2021 IEEE International Conference on Recent Advances in Systems Science and Engineering (RASSE) | 978-1-6654-3441-6/21/$31.00 ©2021 IEEE | DOI: 10.1109/RASSE53195.2021.9686815

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 query systems, especially related to massive data sets (1TB+) in OLTP databases, OLAP databases, and data warehouses. We are seeking the most up-to-date and effective uses of NLP for Speech-to-SQL and Text-to-SQL generation, and the most recent advancements in data warehousing to optimize ELT efficiency and data retrieval, focusing on the highest performing code implementations on the Spider and WikiSQL datasets. Many models, including sequence-to-sequence (seq2seq), sequence-to-SQL (Seq2SQL), and fuzzy semantic to SQL (F-Semtosql), among others, are briefly described and compared. As well, recent advancements in data warehousing technology like multi-disk buffering in the ELT process and hybrid multi-dimensional and relational OLAP databases (HOLAPs) are discussed. The learning gathered here is applied to fill a gap in the current industrial knowledge base in service of increased efficiency in data access, retrieval, and reporting in a customer-facing environment.**

***Index Terms*—Natural Language Processing, Data Query System, Text-to-SQL, Speech-to-SQL, Deep Learning, Machine Learning, Human-Machine-Systems, Energy Systems**

I. Introduction

As the size of data sets become massive (1TB+), classic relational OLTP databases start to lose efficiency for information retrieval. OLAP and data warehousing (DW) has become a popular solution to maintain efficiency for big data. The structure of data warehouses lend themselves well to handling complex queries based on aggregating data for reporting purposes, often with a business intelligence (BI) tool [1]. With increasing complexity of business reports necessary to retrieve the desired information, businesses observe a mirrored, marked increased in the skill, time, and commitment necessary from their knowleadge workers and data engineers to generate these reports. Unsurprisingly, current solutions to these problem areas are becoming prohibitively expensive and require the development of speech-to-query solutions with respect to energy information systems simplifying the report generation and providing real-time reports to the customers.

Machine learning (ML) has stormed the world, gaining traction and applicability in almost any imaginable computing situation. The proportion of end users in industry who know Structured Query Language (SQL) is relatively small, thus pinpointing an area for ML to aid in making business decisions. Turning human speech into machine language, known as NLP, is not a new idea as its first development began in the 1950s. Through advancements in computational power and the development of increasingly more sophisticated ML algorithms, NLP has progressed by leaps and bounds. By simplifying the point of interaction between end users and DW, newly developed tools will allow basically any user to retrieve the desired information from an information system or a DW. It will reduce wait times for reporting, decrease costs normally associated with information retrieval, and generally facilitate the decision-making process.

The ideal NLP algorithm for SQL generation should be able to understand "a natural language (NL) query" and "translate it into a database query" [2]. Also, it should correctly identify the tables and columns, perform any necessary joins, aggregate data if needed, and be error-free. An error-catching mechanism must be considered, as well as clarification capabilities. Additional reporting functions are required for a total system design, but are not necessary as part of the algorithm design.

In this paper, we identify and review various research efforts and recent advances in the area of NLP implementation for

We acknowledge the support of the Natural Sciences and Engineering Research Council of Canada (NSERC), Harris SmartWorks Division of Harris Computers, Okanagan College, and Langara College.

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

The structure of a rule-based algorithm can be broken into three distinct components: an NLP processor, a mapping table, and a mapping and query generation engine. An NLP processor first breaks a natural language query into words (objects, verbs, and associated qualifiers such as where, how, how many, etc.). A mapping table, created by the researcher in advance, is a reference for converting these words into appropriate elements of the subject database (table names, column names, or values). A mapping and query generation engine produces the database query by matching the output from the NLP processor and words from the mapping table.

Note that the NLP processor is responsible for the the processing of a NL query regardless of whether the algorithm is rule-based or not. Essentially, an NLP processor involves the following four components: tokenization, stop word removal, word replacement, and parsing.

1) Tokenization: The tokenization component splits a natural language query into a sequence of single words (tokens). Authors in [4], [5] suggest that tokens can be further classified into several groups. This additional step can speed up the matching process against the mapping table and reduce ambiguity in a rule-based algorithm. The number of classified groups varies. Agrawal in [4] described three groups: reserved keywords (wh-clause), database attributes, and values. On the other hand, Sontakke [5] proposed the classification using more groups: table names, column names, condition, values, command names, operation names, and non-useful words.

