it is advanced further. For international business, the linguistic diversity would help communicate information from databases readily with others who speak a different language. In addition, if multiple, longer sentences were also incorporated, this would broaden the use to others who don't know how SQL works and won't have to tailor their sentences closer to the queries. There are also improvements to be made in adaptability across domains whilst keeping accuracy in single ones. Techniques such as TypeSQL and GNN are only suitable for some benchmarks. Improvements on string matching for unique databases is a research opportunity as well.

How would I approach the problem?

My first intuition is to map natural language from the relative frequency of words given, to a generative artificial intelligence SQL query. The relative frequency of words will be its own database for the words that are used given by the user from the speech to text function with a similar encode, decode model system for inputs of natural language. The key words, will be SQL functions, data types, column names, and possibly person names (language parsing). The generative artificial intelligence will be programmed to constantly be producing different forms of SQL queries in a given database. I can use Seq2SQL for the queries. Then, the model would map these queries with the words that are frequently used when asking for the results of the queries through a validation phase. The more frequent the results are correct, the stronger the connection will be between the words used and the queries used along with it. This will provide room for slight changes in semantics and even in the desired results.

references

http://www.vldb.org/pvldb/vol13/p1737-kim.pdf

https://www.semanticscholar.org/paper/DBPal%3A-A-Fully-Pluggable-NL2SQL-Training-Pipeline-Weir-Utama/ca03e74c4ee41a79b0a4ad678d7ee1df110beca8

https://arxiv.org/abs/1804.09769

https://link.springer.com/chapter/10.1007/978-3-030-86517-7_21

https://docs.google.com/presentation/d/1k3Npb47q5_p2cnY0lvLwwSdS0tHg3ctJLUz1s6vgDLk/edit#slide=id.g6e7c1f8a8a_1_79