IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received January 12, 2022, accepted January 24, 2022, date of publication January 28, 2022, date of current version February 10, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3147586

A Review of NLIDB With Deep Learning: Findings, Challenges and Open Issues

SHANZA ABBAS[1], MUHAMMAD UMAIR KHAN[1], SCOTT UK-JIN LEE[2], (Member, IEEE), ASAD ABBAS[3], AND ALI KASHIF BASHIR[4], (Senior Member, IEEE)

[1]Department of Computer Science and Engineering, Hanyang University, Ansan 15588, Republic of Korea
[2]Department of Computer Science and Engineering, Major in Bio Artificial Intelligence, Hanyang University, Ansan 15588, Republic of Korea
[3]Faculty of Information Technology, University of Central Punjab, Lahore 54000, Pakistan
[4]Department of Computing and Mathematics, Manchester Metropolitan University, Manchester M15 5RN, U.K.

Corresponding author: Scott Uk-Jin Lee (scottlee@hanyang.ac.kr)

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government [Ministry of Science and ICT (MSIT)] (No.2020-0-01343, Artificial Intelligence Convergence Research Center (Hanyang University ERICA)).

ABSTRACT Relational databases are storage for a massive amount of data. Knowledge of structured query language is a prior requirement to access that data. That is not possible for all non-technical personals, leading to the need for a system that translates text to SQL query itself rather than the user. Text to SQL task is also crucial because of its economic and industrial value. Natural Language Interface to Database (NLIDB) is the system that supports the text-to-SQL task. Developing the NLIDB system is a long-standing problem. Previously they were built based on domain-specific ontologies via pipelining methods. Recently a rising variety of Deep learning ideas and techniques brought this area to the attention again. Now end to end Deep learning models is being proposed for the task. Some publicly available datasets are being used for experimentation of the contributions, making the comparison process convenient. In this paper, we review the current work, summarize the research trends, and highlight challenging issues of NLIDB with Deep learning models. We discussed the importance of datasets, prediction model approaches and open challenges. In addition, methods and techniques are also summarized, along with their influence on the overall structure and performance of NLIDB systems. This paper can help future researchers start having prior knowledge of findings and challenges in NLIDB with Deep learning approaches.

INDEX TERMS Text to SQL, natural language processing, NLIDB, database, natural language, deep learning, structured language.

I. INTRODUCTION

In today's digital world, most of the data in the world are stored in relational databases for critical applications. Data like medical records, entertainment applications data, financial transaction applications, Customer relation systems etc., are required to be accessed at all times by domain experts [1]. Domain experts are least likely to know structured query languages (SQL). Therefore, they have to hire technical help that provides them with the graphical user interface to access the required data. That data access comes with minimum flexibility and many constraints. If these databases can get accessed with natural language text queries, the impact of the data can be changed drastically [1]. The pre-requisite of structured language knowledge to access the data is a massive hurdle from utilizing it to its maximum potential. SQL has been a powerful and impactful language, but it's not easily learnable for non-technical individuals. A system that can translate Text to SQL can be a game-changer for the data science world. Natural Language Interface to Database (NLIDB) is a solution that allows users to interact with the database without any additional knowledge of formal or technical languages [20]. Figure 1 illustrates an overview of how NLIDB provides access to the database for users with natural language questions.

NLIDB is a research area at the merge of Natural Language Processing (NLP) and data Sciences [2]. Traditionally, NLIDB were built based on the handcrafted rules, grammar and integrated techniques and methods from NLP (Natural Language Processing) and data sciences, using machine

The associate editor coordinating the review of this manuscript and approving it for publication was Sudipta Roy.

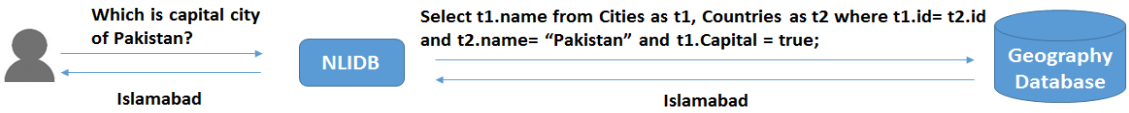


FIGURE 1. Illustration of an NLIDB.

learning merely as a supportive element. These rules are based on the semantic and syntactic considerations of the text to SQL task [1]. Before the rise of Deep learning, NLIDB research was built upon integrating NLP and data science techniques. Work from [3] is an example of such a contribution. Integrating NLP and data science techniques for the NLIDB task are known as pipeline methods NLIDB. A series of procedures and practices from data science and NLP are combined to achieve the task in the pipeline method. Despite that, this area has been of significant interest; still, its performance does not apply to practical usage. A recent boom in Deep learning made a revolutionary impact in machine translation, communication and networking. Even though communication is a mature field and got a higher bar for Deep learning to exhibit the required potential, DL (Deep Learning) methods have been proved to be the competitor of state of the art techniques [4]. Mobile traffic classification is another area where DL can improve complex problems and limitations linked with traditional methods. In traffic classifiers, Deep learning methods do not necessarily need the port information and can also differentiate the traffic coming from various applications. Deep learning with the structure of training the classifier directly from input data by feature representations may improve the Traffic classifiers [5]. The ability of Deep learning to capture complex dependencies reduces human interventions. Therefore, besides the limitations that Deep learning possesses in mobile encrypted traffic classification, it has potential for performance improvement [6].

With all the success and potential that DL has exhibited in other areas, it also brought NLIDB into the focus, making it a particular case of machine translation. Intuitively sequence to sequence model solved the text-to-SQL task with Deep learning [7]. An input question was considered a sequence of tokens and mapped to the output SQL query again. [8] trained the sequence to sequence model with NL question and SQL query paired datasets. Later it got few variations in the basic approach to solve some linked issues such as the order matter problem. A significant variation approach is the sequence to set method that have been explained further in section 3 [9]. Current work for NLIDB with Deep learning is progressing with both techniques in parallel. A review of NLIDB with Deep learning can be helpful to summarize the findings, limitations and research challenges. After having that kind of big picture, it is possible to mitigate the issues and find a combination of methods and techniques to resolve the current hurdles. The paper is organized in the following manner. Section 1 is the introduction, section 2 consist of related work, and Section 3 describes fundamental concepts of the NLIDB. The research method is explained in detail in section 4. Research questions are also formed in section 4. Section 5, 6 and 7 answer the research questions described in section 4. Section 8 concludes the review paper. The taxonomy of this paper is shown in figure 2. Acronyms used throughout the paper are listed in table 1 along with their full forms.

TABLE 1. Acronyms used in the paper.

Acronyms	Expansions
NLIDB	Natural Language Interface to Database
NL Query	Natural Language Query
SQL	Structural Query Lanuage
NLP	Natural Language Processing
DL	Deep Learning
NL	Natural Language Process
ML	Machine Learning
DB	Database
SCN	Supplement Column Names
JTF	Joint Table Filtering
GLoVe	Global Vectors For Word Representation
BERT	Bidirectional Encoder Representation
VLDB	Very Large Database
COLING	International Conference on Computational Linguistics
SPLASH	Semantic Parsing with Language Assistance from Humans
SParC	Semantic Parsing in Context
NMT	Neural Machine Translation
SMT	Statistical Machine Translation
NN	Neural Networks
MRL	Machine-Readable Language
GNN	Graphical Neural Networks
CNN	Convolutional Neural Network
GAN	Generative Adversarial Networks
OSM	OpenStreetMap
LSTM	Long Short Term Memory

II. RELATED WORK

Building Natural Language Interface to Database (NLIDB) has been a numerous challenge since the 1970s. Most of the initial work was based on a rule-based approach with manually created attributes [2], [10]. Later going through the path of querying with specific keywords, pattern-based questions and grammar-based strategies, the NLIDB system got the solution based on pipeline methods [1]. In pipeline methods, techniques and procedures from Natural Language Processing (NLP) and data science are integrated [11]. Various pipeline methods have been proposed to solve issues of NLIDB like semantic issues related to the task [12], dealing with its syntactic and semantic problems separately [13] and utilizing metadata to generate queries [14].

