A Survey of Natural Language Processing Implementation for Data Query Systems

Albert Wong
Mathematics and Statistics
Langara College
Vancouver, Canada
0000-0002-0669-4352

Dakota Joiner
Computer Science
Okanagan College
Kelowna, Canada
0000-0002-3094-0015

Chunyin Chiu
Mathematics and Statistics
Langara College
Vancouver, Canada
0000-0002-5932-5390

Mohamed Elsayed
Mathematics and Statistics
Langara College
Vancouver, Canada
0000-0002-7624-6117

Keegan Pereira
Computer Science
Okanagan College
Kelowna, Canada
0000-0002-2893-3406

Youry Khmelevsky
Computer Science
Okanagan College
Kelowna, Canada
0000-0002-6837-3490

Joe Mahony
Research and Development
Harris SmartWorks
Ottawa, Canada
JMahony@harriscomputer.com

***Abstract*—With increasing complexity and volume of collected data continuing to rise, it is becoming ever more important to develop systems with high interactability. Businesses with an interest in big data continue to seek solutions that limit cost while providing effective, simplified solutions to current issues in data retrieval. Combined analysis and application of a multi-factorial system will likely lead to promising results in ease of reporting of complex data by nontechnical end users. This survey is focused on natural language processing (NLP) implementations for data**

data query systems. Through the review, it will be able to identify and appreciate the various issues involved in such efforts as well as the ideas and tools currently available. We also briefly review the work on the building of a DW in support of the NLP implementation. Unless explicitly stated otherwise, NLP is discussed here under the guise of applied machine learning for Energy Information Systems.

II. Approaches in Converting an NL Query

There are two distinct approaches in converting an NL query into a database query: rule-based algorithms [3]–[11], and ML algorithms [12]–[27].

A. Rule-Based Algorithms

word embedding layer can provide word association and similarity to the deep learning model before any training process. Vathsala shows that using "a pre-trained word embedding layer (GloVe)" [32] can improve the accuracy rate by around 3% [33]. Instead of using word embedding layer, some researchers also considered a BERT [34] pre-trained layer as a first layer [12], [13], [19], [26], [35]. Pal in [36] used another optimized BERT layer – RoBERTa instead of just the original one.

Most of the papers focus on the database query generation based on one single natural language query. Zhang developed a database query generation model which can consider multiple natural language queries, previously generated queries, and the database schema with separate encoders [35]. This model utilizes turn attention to store the hidden state of the historical messages and generates a new database query based on previous messages and the new message.

2) Input and Output Adjustment of the Machine Learning Algorithms: Authorths in [23] showed that there is a need for a pre-processing step on the natural language query. Guo classified the tokenized words into groupings related to either table names, column names, or values. Brunner classified the token into five groups: table, column, value, aggregation, and superlative. Apart from this, the database elements (table and column) are also classified whether they are an exact match, partial match, or value candidate match with the tokenized words [24]. In both papers, the classified result becomes a part of the input of their algorithms to generate an intermediate representation that shows the linkage between the natural language query and database query, called SemQL [23]. Other than scanning the whole database, Ma [26] suggested using an aligner, an unsupervised learning method served as an automated annotator for the classification of tokens.

Apart from using a natural language query as input, researchers also want ML algorithms to consider database elements in order to develop queries that can select the correct target column. The most common method is to use the column names as a part of input [12]–[14], [20], [25], [26]. However, this method is only suitable for data sets that have a single table. In reality, databases contain large amounts of data employing complicated schema with multi-table structures. Therefore, research suggests making the ML models learn the entire structure of the targeted databases. Bogin utilized "a graph neural network (GNN)" [37] to process the database schema and make the algorithm generate a database query with join-clause [22]. To increase the quality of generated database queries, Wang suggests an execution guidance mechanism [38]. The mechanism serves as an output monitor of the generated database queries. It can detect and reject some non-executable queries in the middle of the decoding process. His work shows that this mechanism can improve the accuracy of currently existing algorithms up to 6.4% higher.

III. Discussion of Existing Machine Learning Solutions in Industry

The ML approach to convert a NL query to a database language such as Structured Query Language (SQL) is a considerable development from the normal semantic approach [15], [17]. Different deep learning models such as SQLNet [14], Seq2SQL [20], SyntaxSQLNet [21] and F-SemtoSQL [19] were used on different datasets such as WikiSQL, Spider, Geo and ATIS and will be discussed in the following.