2) Stop Words Removal: It is usually performed on the tokenized natural language query. Authors in [6] suggested to decide what type of words should be kept; for example, only nouns, proper nouns, adjectives, and numbers are kept after the stop words elimination. Other authors proposed their own unwanted words list/ignore list that included prepositions and frequently used nouns [7], [8].

3) Word Replacement: This is a process to convert words from NL into those that fit within a database context. Several approaches in word replacement may be used. Stemming is the process that identifies the root word by removing any suffix(es), including *-s*, *-es*, *-ed*, *-ing*, and *-er* etc., in each tokenized word [3], [7]. Uma in [10] stated that lemmatization is more preferable than stemming because of the fact that the removal of a suffix in stemming does not guarantee the generation of an accurate basic form of a word that exists in a dictionary. For instance, "studied" will be converted as "studi" by stemming, but this word does not exist in the dictionary. But, "studied" will be converted to "study" if lemmatization is used; a more accurate result. Word replacement may also be accomplished through synonyms and hypernyms [3], [8]. For example, the word "employee" can be used to replace a word "staff" (by synonyms) or "office assistant" (by hypernyms). This approach can be used in combination with either stemming or lemmatization: The word "workers" can be replaced as "worker" by stemming and the stemmed word "worker" can be replaced as "employee" [6]. This word replacement process is important as it provides better matching of words in NL with those associated with the target database and therefore helps to improve the relevance and accuracy of the converted query.

4) Parsing: A parser is used to generate a parse tree from a natural language query [4], [6], [7], [9]. A parse tree is a rooted tree that shows the structure of a sentence based on its syntax. Through the parse tree, an NLP processor can understand the relationship or words association; the identification of a verb or noun phrase. A parsing component provides additional information and therefore better understanding on the structure of a natural language query. Depending on the needs of the algorithm design, the NLP processor implements some or all of these components.

From a performance perspective, authors in [4], [5], [8] claimed that their rule-based models were successful in converting natural language requests into valid database queries. The algorithm developed by Uma [10], which can handle two types of specific question in terms of NL, has an 98.89% accuracy rate in 2,880 test cases on a very limited data set made to test their model.

B. Machine Learning Algorithms

ML algorithms application is another approach in NLP implementation for converting natural language requests into queries. A NL request is split into a series of words and serves as input. Then a trained ML algorithm, most likely a deep learning algorithm, produces the three main components required to form a database query as output: selected attributes/columns, aggregate operator, and filtering conditions.

1) Model Architecture for Database Query Generation: As the input format is a sequence of words, a recurrent neural network (RNN) can effectively handle this format. The characteristics of RNN models are 1) able to store the effect from the past input and 2) able to update the hidden layer based on the past effect and the new additional input from the sequence. Given these, an RNN becomes a basic architecture in the algorithms on natural language queries to database queries. The commonly used models including: "long short-term memory model (LSTM)" [28], the "gate recurrent neural network (GRU)" [29], the "sequence-to-sequence model (seq2seq)" [30], and the attention model [31].

Word embedding layers are widely used as the first layers in the deep learning models [14], [16], [19], [20], [27], [33]. The

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 slots and finding relationships between attributes". This model is 3.5% more accurate than the SQLNet model and demonstrates an improvement in accuracy of 2% on "the WikiSQL dataset leading to a final performance improvement of 5.5%".

SyntaxSQLNet [21] is solving the problem with high-efficiency complex and cross-problem domain text-to-SQL generation tasks. The concept is based on a "tree-based SQL decoder and table-sensitive column attention encoders" [19]. This model is capable of solving nested queries on databases and was tested on the Spider dataset. Practical experiments proved that this model can handle numerous complex SQL queries while increasing the accuracy compared to other models by 7.3%. All of the "source code of the above models are available on Github" [17].