Other than these, few repairing architectures have also been proposed to improve the performance by detecting and correcting generated SQL queries [15]; using the "human in loop" to improve performance has also been adopted [16]. [15] and [16] are augmenting systems that

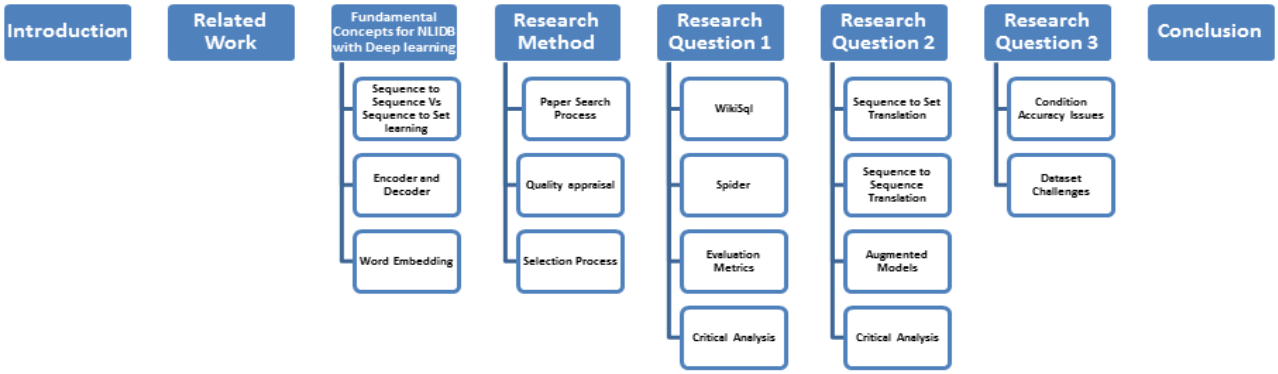


FIGURE 2. Taxonomy of paper.

expand the existing frameworks by integrating some tools pre/post the procedure. Besides all this work done in NLIDB with pipeline methods, NLIDB with machine learning is the future for this area. After recent advancements in machine learning brought the text-to-SQL task into focus again [8]. Work in this area via machine learning had been affected by the fact that no large complex dataset was available. As working with Deep learning tremendously depends on public datasets. That became one of the hurdles in training data-driven complex Deep learning solutions [17]. Recently, after few large annotated datasets have been released, NLIDB with Deep learning based on supervised training is an uprising focus [18]. These datasets contain an extensive collection of natural language questions and their corresponding SQL queries. With available datasets, supervised Deep learning models have brought new possibilities for this area of work [8]. Adapting Deep learning proved to be a potential solution for NLIDB. However, it also has various challenges linked as it is observed that despite being a simple dataset of Wikisql, still, no work has been able to achieve 100% accuracy on the task [19]. It is still vague what it will take to perform better accuracy with available datasets, i.e., Wikisql and spider datasets. Also, the question of how NLIDB can progress enough for industrial use is a primary focus. To clear up mentioned concerns, it is essential to view the findings and problems of the area in one place.

There is not much work available regarding review papers for NLIDB [20]. Available review papers are either not up to date or too generic and do not specifically address the NLIDB with Deep learning. A literature review study by [21] is one of the rare studies done for NLIDB. It left many gaps to fill, such as limited coverage of the area; that means [21] had included selective and less number of contributions for this review. That much work cannot cover the findings for the whole area. The timeline for review had not been specified as well. Therefore, the latest work representation was not the focus. They discussed the advantages and disadvantages of NLIDB systems along with the features comparison of 4 selected NLIDBs. The whole study consists of 10 papers; therefore, it is not a detailed review comprising ten articles, unlike current work consisting of more than 50 papers. A literature review by [19] covered the latest ideas and trends of the area. With a specified timeline, it covered the area more effectively. The scope of this study was broader than the current study. Therefore, feature comparison was the only aspect focused on in this work.

Survey of NLIDB by [21] covers an enormous scope and therefore could not focus on NLIDB with Deep learning enough. In the current study, technical challenges and hurdles are highlighted for researchers. With the recent work that researchers had put in NLIDB with Deep learning, it became essential to cover the scope, ideas, and challenges related to technical aspects. This review paper summarises and analyses the latest work, limiting our content only to Deep learning NLIDB. We have covered the newest contributions for Deep learning NLIDBs but results from articles are compared only for those NLIDB which have been experimented with Wikisql dataset or Spider dataset for equal grounds of comparison. The overall analysis is conducted based on accuracy, the approach adopted, and the combination of encoding techniques and methods have been discussed with their possible effects: research challenges and strategies to tackle them have also been highlighted in this work. Table 2 summarizes the similar work done previously for the NLIDB area, what features are covered, and what further has been dealt with in the current contribution.

III. FUNDAMENTAL CONCEPTS FOR NLIDB WITH DEEP LEARNING

A. SEQUENCE TO SEQUENCE AND SEQUENCE TO SET LEARNING

Initial work on NLIDB with Deep learning was based on Sequence to Sequence machine translation [8]. With Sequence to Sequence machine translation, the input sequence is mapped with the output sequence. These models are trained based on input sequence and given output sequence mappings. This approach faced the "Order matter" problem, which impacted its performance critically. To overcome this issue, sequence to sequence methods use reinforcement learning [17], but it also has not proved to mitigate this

TABLE 2. Comparison of previous review work with current work.

Question	Current Study	E Uand P C 2017 [21]	S Dar,I Lali et al. 2019 [20]	Affolter,Stockinger et al. 2019 [22]
Represent the latest work in the area	Yes	No	Yes	Yes
Current issues and challenges are highlighted	Yes	No	No	Yes
Focus of study	NLIDB with deep learning	NLIDB	NLIDB	NLIDB
Scope	Text to sql	Text to sql	Querying any databases (Sql/NoSql)	Text to sql
Highlighted the collaboration impact of NLP and Deep Learning for NLIDB?	Yes	No	No	No

issue altogether. Sequence to set learning is an alternative approach to avoid the ''order matter'' problem by dividing the prediction process into sections [9]. A separate module predicts a specific part of the query, and its dependency is selective based on attention mechanism instead of the whole sequence. Sequence to set approach has other limitations like lack of context and global dependency between database schema and Natural Language (NL) questions. Attention mechanisms are a possible solution for this problem. Currently, work effort in the NLIDB area is parallel based on both of these machine translation systems.

B. ENCODER AND DECODER

With the Deep learning NLIDB, a concept of encoder and decoder has been proved a practical approach. After the initial adoption of the encoder-decoder approach for machine translation by [7], this has been the most popular approach for text to SQL translation, usually based on LSTM for both encoder and decoder in most cases. Encoder reads the input sequence one at a time and converts the whole sequence into vector representation that can be used to predict the output. The hidden state of the encoder is passed to the decoder for processing and decoding the sequence of predicted results. Various methods and techniques are used for the encoder to create the most effective and impactful input representation [23]. Encapsulating the maximum and most relevant aspects of the input data is an integral part of the process. Similarly, extracting the output based on the hidden states of the encoder is the other half of the process performed by the decoder [24].

C. WORD EMBEDDING

Word embedding converts the textual information into numerical representation to make it interpretable for the Deep learning models [25]. Various word embedding techniques are crafted within proposed models, but some off-the-shelf word embedding models are widely used for encoding purposes [26]. GloVe (Global Vectors for Word Representation) model is an example of such techniques used for word embedding to capture the relation and meaning of words in linear directions. It statistically finds the connection of the words by identifying and counting their co-occurrences [27]. BERT (Bidirectional Encoder Representations) [28] is another example of off the shelf encoders. Using BERT, only the decoder layer is needed to be designed for a prediction model. It has been fine-tuned and utilized in many state of the art NLIDB systems. Table 3 consists of the list of fundamental concepts for NSIDB with deep learning that can help understand the background more effectively. Figure 3 exhibits the general idea of encoder-decoder based NLIDB with Deep learning models.

IV. RESEARCH METHOD

This review aims to provide a detailed understanding of NLIDB concepts, findings, and limitations regarding further improvement. To define a more clear scope of this paper, it is essential to specify the research questions. This paper evaluates the recent work done in the NLIDB area by comparing the recently proposed text to SQL models, their performance, and each model's limitations. This article also emphasizes the importance of NLP methods and their role in the development of NLIDB. NLP techniques and practices that are being utilized recently to cover the performance gaps and bringing further improvement regarding the accuracy are also discussed, along with the issues they impact most. Brief comparisons and detailed discussions in this work are beneficial to give a head start about achieved milestones vs gaps in the area. Following are the research questions to describe these objectives briefly.

RQ1: What are available datasets for text to SQL tasks with supervised learning, and how are they essential to improve performance?

RQ2: What are the approaches that have been adapted from Deep learning for NLIDB until now? And what are their research focuses?

RQ3: What are the focused research challenges of NLIDB with machine learning, and NLP methods and techniques being used to mitigate them?

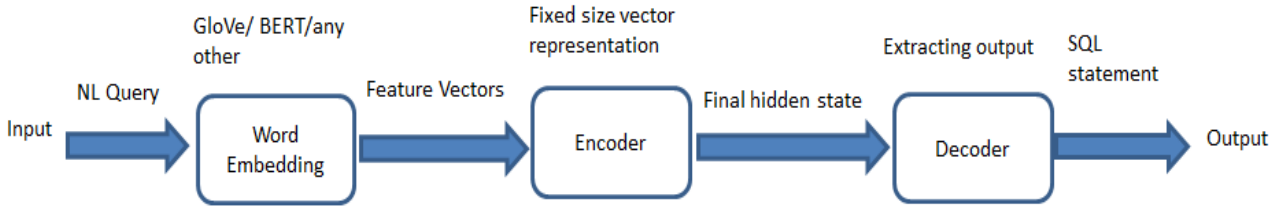


FIGURE 3. NLIDB with deep learning.