One notable artificial neural network approach to resolve the problem of converting NL to SQL is to use an encoder-decoder architecture to compete against semantic analyzers [17].

Seq2SQL is "a deep neural network architecture" that is associated with a rule-based reinforcement learning algorithm. It is formed using three parts: "the aggregation operator, the SELECT column, and the WHERE clause". Models are built using PyTorch and the training is supervised using reinforcement learning that rewards the decoder when one of the serializations is produced. "This type of model has shown to produce very limited results" [17].

Mellah et al. [17] mentioned SQLNet, suggested by [14], a model which utilizes a sketch-based architecture and takes the "structure of an SQL query, and generates it from a dependency scheme". This approach can provide a higher level of accuracy (by 9% to 13%) than older techniques on the WikiSQL dataset.

Mellah et al. [17] also mentioned another model, TypeSQL, that was suggested by Yu et al [39]. This model again utilizes a "sketch-based approach and treats the task as a slot filling one by grouping various slotngth445(s415(th44ln0Hr4E1u5(ario%uip-445o41 [19erationTdolumdol-319(60sed)-327(dol-401359sa19(60s3the))-5-ed dol

masked mechanism performs considerably better than other models described earlier. Li's team in [19] observes that "F-SemtoSQL outperforms the previous best work by 10.58% to 41.68% in accuracy". Additionally, they observed higher accuracy in complex event areas over all compared SQL queries on the test datasets, which included Spider (37.6% development, 41.7% test), WikiSQL (81.8% development, 78.6% test), ATIS (90% development, 87.8% test), and GEO (90.2% development, 89.7% test).

Xu [40] created a new model named NADAQ by considering the previous approach of using grammar structure and integrating it into a sequence-to-sequence (seq2seq) model. The querying of the database in this system combines deep learning with common parsing techniques used by traditional databases. The main structures of this system are speech recognition, translation, rejection, recommendation, and results display [40]. Overall, the NADAQ surpasses in accuracy both older conv-seq and attention-seq systems by a considerable margin due to its advantage of using grammar states, thus saving the wasted work on grammar understanding. The classification accuracy rate for NADAQ in terms of F1 is 0.839. The F1 score of NADAQ on both IMDB and GEO datasets are higher than 80%, which is near the industrial standard [40].

Mellah suggested an approach [27] inspired from the previously mentioned SQLNet [14]. His new methodology is based on classifications, word embedding, and the use of an RNN, specifically LSTM and GRU. The underlying concept is to create a sketch that will produce the query from the natural language with slots to be filled. Neural networks are employed for the prediction of the content of each slot in the sketch. The suggested model can be divided into five modules each of which has a prediction component. The first module is responsible for aggregation, the second and third will find the select and condition columns, respectively. The fourth module will define the correct operation. These four are handled as a classification problem, while the last module will "predict the value of the condition in the where clause. The user query and the schema tables are considered as token sequences." The accuracy of the five modules over the used dataset, which was inspired from WikiSQL, are shown in Table I.

TABLE I
THE ACCURACY OF MODULES OVER USED DATASETS [27]

Modules	Training accuracy	Testing accuracy
AGG	92%	89%

dataset has limited uses for developing practical ML models as it includes "queries with only one column for the SELECT clause and one table in the FROM clause". However, it has been used often in NLP and ML research.

3) GeoQuery: Geoquery, or GEO, is a small dataset that contains information on the geography of the United States, such as states, cities, rivers, and mountains [43]. According to the Computer Science department at the University of Texas, the dataset contains 880 queries in NL and the corresponding queries in a formal query language. It is considered a fixed-schema dataset that can be used for the development of ML models for complex and composite queries over a closed domain [19].

4) ATIS: ATIS is commonly used to evaluate semantic analysis systems [44]. This dataset contains two subsets for training and two for testing. It is also considered as a fixed-schema dataset similar to Geoquery [19].

5) SParC: SParC is a dataset that has context-dependent queries under a cross-domain setting based on Spider [42], [45]. The semantic meaning of the latest queries are dependent on the previous query in a context-dependent querying domain. Interestingly, ATIS [44] is also a context-dependent data set, but it is limited to a specific flight database domain. SParC provides context-dependent queries with 200 databases and its complexity is considerably higher than that of ATIS.