Finally, Li et al. [19] proposed F-SemtoSQL, a model created using Python 3.6, PyTorch, two bidirectional LSTM encoders, and BERT. The model trials are performed on the Spider dataset and validated on WikiSQL, ATIS, and GEO. This model is evaluated based on the accuracy of SQL query conversions, which are generated by atomic, composite, aggregate, and complex (ACC) events as defined by the author. F-SemtoSQL demonstrates the best results on different datasets compared to other techniques, such as SyntaxSQLNet. Its main advantage is the ability to predict complex events while maintaining an accuracy rate that is unaffected by the issue of direct sequencing. Furthermore, "using the attention mechanism allows the model to concentrate on various words and sub-level events". It is important to note that this model uses "the database content to further understand the user query in case it was not written accurately". Utilization of a random

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%
SELCOL	71%	66%
CONDCOL	73%	67%
OP	99%	99%
VALUE	76%	70%

Another approach suggested by Zhang and Weiss [41] is to use an alternative neural network model that uses parsing and tagging including part-of-speech (POS) tags. This stackprop model is based on a feed-forward network that takes an embedded feature matrix as input and feeds it through a hidden layer and a softmax prediction layer. In terms of classification accuracy, this model surpasses all other models, such as the graph-based RGBParser, by a small margin when applied to the same test dataset (78.9% versus 77.6%). Furthermore, the system can be further improved by 2.3% in classification accuracy through the use of a POS tag. However, there is a limitation of the algorithm due to the use of a greedy parsing technique.

Ferreira et al. [16] proposed a model "that relies heavily on pre-processing and post-processing mechanisms". This approach employs a single attention mechanism model that starts with a "pre-processing step in which the sequence is sent to a neural network. Next is the post-processing step in which the final output is grouped and assembled." The pre-processing step will do the preparation for the prediction model by dividing the input string using various methods such as tokenization, removing stop-words, lemmatization, and vectorization. The goal of the model prediction phase is to take the input from the first phase and process it using an ML encoder-decoder architecture. The post-processing phase groups the output by using different techniques "such as vector-to-token conversion, replacement of placeholders, and labelling" to construct the database query [16].

In Kombade's ML approach [18], query prediction is done using multinomial logistic regression. The regression uses a more complex cost Sigmoid function. The concept is based on tagged tokens and mapping for noun(s). The verb list is prepared using iteration and the label encoder gives each token a unique identifier. The model is trained using a logistic regression algorithm. Hence, the model will be able to predict whether the NL statement should be converted to a SELECT statement or data manipulation language (DML) query. This is done based on certain words that appear in the initial query that have synonyms to those words. The accuracy of the proposed model is 98.65% successful in predicting the query type (either a SELECT statement or DML query) [18].

IV. MACHINE LEARNING ALGORITHM DATA SETS

A. Data Sets Overview

In testing all of these different models, many research teams choose to use standardized datasets to form a baseline for comparison. However some still construct their own training and testing sets. The most commonly used testing datasets are Spider, WikiSQL, ATIS, GEO, and SParC. Each will be briefly discussed to provide context for the previously mentioned model accuracy rates.

1) Spider: Spider is a relatively "new large-scale annotated text-to-SQL dataset" [42]. It consists of "10,181 questions, 5,693 unique complex queries, 200 databases, and 138 domains" [19]. A specific advantage of this dataset is that it is designed to mimic real scenarios including query nesting, multiple table joins, and different syntax usage. The dataset can also be divided into several subsets with varying degrees of difficulty in query conversion and can therefore be useful for development and test planning [19].

2) WikiSQL: WIKISQL is a mono-table dataset that contains "80,654 NL sentences with the corresponding SQL queries and 24,241 HTML tables from Wikipedia" [19]. This

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 a spoken query, the model presents text-to-SQL as a visual matrix/multiplot from which the correct, desired query can be selected for execution.

B. Recent Advancements in Data Warehousing

In consideration of a total business solution, not only ML and NLP must be considered to maximize efficiency; the development and utilization of a data warehouse reduces the time taken during the decision-making process [53]. Advances in data warehousing provide a parallel research and development opportunity. Taken in concert, a combined approach utilizing cutting edge research in both NLP and data warehousing is likely to provide the current most efficient and achievable data retrieval and reporting. Specific target areas include increasing efficiency in the extract-load-transform (ELT) process, increasing availability of near-real-time data access, and advanced, but simplified, structural schema design to limit joins.

To begin, Dhomne et al. [54] provide a well-rounded, if simple, design mechanism for natural language interface to database (NLIDB) systems. They point out that two major components comprise an NLIDB: a linguistic component and a database component. The linguistic component corresponds to NLP and is the subject of many recent studies, some of which have already been discussed [19], [33], [56], [57]. The database component has been well explored and many other teams continue to pursue both OLTP and DW solutions. The discussed NLIDB architectures serve as a solid framework on which to develop an in-house system.