TABLE 3. Summary of fundamental concepts for NLIDB with deep learning.

Keywords	Description
Sequence to Sequence learning	Input sequence is mapped with the output sequence wise.
Sequence to Set learning	Output is break down into sets based on clauses or relevance.
Encoder	Encodes input sequence into hidden states.
Decoder	Predict output based on encoded hidden states.
Word Embedding	Converts textual tokens into numerical representations.

FIGURE 4. Research methodology.

After defining the research questions, it is essential to specify the methodology process for our review study. Search and identify the available work relevant to the NLIDB is the first step of the methodology process. Figure 4 illustrates the two steps based methodology process adopted in this study.

A. PAPER SEARCH PROCESS

The search process consists of three phases to cover the most recent work in the area. In the first phase, we searched through three electronic databases "Google Scholar", "IEEE Xplore" and "Scopus". The used search keywords are "NLIDB", "NLIDB and machine learning", "text to SQL ", "Natural language to structured language", and ("NLIDB" and "Machine Learning"). In the second phase, the high-ranked conferences in Machine Learning, data Science, and Natural Language Processing were enlisted based on the BK list, Core and SOC list. The top 3 conferences related to each area (DB, NLP, and ML) were selected and scanned manually. After reading the titles of all the papers from the conferences chosen, We picked seemingly relevant articles.

In the third phase, We manually scanned the bibliography of the selected papers as well. After searching through reference lists of the selected papers, We gathered relevant papers from the bibliography of the previously chosen papers.

Table 4 shows the scanned conferences for relevant articles and the number of papers gathered from each conference. It is observed that most of the NLIDB related papers are found in NLP conferences. But it cannot be taken as a thumb rule, as we have also seen many valuable articles from VLDB (Very Large Databases) conference proceedings which are DB conferences.

On the other hand, machine learning or pattern recognition conferences seem to have no such exception. As it can be observed from table 4, We found most of the relevant papers in NLP conferences, and dB conferences are 2nd on the list. Near to no paper were discovered in Machine Learning conferences. Based on these observations, we can say that this topic is most closely related to the NLP and using Deep learning as a tool is the only latest trend for the issue. Table 4 shows the manually searched and scanned conferences for the latest work in the area of NLIDB.

B. QUALITY APPRAISAL

Our defined criteria for quality appraisal and filtering papers consist of the following rules.

Articles that are case reports are not included.

- Review papers are categorized separately to consider for related work but not included for comparative analysis.

TABLE 4. Manually scanned conferences.

No.	Conference Name	# of Papers
	Natural Language Processing (NLP) Conferences	
1	Empirical Methods in Natural Language Processing	12
2	Annual Meeting of the Association for Computational Linguistics	5
3	International Conference on Computational Linguistics (COLING)	3
	Databases Conferences	
4	ACM SIGKDD Conference on Knowledge Discovery and Data Mining	0
5	VLDB: Very Large Data Bases	7
6	ACM SIGIR Conference on Information Retrieval	0
	Machine Learning (ML) Conferences	
7	Conference on Computer Vision and Pattern Recognition	0
8	International Conference on Machine Learning	0
9	Conference on Neural Information Processing Systems	0

- Papers with no experiments or experiments with local datasets are not included. For comparison analysis, articles based on only the Spider dataset or Wikisql Dataset are included.
- As work in this area is still struggling for practical enough accuracy. Therefore, articles that have not mentioned the accuracy of results are not included for comparison.
- Only closely relevant to the proposing new NLIDB articles are included; NLIDBs dealing with SQL are included.
- For analysis purposes, only NLIDB with machine learning models is included.
- PhD or Masters Studies are not included.
- Conference papers are only included from the previous conferences.

C. SELECTION PROCESS

Papers collected from the search process are further studied and evaluated. The first filter is made based on their publication venue to keep up the quality from the beginning. Among 349 articles, We selected 105 articles. Later on, review papers, case reports, and distantly relevant papers were filtered out, leaving 85 papers on the list. We applied a further filter for NLIDB with machine learning and categorized filtered articles according to their direct or indirect link with the Deep learning NLIDB. Strictly related to "text-to-SQL" tasks with Deep learning was selected for comparison purposes, and NLIDB with heuristic approaches were listed to study and timeline reference of this problem. This filter left 60 articles experimented with well-known standard datasets. Papers selected for the comparison were categorized according to the dataset they were trained and tested with. It is essential to compare the articles that experimented on the same dataset for direct comparison and fair analysis. Therefore, NLIDB models that have used other than Wikisql or Spider dataset for training and testing purposes were filtered out only for reference and discussion. Finally, there were 30 papers for the comparative analysis process. Table 5 shows the scrutinized articles after each step of the process.

TABLE 5. Number of articles after quality criteria applications.

Sr.	Quality criteria application step	No. of Papers
1	Search results with year filter	349
2	Quality filter	85
3	Closely relevant papers	60
4	Articles with Wikisql or Spider Dataset	30

V. RQ1: WHAT ARE AVAILABLE DATASETS FOR TEXT TO SQL TASKS WITH SUPERVISED LEARNING, AND HOW ARE THEY ESSENTIAL TO IMPROVE PERFORMANCE?

For supervised Deep learning, text to SQL task needs labelled dataset. There are few datasets available for this purpose with the most straightforward queries. Those simple queries are not complex enough to train a model for practical usage. The most well-known datasets are ATIS, GeoQuery, Restaurants, Scholar, Academic, Yelp and IMDB, Wikisql, and Spider. Most of them are related to one database or consist of single table databases [29]. The recently constructed dataset SPLASH (Semantic Parsing with Language Assistance from Humans) [30] offers complicated databases and queries to experiment further. This dataset is mainly focused on correcting the SQL queries based on human response. This dataset is not much tested and worked with yet. Another dataset by [31] is another recently constructed dataset for the text to SQL task. This dataset mainly consists of medical records which contain additional difficulty based on abbreviations and technical terms. Besides that, these recent datasets are comparatively more complex and appropriate to represent real-time issues. But they have not been used and experimented with yet. Among all these datasets, Wikisql and Spider are the most widely used datasets [32] as they have strong baseline models and a standard evaluation matrix. Therefore, they are more convenient to experiment and compare results.

Moreover, they have the most enriched and complex datasets among other available datasets [8]. Restaurants dataset consists of user questions about food and location etc. and not a labelled dataset with SQL queries. The scholar dataset is about academic publications and their corresponding SQL verified from users. The academic dataset has a similar domain with scholars, but their schema is different. Geoquery contains questions about US geography with SQL

TABLE 6. Summary of recent dataset for NLIDB.

Dataset	#DB	Cross Domain	Joins	Order By	Group By	Nested	Having
ATIS	1	No	Yes	No	Yes	Yes	No
GeoQuery	1	No	Yes	Yes	Yes	Yes	Yes
WikiSQL	26521	No	No	No	No	No	No
Spider	200	Yes	Yes	Yes	Yes	Yes	No
SParC	200	Yes	Yes	Yes	Yes	Yes	Yes

annotation [29]. The advising dataset includes the questions about the University of Michigan and Yelp, and IMDB has about the yelp website and online movie database. ATIS corpus was designed for speech query systems for relational databases. This dataset contained questions about flight booking and consisted of a single database [33]. Therefore, it did not offer much logical complication as compared to the recently presented datasets. However, [34] introduced the expanded version of the ATIS dataset, including the context dependency within the questions. Besides, the context-dependency ATIS dataset offers limited logical complexity as compared to the SParC dataset. It is a multi-turn version of the Spider dataset [35]. Semantic Parsing in context (SParC) dataset is the context dependant variation of the Spider dataset. This dataset is used mainly for interactive systems where series of queries are interconnected via context dependency. It has more logical complications because of the cross-domain and multi tables based queries.

In this study, our comparison base is Wikisql and Spider datasets as they have a vast range of implemented models, and many state of the art models are training with either one of these two or both of these datasets.

A. WIKISQL

An extensive collection of automatically generated questions about individual tables from Wikipedia, paraphrased by crowd workers to be fluent English [8]. It is an extensive collection of hand-annotated data. As Wikisql contains a large variety of databases, therefore it offers query diversity for the training process. This dataset does not have any joins as each database consists of one table only [29]. Therefore, NLIDB models trained with this dataset do not cover joins. They only cover the select and where clause. Evaluation Metrix for this dataset includes execution accuracy, query match accuracy and logical form accuracy.