V. DW and NLP Integration

A. NLP use for Database Search Current State

Much of the current state of applied NLP use with database and DW systems occurs in the medical field. Electronic medical records and the databases in which they are stored are scanned using NLP algorithms to aid in differential diagnosis of patients presenting with potentially unclear pathologies. The use of NLP has aided in diagnosing fatty liver disease [46] and identifying smoking status [47], among others. Researchers often choose to use the designed-for-medicine analytical algorithms "CLAMP (Clinical Language Annotation, Modeling, and Processing)" [48] and "cTAKES (clinical Text Analysis and Knowledge Extraction System)" [49]. Frequently, these tools are used together to produce actionable results. Each builds from an ML basis with tactics discussed earlier including a sentence boundary detector, tokenizer, part-of-speech tagger, a parser, and an encoder. CLAMP, as the newer procedure, expands on these further with additional functionality in recognizing acronyms and shorthand, assertion and negation detection, and a complicated rules engine allowing the user to define their own rules to fine tune searching, querying, and performance. Though the structure of medical databases, data warehouses, and electronic medical records vary drastically from classical database structures, the impact and implication of applied ML affords support for pathways of development in many other fields.

Outside of the medical industry, there have been developments in NLP in a handful of other areas. O'Halloran, Pal, and Jin [50] have worked together to utilize NLP tools in a multimodal analytical platform to scrape and analyze information from many media websites including Facebook, Twitter, and Reddit, among others. Rather than using NLP to translate text to SQL, their work aims to allow for the use of NLP as a tool to more accurately associate text to other related multimedia formats. Wei, Trummer, and Anderson of Cornell University [51], [52] have put forward interesting research this year where they aim to not only optimally translate text or spoken language to SQL queries, but to visualize it to the user to aid in choosing the correct queries. Following

Both the P-HYBRIDJOIN and QaS-HYBRIDJOIN result in approximately three orders of magnitude higher tuple uploads per second ($7x10^6$ and $8x10^6$, respectively) over previously identified optimized hybrid join methods from the literature. The observed increase results from decreasing I/O cost through dissolution of interdependent processes on the disk buffer loading and probing phases. More simply, a multi-disk buffer facilitates parallel loading resulting in reduced I/O cost.

C. NLP to SQL Code Generation

There seems to be great difficulty in *fully* correct NLP translation, but several groups, including Li [19], Karthik [57], and Vathsala [33], have been working to develop systems that more accurately translate spoken or written text into executable SQL statements that retrieve the correct and desired data. As described above, Li et al. have worked recently to develop F-SemtoSQL, a slot-filling method where relationships between attributes are captured "by grouping the different slots in a graph dependency way" [19]. As has been recognized in many NLP tasks, most spoken sentences or statements are filled with words that are not used or are not relevant to building out an SQL query. To minimize an potential interference by non-useful words, specific slots in basic SQL syntax were identified as needing to be filled to generate the correct query (see Listing 1):

Listing 1. An Example of Specific Slots in Basic SQL Syntax [19]

```
SELECT $ AGG $ COLUMN
       WHERE $ COLUMN $ OP $ VALUE
```

Y5(R).6]8w QBT/F9 7 Tf/DeviceRGB CS0 0 0 SC10 TL49.8175 21.908 Td(Authorized licensed ur2he)iited to: The Univers

Advancements in data warehousing also suggest promising results in facilitating a true whole-system solution incorporat-

Language Processing, Proceedings of the Conference, 1724–1734. https://arxiv.org/abs/1406.1078v3.

[30] Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to Sequence Learning with Neural Networks. Advances in Neural Information Processing Systems, 4(January), 3104–3112. https://arxiv.org/abs/1409.3215v3.

[31] Luong, M.-T., Pham, H., & Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. Conference Proceedings - EMNLP 2015: Conference on Empirical Methods in Natural Language Processing, 1412–1421. https://arxiv.org/abs/1508.04025v5.

[32] Pennington, J., Socher, R., & Manning, C. D. (2014). GloVe: Global Vectors for Word Representation. EMNLP 2014 - 2014 Conference on Empirical Methods inon EmpiricalmpiricalO3.