Reddy et al. [58] discuss the basic principles of DW as a decision-making tool that is market-oriented. Further, Garani et al. [59] use a case study to demonstrate the idea that intelligent and meaningful design of hybrid multi-dimensional and relational OLAPs (HOLAP) can result in optimized efficiency in retrieval of integrated data. They present that a starnest schema results in an optimal design pattern for a DW to capture spatiotemporal data that is common in the utilities industry, for example. Increases in query runtime efficiency were observed in nested tables over flat tables, a fact supported by the number of rows accessed being lesser in the nested tables.

Aziz, Anees, and Mehmood [60] discuss a set of semi-stream join algorithms to provide increased efficiency in the ELT process through proposed "parallel hybrid join (P-HYBRIDJOIN)" and "Queue and Stack hybrid join (QaS-HYBRIDJOIN)" methods. The identified drawback in current hybrid joins is that "a single buffer is used to load the disk partition" resulting in sub-optimality due to an I/O bottleneck.

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
       (GROUP BY / ORDER BY)
```

"The slots ($ X_{id}) are identified as atomic events that can additionally be further drilled down to higher-level composite events, aggregate events, and complex events." These combos of events allow for more complex querying including options like ORDER BY DESCENDING, ORDER BY ASCENDING, GROUP BY HAVING, nested SELECTs, operators ($<$, $>$, $\geq$, etc.), and more. The improvements in accuracy of this method were previously noted in Section III.

Alternatively, Vathsala et al. [33] propose a different model to achieve high accuracy of translation and over traditional seq2seq models. SQL statements are instead generated using a policy gradient algorithm as outlined by Sutton in [55]. Their method incorporates Memory Augment Policy Optimization (MAPO) alongside a pretrained word embedding with GloVe and an epsilon greedy strategy. Using the WikiSQL data set, the model achieved a "dev accuracy of 76.8% and a test accuracy of 77.6%". These levels are consistent with Li [19] and Rubin [56]. Interestingly, their results demonstrate a markedly increased development accuracy as the number of training epochs increases, but present with a negative curvature suggesting an upper limit to the effectiveness of greater training epochs.

Karthik et al. proposed yet another method to convert human speech into machine readable queries using NLP techniques in a metadata-driven framework [57]. They point out early a major drawback to their approach is in that common mapping rules required to parse a tree structure to SQL will fail with complex queries. Specifically, finding the correct columns and tables from which to pull the data must be identified from an NL statement unambiguously. Their work focuses on metadata information to facilitate entity detection and table joins. They do not mention results against standard data sets like Spider or WikiSQL, but their ideas point toward a different approach path. The domain independence of this translation mechanism point to its cross-system portability and generalizability for non-technical end users.

Brunner and Stockinger [24] have developed specifically targeted tools dubbed *ValueNet* and *ValueNet light* that seeks to post high execution accuracy rates on the Spider data set. *ValueNet* and *ValueNet light* build on Abstract Syntax Trees (AST) rather than directly translating to SQL as in a seq2seq model to avoid the problems where an input user does not provide enough information to generate a full query. Using this model, they achieve accuracy rates ranging from 77% correct for easy queries to 43% for extra-hard queries, with *ValueNet light* achieving an approximate 4% higher accuracy rate on average over *ValueNet* for various training epochs. Specifically highlighting error analysis, they chose incorrect queries at random for more in-depth analysis to determine the root cause of incorrectness and found that in approximately 50% of those queries the wrong column was selected. Further, they noticed that of those 50%, another 25% selected the wrong columns in the wrong tables. The results they posted are promising and further external research can be carried out to correct the column selection errors as Brunner and Stockinger have posted their source code for independent peer-review.