B. SPIDER

A most recent large-scale, human-annotated and cross-domain Text-to-SQL benchmark. Dataset is categorized in Easy, Medium, Hard and Extra Hard levels [32]. Queries with more than two SELECT columns, more than two WHERE conditions, and GROUP BY two columns, or contains EXCEPT or nested queries are considered hard. Anything above that is extra hard. Evaluation matrix includes Execution accuracy and query match accuracy [29]. The Spider dataset has multiple tables therefore contains joins also. Furthermore, it has queries with other clauses to make a more complex and real time dataset. It is observed that the Spider dataset has the closest similarity to the actual queries. It is the latest trend for training and experimenting purpose in the recent NLIDB area. Current datasets that have been used for NLIDB are listed along with their features in table 6.

C. ACCURACY MATRIX

WikiSQL was launched with logical form accuracy evaluation metrics and execution accuracy. Logical form accuracy meaning if the predicted query matches its gold query in terms of logic, but the logically correct queries can be executed error-free giving unintended results [8]; this is why execution accuracy gives vague results in this text-to-SQL tasks. Although Execution accuracy requires more minor details to be taken care of, it also provides false positives. Similarly, the Spider ladder board has two types of accuracy metrics. One of them is exact matching accuracy, and the other is execution accuracy [32]. When the output query matches the components of the gold query, it is exact match accuracy. Exact match accuracy ignores the order of select columns but does not evaluate the where clause values. Execution accuracy is when the executed result of output and gold query matches [36].

D. CRITICAL ANALYSIS

As can be seen from table 6, the most complicated dataset available for this task is the Spider dataset to this date. ATIS data has joins and data in multiple tables, but its data is from a single domain and has no order by clause. Similarly, the Geoquery dataset has some complicated queries but do not have cross-domain data. Wikisql is a more extensive dataset than both previously discussed datasets and, therefore, much more used than the other two. But Wikisql contains most simple queries, and models trained with such datasets cannot cope with complications of real-time databases. It can be observed that previous datasets either consist of a single domain or a single table. Therefore, not contently enough to be trained for real-time databases. Spider dataset, being most recent, has solved some of those problems but not all. It has a cross-domain dataset as well as joins and complicated queries also. But the number of queries containing order by and group by clauses is not enough to train a practically applicable model.

VI. RQ2: WHAT ARE THE APPROACHES THAT HAVE BEEN ADAPTED FROM DEEP LEARNING FOR NLIDB UNTIL NOW? AND WHAT IS THEIR RESEARCH FOCUS?

There have been two types of primarily used approaches for NLIDB recently. Sequence to sequence approach and sequence to set/modularized models set method [37].

Sequence to sequence approaches takes a series of input and provides a sequence output. Existing datasets for text to SQL do not have a complete query set as ground truth. Therefore, Sequence to Sequence suffers from the Order Matter problem regarding where clause [38].

On the other hand, the sequence to set approach deals with previously predicted tokens as a form of set, and dependency is selective based on different methods. Usually, they have a separate module for total columns in the query and then a particular module for predicting database entities for each part of the query. [36] resolves the order matter problem but creates another issue of not utilizing relation of columns and conditions in where clause.

A. SEQUENCE TO SEQUENCE TRANSLATION

Work by [7] is one of the pioneers who used Deep Neural Nets to perform ''End to End'' Translation through Seq2Seq Learning. They demonstrated that LSTM could be used with minimum assumptions, proposing dual LSTM Encoder-''Decoder'' architecture to do Language Translation, showing the promise of Neural Machine Translation (NMT) over Statistical Machine Translation (SMT) with a limited vocabulary. A modified model for SQL semantic parsing Seq2SQL by [8] outperformed previous baselines and showed the state of the art performance. It also minimized the issue of the query's unordered nature by using reinforcement learning. Later work in Sequence to Sequence-based approaches adopted sequence to set the concept of modularization. Work by [39] is Sequence to Sequence based encoder-decoder model with CNN usage for input embedding, distributed in 3 sub-modules, select column, and aggregate and where condition. A model with three decoding channels [40] is used for SQL keyword prediction, column name prediction, and cell value prediction.

The switching gate model is trained for switching between channels. Another channel controlling approach by [41] attention and copying mechanism is adopted along with a training approach. A type system is introduced to control the decoder. Based on the type, a type decoder gets selected to copy the constants from NL question or table headers as column name/cell value or pick up the words from a fixed vocabulary set of SQL operators. Some models attempted to solve the semantic gap with the help of external knowledge. In this context, [17] proposed a sequence to sequence pre-processing focused model. Input to its encoder is an input question, an annotated form of a question that was processed based on database schema information such as column names and values of columns, along with a set of possible phrases for a column name. After detecting the mention of columns and cell values in the question, they are replaced with dummy terms and turn NL question into annotated question form. Non-column/cell values are replaced with SQL keywords and generate annotated SQL queries. Input to the encoder is annotated question and table header/column names. A trained model can be adapted for the cross-domain with anonymized questions. Another cross-domain model proposed by [42] has a similar concept of stripping the query structure out of the question by tagging schema elements. After tagging those elements, various queries become identical and can be generalized over domains. Parser of an end to end Natural language interface to the database by [43] is based on the sequence to sequence approach. It mainly emphasizes on query correctness module. During the decoding process, a generated SQL query is tested against the query execution module, and if it is not executable, it is presented to the user for correctness. The ability to handle the feedback in the text suggests that it has an impressive GUI to support the whole process with a separate communication section, the results section and the database schema display section on the screen. The work's primary focus is query correction with user interaction and Metadata inclusion as part of the feature input to the encoder. Sequence to action parsing [44] combines Sequence to Sequence and Sequence to set approaches. It has grammar-based rules with a sequence of parsing decisions dynamically. Every step is based on previous path history, the current input and defined policy grammar, advancing according to the learnt policy. That can be different every time it is executed, and it is not fixed. That's why it's called the non-deterministic incremental approach.

The idea of multiple correct output/paths is proposed. For more semantic assistance with cross-domain dataset spider, the editing mechanism by [45] is vital for query generation with the help of context vector, i.e., last output query. The current query checks the probability with a context vector that can copy any component from the previous query in the current one. Input for its decoder is NL utterance along with most relevant column names, context vector and database schema. More work has been done to add context as part of input in any form. Such as [46], RAT SQL is based on sequence to sequence work, but they expanded it with a tree-like structure. Mainly they focused on including schema information with the added context of the natural language question. A joint hidden stated encoding is proposed to add more context and more schema information according to the natural language question context. The schema linking approach for tagging the natural language question with schema related info is integrated with the GNN graph of schema. Another schema encoding based work by [47] incorporated the schema information with a separate schema encoder for this purpose, along with a sequence encoder. It enhanced the contextual information for the decoder.

Although most of the work has experimented with Wikisql or Spider dataset, not all have worked with Wikisql or spider dataset. Therefore, they are not part of our comparisons. But their pioneer work settled the base for further work in this area. Such as NEURAL ENQUIRER [48] is one of the pioneers of NLIDB with end to end Neural Network (NN). It is entirely based on end to end training of neural networks with input-output training examples. It has limited implementation as the experiment was based on one table dataset. Natural language interface specifically for OpenStreetMaps database [49] recognizes location keywords from

NL question and preprocessing of the dataset. Recognized area mapped to the OSM object by using OSM tool nomination and search technique with string matching. Then a base semantic parser is used to convert NL to Machine Readable Language. The base parser, in this case, is an SMT system that translates from natural language to a machine-readable language (MRL).

B. SEQUENCE TO SET TRANSLATION

A more simplified sequence to set model was introduced by [9] and resolved the ''order-matters'' problems from Seq2SQL without using Reinforcement learning. It further proposed a column attention structure for adding column context information. A slot filling approach with type recognition was proposed by [50] based on SQLNet. From the NL question, the types of each token are recognized and paired with the tokens. This type, word pairs are also part of the input feature for the encoder-decoder NLIDB model. The idea of decoupling of SQL syntax problems from schema issues adopting slot filling approach with dual encoder model was proposed [18]. The model also adds contextual information with a dual attention mechanism.

The first model generates the SQL syntax sketch, which is encoded as input and the natural language query. The second model is for SQL generation, which takes the encoded sketch and NL question as input. SQL sketch generation model was based on the sequence to sequence approach. It also utilized the attention mechanism for SQL sketch and database columns to add full context from both sides. Coarse2fine [37] is another attempt at separating semantic and structure issues. The central concept is to generate a rough but meaningful sketch first as an intermediate form. That sketch is later transformed into a structured query based on natural language questions, database schema and sketch itself.