VI. Conclusion

In this paper we discussed recent advances in NLP and DW as an integrated solution for information retrieving from DW using human speach.

The basic structure of NLP algorithms have been discussed and compared from both a rule-based standpoint and an ML standpoint. Rule-based NLP often follows the pattern of tokenization, stop-word removal, word replacement, and lemmatization. ML models are widely varied in their implementation and specifics, though most fall under four broad categories: "Long Short-Term Memory (LSTM), Gated Recurrent Unit Neural Networks (GRU)", Sequence-to-Sequence models (seq2seq), and attention models. The use of recurrent neural networks to store effects over time is widespread and facilitates the "learning" aspect of ML in these applications. Further advancement and accuracy are achieved in models employing additional training mechanisms like word embedding and output monitoring mechanisms like execution guidance. Among large testing data sets, such as WikiSQL and Spider, relatively high accuracy rates in the ranges of 65%+ have been achieved though there is yet room to improve accuracy rates. Identified areas for improvement include targeting column and table selection, slot filling for complex queries, and correctly identifying metadata information for SQL generation.

Advancements in data warehousing also suggest promising results in facilitating a true whole-system solution incorporating ML models. Hybrid OLAPs with starnest schema represent a target for increasing data retrieval efficiency aimed at limiting the number of table joins. Semi-stream join algorithms also increase efficiency by presenting an ELT process with an approximate three orders of magnitude faster load time resulting in near-real-time delivery of data from a continually updated data warehouse. Multi-disk buffering in yet even larger warehouses suggests continual improvements in decreasing load times in the ELT process.

In service of this review, we seek to implement a multi-factorial and integrated NLP and DW solution for real-time retrieval and reporting of business data from an Energy Information System. Our previous research areas [61]–[64], alongside this review, will facilitate this development. We seek to further develop accurate and responsive reporting models to contribute to the applied research community.

ACKNOWLEDGMENT

We acknowledge and thank Diomari Fortes (of Okanagan College) for his work in building a testing data warehouse and gathering research as a student and former member of our research team.

REFERENCES

[1] R. Kimball and M. Ross, The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling, 3rd Edition. 2013.

[2] Narechania, A., Fourney, A., Lee, B., & Ramos, G. (2021). DIY: Assessing the Correctness of Natural Language to SQL Systems. 26th International Conference on Intelligent User Interfaces, 597–607. https://doi.org/10.1145/3397481.3450667

[3] M. Samsonova, A. Pisarev, and M. Blagov, (2003). Processing of natural language queries to a relational database. Bioinformatics (Oxford, England), 19 Suppl 1. https://doi.org/10.1093/BIOINFORMATICS/BTG1033.

[4] R. Agrawal, A. Chakkarwar, P. Choudhary, U. A. Jogalekar, and D. H. Kulkarni, (2014). DBIQS - An intelligent system for querying and mining databases using NLP. Proceedings of the 2014 International Conference on Information Systems and Computer Networks, ISCON 2014, 39–44. https://doi.org/10.1109/ICISCON.2014.6965215.

[5] A. Sontakke and A. Pimpalkar,(2015). A rule based graphical user interface to relational database using NLP.

[6] A. Mohite and V. Bhojane, (2015). Natural language interface to database using modified co-occurrence matrix technique. 2015 International Conference on Pervasive Computing: Advance Communication Technology and Application for Society, ICPC 2015. https://doi.org/10.1109/PERVASIVE.2015.7087045.