Another such model is proposed by [51], where they separate schema related information in NL query by entity linking. This method enhanced the performance in the domain and made the model trainable for the cross-domain. A data augmentation algorithm was also proposed to reduce the human effort for preparing training data at the target domain. One of the current research focuses on cross-domain models is to get equivalent or better performance via models with fewer data requirements. Thus, the proposed structure by [52] is among such contributions. It mainly deals with turning a regular supervised learning task into a Meta-learning task for semantics. It presented a design to create a pseudo task with the help of the relevance function. Hence, a new system can be trained quickly and with a small dataset. Also, it makes learning more specific to each example, increasing the semantic and syntactic mapping. The semantic gap mainly comes from misinterpreted column names and wrong cell-column values. Another slot filling approach proposed by [53] combined the rule-based method by turning the text to SQL task into week supervised learning. Framework worked in two parts; first creating the SQL with the help of database grammar rules and then using a neural network based on explored SQL queries. An algorithm is proposed based on standard database rules as grammar rules for the first part of the process.

In the second part, three main modules are trained: the select column, the select aggregate, and the where clause. It takes natural language question and table header hidden state as input—similar architecture proposed by [54] with distinction where they focused on encoding the input with BERT. Work on grammar-based slot filling approach by [55] with neural network showed the minimum over a generation for the task. Cell value information to cover the wrong condition value problem was focused on in work by [19]. Values are identified from the NL question, and then value-column pairs are generated. Value context is also calculated based on value and input questions. Value abstraction is done by replacing values in question with the constant token ''ENTITY''. All these preprocessed sets are added to the encoder as input features.

In the context of a more complex SQL task spider dataset, SyntaxSQLNet [56] network is a state of the art framework. It works based on a predefined SQL structure, and its module predicts in the sequential pattern based on history token. It takes NL question, DB schema, current SQL decoding history path and manually created grammar as input. The current SQL decoding history path adds the syntax in the upcoming prediction. The decoder structure is based on nine separately trained independent modules called for execution based on manually featured SQL grammar. SQL recursive clause wise decoding proposed by [24] predicts the SQL sketch as the first step instead of taking SQL sketch as part of input like SyntaxSQLNet. It is different from SyntaxSQLNet, as this framework works in a procedural way instead of in a sequential format. Three modules are trained for each clause, such as 1- Prediction of sketch, 2- Column prediction, 3- Operator prediction module. Also, sequence to sequence architecture is applied for column prediction instead of sequence to set. As the order is essential for the group by and order by clauses, Recursive clause wise work is further done by [57]; they emphasize the complex queries in terms of subqueries.

A recursive SQL template builder is proposed that splits the complex query into multiple subqueries and fill the slot accordingly. Statement Position Code idea is explicitly presented to deal with nested queries with a sketch based slot filling approach. IRNet is another work on spider task by [58] that tackles two main issues of NLIDB. Mismatch problem and lexical problem. This model sequentially generates an intermediate representation of NL question and schema. Tree-like structured intermediate representation does not contain exact SQL syntax but contains schema information regarding NL input. It takes NL question, database schema and entity linking information as input. IRNet was expanded [36] by integrating necessary preprocessing as part of the whole procedure. Their focus was to minimize the semantic gap between values of a natural language question and database schema. Values are not always explicitly mentioned in the question statement. Therefore, the model suffers the vagueness of their detection. They used the techniques of Question

hint and schema hint for this purpose. For each token in the question statement, it was tagged as if it is a column, table or value. Vice versa was done for the schema elements by finding a set of significant columns and tables from the question. This information is passed to the neural network as additional knowledge. Another work by [23] emphasized lexical problems and proposed an entity linking supervised learning model. An anonymization model is integrated with some base parser, and an anonymous utterance set acts as an additional input vector for the parser. Creating a dataset for anonymization model training is a significant limitation of this model. Other than semantic issues, some work focused on encoding the schema elements as effectively as possible. Work by [59] proposed to encode the relationship of all the schema elements to impact the context and from clause.

As spider dataset, queries are executed against the unseen complex dataset. Therefore, schema modelling also got some attention for NLIDB. A model proposed by [60] emphasized schema modelling to handle unseen schema issues regarding spider datasets. For this model, the database schema is encoded in its complete representation. To capture all the relations and links in the database schema, GNN is learnt to represent schema graphs with nodes and their relevance score, which also covers foreign key and primary key relations with the help of defined nodes and edges. This work was extended further by [61] adding a global node to the schema graph, hence considering schema context globally instead of relying on local relevance or similarity functions. Moreover, they also proposed a re-ranking model integrating with the prediction model. The Re-ranking model captures the candidate queries from the parser's beam and calculates their score with the re-rank loss function and learnt parameters. Re-ranking queries based on the global alignment of question words with the database constants improves the accuracy. A two-phased one-shot learning model is proposed by [62]. It consists of two models, for SQL template classification and the other is for slot filling. The basic idea is to group similar template-based queries, and each group can be predicted by a model trained with only one similar template. The model has experimented with four datasets Advising, Atis, GeoQuery and Scholar. An Encoder decoder based model SQLLova is presented by [63], integrating and collecting approaches from all over the area to get the highest accuracy score. [63] Proposed a model reusing BERT, SQLNET, and execution guidance methods mainly. BERT is used for encoding, and the decoder is based on SQLNet, with the essential variations. In contrast to SQLNet, SqlLova doesn't share parameters inter modules. Another variation is, for where-values, it depends on where selected columns and operators instead of relying on sequence network.

C. INTEGRATED MODELS

Some efforts are made to integrate a model pre/post the prediction based language model. Primarily they are trained to identify the errors and correct them. A mechanism of execution guidance was introduced by [64] to intervene the base model at a stage where they have candidate predictions. This model, integrated with any autoregressive base program, executes the partially generated query and detects and excludes faulty programs during the decoding procedure. User feedback is another popular approach to verify and improve the output. A Structured query generation framework DialSQL [65], has been proposed to boost the performance of existing algorithms via user interaction. After identifying potential errors in the query, user feedback was collected to validate them via simple multiple questions. These models experimented with Wikisql hence can work only with the most straightforward queries. For a more complex dataset spider, a user interaction model-based intelligent agent is proposed [38] to integrate with some base semantic parser to validate the output and boost accuracy results. MISP-SQL mainly consists of Agent State, Error Detector, Actuator and world model. It captures partial SQL query from the base parser, detects the error probability, and generates a question for the user. Error probability is based on comparing the base model's uncertainty score with a threshold. An approach proposed by [66] proposed a method to utilize the BERT language model. They only used BERT encoding to emphasize its complete usage and did not use any additional encoder of their own.

They integrated the output column from the language model with the questin instead of a set of candidate columns. They used execution guidance to run the query during decoding and correct it in case of run time error. Another BERT based model was proposed by [25], focusing on the correctness of the query. Based on the search beam, a re-ranker was integrated with the GNN parser, which takes the candidate queries and re-ranks them to the correct query. The re-ranking process improves the performance when the valid query is always in the candidate list but not selected as the final query. They fine-tuned the BERT language model by integrating it with the GNN parser. For the mismatch problem, integration with the base parser is proposed by [67]. They leveraged an external knowledge set of Adjective-noun pairs for operator prediction. Adjective-noun pairs are extracted from web corpora; semantic analysis is applied in 3 steps, extraction of adjectives and nouns, finding their relation, making 2 clusters of a positive and negative relation between adjective and noun. This external knowledge is passed as one of the features to any existing models; in this case, the SyntaxSqlNet model has been tested. This model mainly focuses on predicting comparison operators for columns based on adjective words in NL questions. Another significant contribution is [68], but We cannot compare it with the majority of the work done for NLIDB with Deep learning. As they have used unsupervised learning for the task, most of the work in this area is with supervised learning. They used policy gradient-based reinforcement learning with three types of coverage rewards to guide the learning process. They have used unsupervised learning with a dataset consisting of pairs of questions and answers instead of labelled datasets. Because of different datasets than other contributions, direct comparison of this

work is not possible. Tables 7 and 8 compare significant work done with the WikiSQL dataset and spider dataset, respectively.

D. CRITICAL ANALYSIS

Overall recent work in NLIDB with the Deep learning area can be categorized into 3 major categories. Sequence to Sequence work where model follows the pattern of mapping sequence of input tokens to the sequence of output tokens. This is not an ideal text pattern for SQL tasks because the order does not matter in the whole SQL query. In a sequence of tokens, order matters, and that restricts the training process. In SQL query, order only matters in the group by and order by clauses and creates overall syntax of the query. It does not matter in the where clause or in the select clause. Therefore, recent datasets like Wikisql and Spider have clause-wise evaluation schemes for this task. Sequence to sequence approach does not fit in this scenario and performs relatively poorly in condition accuracy, which ultimately raises the order matter problem.