[7] K. Javubar Sathick and A. Jaya, (2015). Natural language to SQL generation for semantic knowledge extraction in social web sources. Indian Journal of Science and Technology, 8(1), 1–10. https://doi.org/10.17485/IJST/2015/V8I1/54123.

[8] Kate, A., Kamble, S., Bodkhe, A., & Joshi, M. (2018). Conversion of Natural Language Query to SQL Query. Proceedings of the 2nd International Conference on Electronics, Communication and Aerospace Technology, ICECA 2018, 488–491. https://doi.org/10.1109/ICECA.2018.8474639.

[9] A. Malik and R. Rishi, (2015). A domain and language construct based mapping to convert natural language query to SQL. International Journal of Computer Applications, 116(4), 29–34. https://doi.org/10.5120/20325-2444.

[10] Uma, M., Sneha, V., Sneha, G., Bhuvana, J., & Bharathi, B. (2019, February 1). Formation of SQL from natural language query using NLP. ICCIDS 2019 - 2nd International Conference on Computational Intelligence in Data Science, Proceedings. https://doi.org/10.1109/ICCIDS.2019.8862080.

[11] Baik, C., Jagadish, H. V., & Li, Y. (2019). Bridging the Semantic Gap with SQL Query Logs in Natural Language Interfaces to Databases. Proceedings - International Conference on Data Engineering, 2019-April, 374–385. https://doi.org/10.1109/icde.2019.00041.

[12] Hwang, W., Yim, J., Park, S., & Seo, M. (2019). A Comprehensive Exploration on WikiSQL with Table-Aware Word Contextualization. https://arxiv.org/abs/1902.01069v2.

[13] Guo, T., & Gao, H. (2019). Content Enhanced BERT-based Text-to-SQL Generation. http://arxiv.org/abs/1910.07179.

[14] Xu, X., Liu, C., & Song, D. (2017). SQLNet: Generating Structured Queries From Natural Language Without Reinforcement Learning. Retrieved from https://arxiv.org/abs/1711.04436v1

[15] Ahkouk, K., & Machkour, M. (2019). Human language question To SQL Query Using Deep Learning. 2019 3rd International Conference on Intelligent Computing in Data Sciences, ICDS 2019. https://doi.org/10.1109/ICDS47004.2019.8942342

[16] Ferreira, S., Leitao, G., Silva, I., Martins, A., & Ferrari, P. (2020). Evaluating human-machine translation with attention mechanisms for industry 4.0 environment SQL-based systems. In 2020 IEEE international workshop on metrology for industry 4.0 and IoT, MetroInd 4.0 and IoT 2020 - Proceedings (pp. 229–234). Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/MetroInd4.0IoT48571.2020.9138181

[17] Mellah, Y., Ettifouri, H. El, Bouchentouf, T., & Belkasmi, M. G. (2020). Artificial neural networks for text-to-sql task: state of the art. In lecture notes in Electrical Engineering (Vol. 684 LNEE, pp. 557–565). Springer. https://doi.org/10.1007/978-3-030-53187-4_61

[18] Kombade, C. (2020). Natural language processing with some abbreviation to SQL. International journal for research in applied science and engineering technology, 8(5), 1046–1048. https://doi.org/10.22214/ijraset.2020.5166

[19] Q. Li, L. Li, Q. Li, and J. Zhong, "A Comprehensive Exploration on Spider with Fuzzy Decision Text-to-SQL Model," IEEE Trans. Ind. Informatics, vol. 16, no. 4, pp. 2542–2550, Apr. 2020, doi: 10.1109/TII.2019.2952929.

[20] Zhong, V., Xiong, C., & Socher, R. (2017). Seq2SQL: Generating Structured Queries from Natural Language using Reinforcement Learning. http:arxiv.org/abs/1709.00103

[21] Yu, T., Yasunaga, M., Yang, K., Zhang, R., Wang, D., Li, Z., & Radev, D. (2018). SyntaxSQLNet: Syntax tree networks for complex and cross-domain Text-to-SQL task. Proceedings of the 2018 conference on empirical methods in natural language processing, EMNLP 2018, 1653–1663. Retrieved from https://arxiv.org/abs/1810.05237v2

[22] Bogin, B., Gardner, M., & Berant, J. (2019). Representing Schema Structure with Graph Neural Networks for Text-to-SQL Parsing. ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, 4560–4565. https://arxiv.org/abs/1905.06241v2.

[23] Guo, J., Zhan, Z., Gao, Y., Xiao, Y., Lou, J.-G., Liu, T., & Zhang, D. (2019). Towards Complex Text-to-SQL in Cross-Domain Database with Intermediate Representation. ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, 4524–4535. https://arxiv.org/abs/1905.08205v2.