The second category in which we have categorized the work is the sequence to set. This is another most commonly adopted approach for text to SQL tasks with Deep learning. This has been proposed and used as an alternative of sequence to sequence to avoid order matters problem. In this approach, all the clauses are made separate and treated as an individual set. A particular module is trained for each clause, and one module predicts the syntax of placing all the clauses together. This way, all clauses are predicted independently, and the prediction model can avoid the order matters problem. It invites other issues, though, like not considering enough context information for a particular prediction. Various methods and models have been proposed with Sequence to Sequence and Sequence to set approaches to resolve these issues. Still, no one is accurate enough to make it practically useable until now.

In the third category, a different kind of work has been placed than both previously discussed. Integrated models focus on the pre/post-processing part of the whole process and provide better results. In such work, they trained an extra model to process the input/output of some existing models and integrated them with the existing ones. This sort of work has also contributed significantly to the current accuracy of the NLIDB systems. Therefore, We cannot complete a brief review picture without including the review of integrated models. Most of this work is focused on methods to input the database information and techniques to gap the semantic issues by machine-human interaction.

VII. RQ3: WHAT ARE THE FOCUSED RESEARCH CHALLENGES OF NLIDB WITH MACHINE LEARNING, AND WHAT ARE NLP METHODS AND TECHNIQUES BEING USED TO MITIGATE THEM?

Major challenges for NLIDB with deep learning are distributed among 2 categories of dataset challenges and condition accuracy challenges. Figure 5 shows the subcategories

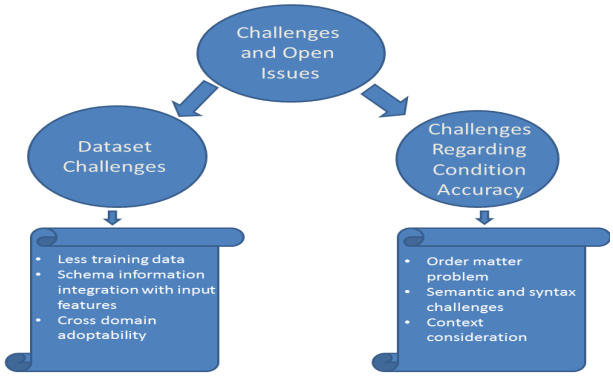


FIGURE 5. Challenges and open issues.

and they have been explained in details in the subsections. Table 10 shows the challenges faced in NLIDB area along with the solutions that have been proposed to mitigate those problems specifically.

A. CONDITION ACCURACY ISSUES

The clause ''where'' of the query holds the condition/s for the data that needs to be fetched. This part is critical in terms of query execution results. It is observed that accuracy regarding conditions in the ''where'' clause is lowest compared to the other clauses in a simple query [37]. In the NLIDB area, this problem is known as the condition accuracy issue. This review has observed that most of the challenges and problems are linked with the ''where'' clause of the query. Although most of the work in this area focuses on improving condition accuracy, it has been the most challenging part of the task [65]. As few articles have provided ablation studies and detailed results, this can provide us with more insight into the current situation of condition accuracy for NLIDBs. It can be observed from Table 9 that where clause has the least accuracy among all other clauses. Many sub-challenges are contributing to the condition accuracy overall. In this study, we will discuss those challenges in detail along with the NLP techniques used as support to cope with these issues.

1) ORDER MATTER PROBLEM

Order the conditions in ''where clause'' does not matter during execution, but it matters syntax wise while generating the query. Two queries with different conditions are similar for execution but are considered unique queries based on their different syntax [11]. There might be two queries with the same execution in training data, but because of varying condition order, they seem to be two separate queries syntactically. Finding the correct ordering for the sequence to sequence models becomes difficult, as they need single ground truth query labelling for training purposes [42]. The ''order matter'' problem has been faced by sequence to sequence-based models because they depend on the whole sequence of the tokens taking into account their order. It Creates many false negatives based on the syntax of the outputs

TABLE 7. Summary of NLIDBs with WikisQL dataset.

Reference	Execution Accuracy	Query Match Accuracy	Research Focus	Advantages	Disadvantages
Zhong,Xiong et al. 2017 [8]	59.4%	-	Reinforcement learning	State of the art accuracy among pioneer studies	Limited accuracy improvement
Xu,Liu et al. 2018 [9]	68.0%	61.3%	Solution to Order Matter problem	Where clause improvement	Context information decrease
Hosu, Iacob et al. 2018 [18]	-	38.99%	Rule based, machine learning sketch generation	syntax improvement	No attention between sql sketch and textual question
Huang, Wang et al. 2018 [55]	68.0%		Meta Learning Task	Specific task coverage	More relevance functions required for significant improvement
Gur, Yavuz et al. 2018 [19]	-	84%	Human in loop	Improved query correction	User involvement required
Wang, Tatwawadi et al. 2018 [64]	83.8%	75.4%	Guided execution of generated queries	Drop of execution errors	Gap between drop of execution errors and raise of execution accuracy
Yu, Li et al. 2018 [50]	82.6%	75.4%	Capturing relationship between attributes	Improved encoding of rare entity types and numbers	Accuracy still not realistic
Yavuz, Gur et al. 2018 [19]	-	72.4%	Condition value error	Where clause accuracy improvement	Database content usage
Shi, Tatwawadi et al. 2018 [44]	87.1%	-	Non deterministic Sequence to action grammar based architecture	Contextual information of whole sequence	Accuracy not for practical usage
Sun, Tang et al. 2018 [40]	75.1%	61.7%	Mismatch Problem	Where clause improved	Column prediction errors
Wang, Tian et al. 2018 [17]	82.2%	72.1%	Separating schema and data via annotation mechanism	Transferable model for multiple databases	Only works for SQL sketch compatible datasets
Dong and Lapata 2018 [37]	79.6%	73%	Semantic and structure separation	Improved syntax	Less Accuracy results
Guo and Gao 2019 [39]	69.0%	62.5%	CNNs for bidirectional attention	Column-select improvement	Words out of GloVe embedding not supported
Wang 2019 [42]	82.7%	74.5%	Detaching domain elements	Cross domain generalization	Accuracy not improved largely
Hwang, Yim et al. 2019 [63]	90.2%	84.2%	Integrated previous approaches forming a novel architecture	Accuracy boost	High error rate for vague columns names
He, Mao et al. 2019 [47]	86.2%	**92.3%**	Increased contextual information	Where clause improved	Not applicable for complex dataset yet
Guo and Gao 2019 [53]	81.2%	68.7%	Weak supervise learning	Logical form training skipped	Not work well with multiple conditions where clause
Wang, Shin et al. 2019 [46]	79.5%	-	Integrated schema linking with GNN schema graph	Additional syntax information in schema graph	Where clause error rate is high
Guo and Gao 2019 [54]	91.1%	85.4%	Involving table content for improved accuracy	Semantic gap minimized	Not suitable incase table content is confidential
Lyu, Chakrabarti et al. 2020 [66]	**92.2%**	86.5%	Utilizing the base language model BERT for increasing accuracy	Additional encoding not required	HydraNet do not support full SQL grammar

and drastically drops the training process's performance [9]. This issue has been proved to be one of the significant hurdles for boosting Sequence to Sequence NLIDBs. Reinforcement learning is one of the methods to resolve the challenge, using it on top of the standard supervised training procedure.

Value-based loss functions are run on a standard sequence to sequence model, based on its output to fine-tune the training purpose. They compute the reward after decoding the output, based on whether it is a well-formed query or not. That reward is used to fine-tune the algorithm by trying to maximize the total reward. The system learns the correct answer after many trials and errors with the help of rewards and penalties [9]. Although usage of Reinforcement learning improves the results, improvement is still limited and progressing slowly. Another way to avoid the order matter problem is to adopt an altogether different model structure like the sequence to set approach. This approach has been adopted widely to resolve this issue but has brought other limitations along with it. As a sequence to set process, it does not consider any previous output while predicting a token, and each model predicts a separate part of the sequence.

TABLE 8. Summary of NLIDBs with spider dataset.

Reference	Query Match Accuracy	Research Focus	Advantages	Disadvantages
Yu, Yasunaga et al. 2018 [56]	22.0%	Path history and column attention as input	Complex and cross domain text to SQL solution	Accuracy not usable
Lee 2019 [24]	24.3%	Clause wise separate prediction	Syntactic correctness	Overall accuracy still not up-to the usage level
Dong, Sun et al. 2019 [23]	29.9%	Learning based Sequential tagging	Novel method introduced for question tagging	Not ideal for complex and cross domain dataset
Lin, Bogin et al. 2019 [55]	33.8%	Schema dependent grammar for SQL	Minimize over generation	Schema elements not present in question are not parse able
Liu, Fang et al. 2019 [67]	40.4%	External Knowledge incorporation	Comparison operator accuracy improved	Not supporting for phrase level knowledge
Bogin, Gardner et al. 2019 [60]	40.7%	DB schema Encoding	Efficient from clause	Low accuracy overall
R Shin 2019 [?]	42.94%	Relation aware Self-attention among schema elements	Joins performance improvement	Not much impact at overall query match accuracy
Bogin, Gardner et al. 2019 [61]	47.4%	Global Gating, Discriminative Re-ranking	Correct query ratio increased	Aggregate functions not handled well
Yao, Su et al. 2019 [38]	52%	Query verification via human interaction	Query correctness improved	Performance based on Human interaction is unpredictable
Zhang, Yu et al. 2019 [45]	53.4%	Editing mechanism based on interaction history	Error propagation robustness	Performance is affected by the base model
Guo, Zhan et al. 2019 [58]	54.7%	Intermediate representation with domain specific language	Minimized Lexical problem (Out of Domain words)	Column name prediction face maximum ratio of errors
Wang and Shin et al. 2019 [46]	69%	Integrated schema linking with GNN schema graph for unseen DB	Cross domain generalization	High error rate for vague columns names
Kelkar, Relan et al. 2020 [25]	58.5%	Correct query selection from candidate queries	Query robustness	One re-ranker implemented for all hardness levels
Zeng, VLin et al. 2020 [33]	63.2%	Query correction based on user responses	Query robustness	Error analysis not provided
Brunner and Stockinger 2020 [36]	63% (execution accuracy)	Semantic gap coverage by including cell values	Vague Column and values selection improved	Multiple similar name columns not detectable
Choi, C Shin et al. 2020 [56]	66.6%	Sketch based slot filling approach for nested queries	Improved results for nested queries specifically	High error rate for vague columns names

TABLE 9. F1 scores of SQL component matching on the dev set.

Reference	SELECT	WHERE	GROUP BY	ORDER BY	KEYWORDS
Yu,Yasunaga et al. 2018 [56]	48.2%	31.6%	28.9%	58.4%	68.9%
Lee 2019 [24]	68.7%	39.0%	63.1%	63.5%	76.5%
Liu, Fang et al. 2019 [67]	63.2%	40.8%	48.1%	61.1%	-

Therefore, context information is often ignored, which can otherwise contribute to enhanced accuracy [39].

For this reason, sequence to sequence models are more practical to capture some of the context information of natural language questions and SQL queries. Until now, methods that are being used to solve order matter problems are either not as supportive to produce practical results or are creating new issues linked with them. Therefore, the order matter problem is still an open issue and needs more insights and novel ideas for better performance of the where clause and consequently better accuracy overall.

2) CONTEXT CONSIDERATION

Lack of context is another problem related to where clause accuracy issues. The sequence to set approach was adopted to solve the order matter problem in sequence to sequence models. But it faced another major issue of lack of context included while token prediction [39]. This issue is faced by sequence to set approaches mostly because their prediction is in groups and clauses, which is not dependent on the whole NL question or any previously decoded token in the sequence. Any additional information about Natural language questions or previously predicted tokens is counted as contextual

TABLE 10. Research challenges.

Reference	Research Challenges	Techniques
(Zhong, Xiong et al. 2017) [8], (Sun, Tang et al. 2018) [40] , (Dong, Sun et al. 2019) [23]	Order Matter Problem	Reinforcement Learning, Sequence to Set adoption
(Hosu, Iacob et al. 2018) [18] , (Lee 2019) [24], (Guo, Zhan et al. 2019) [58], (Sun, Tang et al. 2018) [40] , (Lin, Bogin et al. 2019) [55] , (Shi, Tatwawadi et al. 2018) [44], (Choi, C Shin et al. 2020) [57], (Guo and Gao 2019) [54]	Syntax problem, Runtime error correction	SQL Sketch generation, Grammar building, Execution Guidance
(Huang, Wang et al. 2018) [52]	Faster Learning, Cross Domain Adaptation	Pseudo Task Support Group generation, Meta Learning
(Xiong and Sun 2019) [51], (Guo, Zhan et al. 2019) [58], (Bogin, Gardner et al. 2019) [61], (Dong, Sun et al. 2019) [23], (Lin, Bogin et al. 2019) [55], (Wang, Tian et al. 2018) [17] , (Yu, Li et al. 2018) [50] , (Wang and Shin 2019) [46], (Choi, C Shin et al. 2020) [57], (Guo and Gao 2019) [53], (Brunner and Stockinger 2020) [36], (Wang 2019) [42]	Cross Domain Adaptation, lexical problem, Clear Column Names	Schema Linking/Entity linking/anonymization, Type Annotation, Supplemented Column Names, Leveraging database rules
(Zhang, Yu et al. 2019) [45] , (Yu, Li et al. 2018) [50], (Sun, Tang et al. 2018) [40], (Wang and Shin 2019) [46], (He, Mao et al. 2019) [47], (R Shin 2019) [59]	Context Consideration	Utterance-Column / Column-Column / Column-Cell attention mechanism, schema linking integrated with GNN schema graph
(Bogin, Gardner et al. 2019) [60], (Bogin, Gardner et al. 2019) [61], (Wang and Shin 2019) [46], (Guo and Gao 2019) [54]	Schema Information Input, Table values ad input	GNN Schema Graph Representation, BERT encoding
(Xiong and Sun 2019) [51], (Xu, Liu et al. 2018) [9]	Less Training Data	Data Augmentation
(Choi, C Shin et al. 2020) [57]	Efficient Joins	Joint Table filtering

information. Contextual information also impacts prediction accuracy. For Sequence to Sequence NLIDB, some extent of context is added implicitly as the prediction of one token is based on the previous output of the decoder [41]. On the other hand, sequence to set approach where a separate module is trained for each part of the sequence, implicit additional information involved is minimized. Therefore, they usually work based on string matching or local word to word effect. The context must be added explicitly to make prediction global instead of string matching or local word to word effect. NLP techniques support the basic Deep learning NLIDB to cope with related challenges.

Various Natural Language Processing (NLP) methods are adopted for context consideration problems. The attention mechanism is one of the NLP techniques used to find the relation between two components. An attention mechanism was initially introduced for the sequence to sequence models to enhance the context information from just one previous token to the whole sequence. For the NLIDB area, it was adopted with sequence to set initially to add some context information. The Self-attention version of this mechanism is widely adopted for text to SQL models. Running an attention mechanism between input elements is called self-attention [67]. The attention mechanism is to find the weighted relation between given components. Calculated score from the attention mechanism presents the importance of the parts of a sequence regarding one token. It maps the decoder with the hidden states of the complete input sequence providing a global effect for one particular token. The decoder has access to all the input tokens and their attention score. It can select any of them based on their importance regarding the one being predicted. Various combinations have been tried and tested to boost the performance of text to SQL model in terms of context consideration. Self-attention between columns, column to cells, cell values, and natural language tokens have experimented with a clear performance boost. Integration of schema linking with GNN schema graph has also been experimented with to consider additional contextual information for better results [46]. From current work, it is observed that besides boosting performance, condition accuracy is still not up to the point of practical usage yet and where clause still suffers from the most errors. Hence,

the attention mechanism for context consideration is still an open topic for more ideas so that it can be utilized effectively for better condition accuracy. Relevance functions are also used for context addition [64]. A numerical value is calculated based on relevance between that targeted token and query tokens as relevance score. Relevance score is further used to pick the relevant tokens from the sequence for the training and prediction process. However, these techniques boost the where clause performance a bit, indicating the potential future improvement. They have not been able to impact the process to the point of implementable NLIDB development. Therefore, context consideration is another open challenge that needs to be solved in this area.

3) SEMANTIC AND SYNTAX PROBLEM

The difficulty of text to SQL tasks lies mainly in the vagueness and complexity of Natural language. Therefore, understanding a user's intention regarding some clause, i.e., where, aggregator, etc., is the hardest part of this task [12]. Finding the correct structure of the SQL query according to the requirements in the natural language question is a syntax-related issue. Mapping the user's intention with the correct SQL structure is known as a syntax problem. Syntax problem is challenging because most of the time, structural information is not available in the question directly. It is predicted from the text analysis and the relationship between attributes and cell values. An example of such a task is interpreting the proper sense of adjectives mentioned in the question and mapping them with aggregator functions in SQL query. Methods to cope with this issue include using external knowledge as feature vectors [38]. External knowledge is some supporting material not exactly in the question or given database. An example of this method is portrayed by [67], gathering pairs of nouns and adjectives in the given domain, then making two clusters of positive and negative adjectives for one noun. Those pairs and their respective cluster representations were passed to the model as part of the input feature vector. Noun-adjective pair and positive/negative information provide additional information for deciding aggregate function according to the requirement in the NL question. Various other approaches simplify the syntax problem, but they are also connected with semantic issues. These two problems, i.e., syntactic problem and semantic problem, are correlated in many ways. The semantic problem is related to the terminology mapping of natural language questions and database entities. This is another Prime issue to map natural language question semantics with schema entities [23]. Words and terms used for required data in natural language queries can be different from the table/column names in the database. To map the right words from the NL question to the correct column/table names is this task's semantic or lexical challenge.