[24] U. Brunner and K. Stockinger, "Valuenet: A natural language-to-sql system that learns from database information," in 2021 IEEE 37th International Conference on Data Engineering (ICDE), 2021, pp. 2177–2182.

[25] Pal, D., Sharma, H., & Chaudhari, K. (2020). Data Agnostic RoBERTa-based Natural Language to SQL Query Generation. 2021 6th International Conference for Convergence in Technology, I2CT 2021. https://arxiv.org/abs/2010.05243v3.

[26] Ma, J., Yan, Z., Pang, S., Zhang, Y., & Shen, J. (2020). Mention Extraction and Linking for SQL Query Generation . 6936–6942. https://arxiv.org/abs/2012.10074.

[27] Mellah, Y., Ettifouri, E. H., Rhouati, A., Dahhane, W., Bouchentouf, T., & Belkasmi, M. G. (2021). Sql generation from natural language using supervised learning and recurrent neural networks. In lecture notes in networks and systems (Vol. 144, pp. 175–183). Springer. https://doi.org/10.1007/978-3-030-53970-2_17

[28] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural Computation, 9(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735.

[29] Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. EMNLP 2014 - 2014 Conference on Empirical Methods in Natural

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 in Natural Language Processing, Proceedings of the Conference, 1532–1543. https://doi.org/10.3115/V1/D14-1162.

[33] H. Vathsala and S. G. Koolagudi, "NLP2SQL Using Semi-supervised Learning," in Advanced Computing, 2021, pp. 288–299.

[34] Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference, 1, 4171–4186. https://arxiv.org/abs/1810.04805v2.

[35] Zhang, R., Yu, T., Er, H. Y., Shim, S., Xue, E., Lin, X. V., Shi, T., Xiong, C., Socher, R., & Radev, D. (2019). Editing-Based SQL Query Generation for Cross-Domain Context-Dependent Questions. EMNLP-IJCNLP 2019 - 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Proceedings of the Conference, 5338–5349. https://arxiv.org/abs/1909.00786v2

[36] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach. https://arxiv.org/abs/1907.11692v1

[37] Li, Y., Tarlow, D., Brockschmidt, M., & Zemel, R. (2015). Gated Graph Sequence Neural Networks. 4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings. https://arxiv.org/abs/1511.05493v4.

[38] Wang, C., Tatwawadi, K., Brockschmidt, M., Huang, P.-S., Mao, Y., Polozov, O., & Singh, R. (2018). Robust Text-to-SQL Generation with Execution-Guided Decoding. https://arxiv.org/abs/1807.03100v3.

[39] Yu, T., Li, Z., Zhang, Z., Zhang, R., & Radev, D. (2018). TypeSQL: Knowledge-based Type-Aware Neural Text-to-SQL Generation. NAACL HLT 2018 - 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference, 2, 588–594. https://arxiv.org/abs/1804.09769v1

[40] Xu, B., Cai, R., Zhang, Z., Yang, X., Hao, Z., Li, Z., & Liang, Z. (2019). NADAQ: Natural Language Database Querying Based on Deep Learning. IEEE Access, 7, 35012–35017. https://doi.org/10.1109/ACCESS.2019.2904720

[41] Zhang, Y., & Weiss, D. (2016). Stack-propagation: Improved representation learning for syntax. 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016 - Long Papers, 3, 1557–1566. https://doi.org/10.18653/V1/P16-1147

[42] Yu, T., Zhang, R., Yang, K., Yasunaga, M., Wang, D., Li, Z., Radev, D. (2018). Spider: A Large-Scale Human-Labeled Dataset for Complex and Cross-Domain Semantic Parsing and Text-to-SQL Task. Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018, 3911–3921. Retrieved from https://arxiv.org/abs/1809.08887v5

[43] Zelle, J., & Mooney, R. (1996). Learning to Parse Database Queries Using Inductive Logic Programming. AAAI/IAAI, 2.

[44] Hemphill, C. T., Godfrey, J. J., &; Doddington, G. R. (1990). The ATIS Spoken Language Systems Pilot Corpus. https://aclanthology.org/H90-1021

[45] Yu, T., Zhang, R., Yasunaga, M., Tan, Y. C., Lin, X. V., Li, S., Er, H., Li, I., Pang, B., Chen, T., Ji, E., Dixit, S., Proctor, D., Shim, S., Kraft, J., Zhang, V., Xiong, C., Socher, R., & Radev, D. (2019). SParC: Cross-Domain Semantic Parsing in Context. ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, 4511–4523. https://arxiv.org/abs/1906.02285v1

[46] J. S. Redman et al., "Accurate Identification of Fatty Liver Disease in Data Warehouse Utilizing Natural Language Processing," Dig. Dis. Sci., vol. 62, no. 10, pp. 2713–2718, 2017, doi: 10.1007/s10620-017-4721-9.

[47] X. Y. P. L. H. Wang Liwei; Ruan, "Comparison of Three Information Sources for Smoking Information in Electronic Health Records," Cancer Inform., vol. 15, p. CIN.S40604, 2016, doi: 10.4137/CIN.S40604.

[48] E. Soysal et al., "CLAMP - a toolkit for efficiently building customized clinical natural language processing pipelines.," J. Am. Med. Inform. Assoc., vol. 25, no. 3, pp. 331–336, Mar. 2018, doi: 10.1093/jamia/ocx132.

[49] G. K. Savova et al., "Mayo clinical Text Analysis and Knowledge Extraction System (cTAKES): architecture, component evaluation and applications.," J. Am. Med. Inform. Assoc., vol. 17, no. 5, pp. 507–513, 2010, doi: 10.1136/jamia.2009.001560.

[50] K. L. O'Halloran, G. Pal, and M. Jin, "Multimodal approach to analysing big social and news media data," Discourse, Context & Media, vol. 40, p. 100467, 2021.

[51] Z. Wei, I. Trummer, and C. Anderson, "Demonstrating Robust Voice Querying with MUVE: Optimally Visualizing Results of Phonetically Similar Queries," in Proceedings of the 2021 International Conference on Management of Data, 2021, pp. 2798–2802.

[52] I. Trummer and C. Anderson, "Optimally Summarizing Data by Small Fact Sets for Concise Answers to Voice Queries," in 2021 IEEE 37th International Conference on Data Engineering (ICDE), 2021, pp. 1715–1726.

[53] K. S. Ranti, D. Tuapattinaya, C. Chang, and A. S. Girsang, "Data warehouse for analysing music sales on a digital media store," J. Phys. Conf. Ser., vol. 1477, p. 32013, Mar. 2020, doi: 10.1088/1742-6596/1477/3/032013.

[54] P. A. Dhomne, S. R. Gajbhiye, T. S. Warambhe, and V. Bhagat, "ACCESSING DATABASE USING NLP," Int. J. Res. Eng. Technol., vol. 02, pp. 589–594, 2013.

[55] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour, "Policy Gradient Methods for Reinforcement Learning with Function Approximation," in Proceedings of the 12th International Conference on Neural Information Processing Systems, 1999, pp. 1057–1063.

[56] O. Rubin and J. Berant, "SmBoP: Semi-autoregressive Bottom-up Semantic Parsing," CoRR, vol. abs/2010.12412, 2020, [Online]. Available: https://arxiv.org/abs/2010.12412.

[57] M. N. Karthik and G. Makkar, "NLIDB Systems for Enterprise Databases: A Metadata Based Approach," in Advances in Intelligent Systems and Computing, 2021, vol. 1174, pp. 231–241, doi: 10.1007/978-981-15-5616-6_17.

[58] G. S. Reddy, R. Srinivasu, M. P. C. Rao, and S. R. Rikkula, "Data Warehousing, Data Mining, OLAP and OLTP Technologies are essential elements to support decision-making process in industries," Int. J. Comput. Sci. Eng., vol. 2, no. 9, pp. 2865–2873, 2010.

[59] G. Garani, A. Chernov, I. Savvas, and M. Butakova, "A Data Warehouse Approach for Business Intelligence," in 2019 IEEE 28th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), 2019, pp. 70–75, doi: 10.1109/WETICE.2019.00022.

[60] O. Aziz, T. Anees, and E. Mehmood, "An Efficient Data Access Approach With Queue and Stack in Optimized Hybrid Join," IEEE Access, vol. 9, pp. 41261–41274, 2021, doi: 10.1109/ACCESS.2021.3064202.

[61] G. Hains, C. Li, D. Atkinson, J. Redly, N. Wilkinson, and Y. Khmelevsky, "Code generation and parallel code execution from business UML models: A case study for an algorithmic trading system," 2015, doi: 10.1109/SAI.2015.7237130.

[62] G. Hains, C. Li, N. Wilkinson, J. Redly, and Y. Khmelevsky, "Performance analysis of the parallel code execution for an algorithmic trading system, generated from UML models by end users," in Parallel Computing Technologies (PARCOMPTECH), 2015 National Conference on, 2015, pp. 1–10, doi: 10.1109/PARCOMPTECH.2015.7084518.

[63] G. J. D. R. Hains, Y. Khmelevsky, and T. Tachon, "From natural language to graph queries," in 2019 IEEE Canadian Conference of Electrical and Computer Engineering (CCECE), 2019, pp. 1–4.

[64] Y. Khmelevsky, G. G. Hains, and C. Li, "Automatic Code Generation Within Student's Software Engineering Projects," in WCCCE '12, 2012, pp. 29–33, doi: 10.1145/2247569.2247578.