Various methods that have been proposed for their solution consider them correlated to each other. Most techniques focused on separating these two parts in the overall process so that they can be solved one by one. SQL sketch generation is one of the methods that have been used widely to deal with syntax before so that the prediction model can focus more on the semantic issue. In SQL sketch generation, the model predicts the SQL syntax overall as a first step, according to the natural language question requirements. That sketch is treated as a template at a later stage. Each slot is predicted and filled one at a time via separate prediction modules. This way, the model takes care of syntax issues separately and can focus more on semantic issues with given syntax. SQL sketch is predicted in different forms, i.e., set of rules to call other prediction modules one after the other, syntax trees, intermediate form for NL question and SQL query etc. intermediate representation is a middle form of NL question not having fully syntax or semantic structure. Later it is used to generate a complete SQL query. Grammar building or a set of rules is another helpful technique to tackle syntax problems [63]. Another way is to tackle the semantic issue first and focus syntax part later. Both of these methods have their own set of pros and cons along with their constraints. For the syntax first method, an additional module is required to predict the sketch only. For the semantic first method, anonymization or entity linking techniques are used. Anonymization means making a query anonymous for database content by tagging the column names and cell values as the column, and cell respectively. Only tag of column/cell/table value in an NL question is identified and not their exact value or exact table that they are related. With anonymizing the query, many queries become similar because of the absence of required values. One SQL syntax is predicted for each group of similar queries. These anonymized statements are fed as an input feature to the model for the full query. It acts as a piece of additional information for the model that later adds syntax information. Entity linking is another word for anonymization. In entity linking, database entities are identified beforehand, along with which table they are related to. [36] Utilize the entity linking in Question hints and schema hints. Question hints meaning question tokens contain the information of which tokens are most likely table, column or values based on schema information. Vice versa schema hints contain the significant column and table names according to the natural language question.

The joint table Filtering (JTF) method is also used to fine-tune the selection of tables in from clause [57]. In the context of JTF, irrelevant tables that are required to link the required table only are removed at training time added later with the help of a foreign key. This way, table noise is removed, and a bit of efficiency is included in the process. Discretely relevant and accurate table and question alignment is made. Supplement Column Names (SCN) [57] removes vagueness when the question tokens are tagged with schema elements. Column names are supplemented with the respective table names in case of similar column names in different tables. The intermediate representation is another technique used widely to generate a rough representation of NL questions that can provide the information of semantics in some structural way. They are in a semantic tree or replacing

database contents with predefined tokens [58]. Intermediate representation being middle statement is partially a SQL query and partially NL question. All of these are methods and techniques for semantic and syntax issues contributing to the existing performance of NLIDBs. But as observed from results achieved so far, condition accuracy is not up to the point of real-time implementation. Therefore, more ideas and combinations of these techniques to solve the syntax and semantic problem is a challenge.

B. DATASET CHALLENGES

NLIDB systems with a Deep learning model primarily using supervised learning. For supervised learning, labelled datasets are required for training purposes [32]. Dataset is an impactful part of any supervised learning task. Given that NLIDB with Deep learning is a comparatively recent area, it has more issues with the dataset. Available standard datasets for NLIDB have been discussed in section 4 and their problems and advantages. In this section, issues and challenges regarding the dataset in NLIDB are explained below.

1) UNAVAILABILITY OF DATASET

Deep learning end to end models requires labelled dataset for training purpose. Available datasets for text to SQL tasks are not complex enough to represent the real world questions asked by the users. Few available standard datasets for this purpose includes two latest one, i.e., Wikisql and Spider datasets [44]. These two are the most used datasets for NLIDB research and experimentation. They are convenient options in terms of comparisons and efficiency analysis. Wikisql consists of single table databases; therefore far too much simple as compared to actual user's queries. Thus, although some work has achieved significant performance on the Wikisql dataset, it cannot be considered for real-time usage.

Spider is the more recent dataset and is more complex than Wikisql. It contains cross-domain databases, multiple tables databases, and other clauses like order by, group by, etc. Still, Spider is also not complex enough to the level of practical usage. The number of examples that contain complex clauses like group by is not enough to train the end to end text to SQL model for these clauses. There are techniques to utilize DB logs for reverse creating the NL questions to create the labelled dataset. But such generated data suffer from biasness and cannot fully serve the purpose. Some augmentation methods are being used to expand existing complex data.

GAN based augmentation methods are one of those examples [17]. GAN based augmentation model is widely used in NLP (Natural Language Processing) area overall. In GAN based model, a generator is trained with some loss function to generate Natural language queries from given SQL building Natural language questions and SQL query dataset. A biased dataset can be a side effect of this method. However, such methods are evaluated later during human crowdsourcing with a random set of samples. Still, data generated from such a model cannot be as original and complex as human-generated data. Therefore, a biased dataset is a significant issue of these methods. Some manual techniques have been adopted to make the process more authentic. One of them includes tagging the schema related information in natural language questions. When taking schema information out, many statements might look similar [51]. Similar statements are grouped to form clusters of similar types. Finally, their multiple possible combinations are generated, expanding the existing data. This method involves human effort at many levels. Therefore, We can count it as more authentic than the previously discussed ones. Overall, these methods coping with this dataset issues are working well in the case of fewer datasets [52]. But they are not enough to build an NLIDB system for actual usage. Some solution for complex dataset issue is still an open challenge in this area.

2) CROSS DOMAIN ADOPTABILITY

Cross-domain adaptability of a model is another open challenge for NLIDB. As it is not practical to train a model from scratch every time dealing with a new domain, the recent task Spider dataset includes the databases containing multiple domain data, providing the option to train and test the cross-domain adaptability. A model needs to handle unseen datasets to effective results on that dataset [58]. Therefore, cross-domain adaptability has been a focus in recent models to experiment with spider datasets. Some of the methods adopted for syntax and schema issues have been useful for cross-domain adaptability issues. General ideas to deal with domain issues consist of separating the schema information from syntax. Common methods adopted from the NLP area to mitigate this issue include anonymization or entity linking. For anonymization, schema-related information replaces predefined tokens and makes the overall statement anonymous regarding any schema. After anonymized statements become schema independent therefore can be utilized to train the model generally. Editing based mechanism is another approach for cross-domain adaptability [43]. A partial query is evaluated and edited for improvement in the target domain. The intermediate representation is another way to tackle the cross-domain issue. An intermediate form of natural language question and SQL query is generated containing syntax and predefined context-free grammar. These methods are an initiative toward tackling this problem but not providing a complete solution yet. Therefore, cross-domain adaptability is included in open issues as well.

3) CAPTURING SCHEMA INFORMATION AS INPUT FEATURE

For Deep learning models for the text to SQL task, basic input features are labelled data of Natural language questions and SQL queries for training. Besides these two schemas, related information is treated as an additional input feature. The natural language question is represented in tokens and their word embedding for the model testing phase [60]. Regarding the schema information, it varies with every model that how much information they utilize. Most commonly, models consider only column names as schema information. It varies

up to tables' names, column names, column data types and sometimes cell values. It seems that schema information is not being considered fully for this task until now [47]. In sequence to set models, other clauses are handled one by one except the ''from'' clause. Models predict the ''from'' clause based on their supervised training and do not have schema relations information. Additional schema information such as relationships of the tables can help planning a way to predict from clause as well, instead of relying on training only. Therefore, how much schema information can be integrated with input features and influence the output is an issue that needs to be resolved [59]. Few models have been proposed to convert the schema information into a graph with a GNN model. Later calculating the weight of nodes and edges based on their relationship and overall impact. Finally, they used their weighted nodes and edges to capture the relationships among entities completely. Including the schema information as part of the input can improve logical form accuracy and query efficiency regarding the number of tables joining in the query [46]. Recently the significant amount of work for NLIDB focused on the schema encoding issue. But it still has room for potential improvement. Therefore, it needs more attention and is considered an open challenge for NLIDB with the Deep learning area.

VIII. CONCLUSION

In this work, we have reviewed the NLIDB with Deep learning in-depth. We have summarized the findings of the area, highlighted the issues and challenges, and discussed the methods and techniques proposed to cope with them. Recent work has been compared to emphasize achieved performance and to find out limitations linked with them. Some tools and techniques that have been adopted to solve the challenges are also explained, along with their pros and cons. It has been concluded from this review paper that supervised learning with RNN models is most used for NLIDB systems. Sequence to Sequence and Sequence to set are two basic approaches for building text to SQL models. Both of these have their respective limitations and advantages as well. Such as sequence to sequence approaches face order matter problem and sequence to set approaches face the lack of context. Currently, both methods are being adopted and experimented in parallel, and efforts are being invested in mitigating their issues. Reinforcement learning is used on top of the standard sequence to sequence model to minimize the order matter problem. For the lack of context problem in sequence to set approaches, attention mechanisms are used widely to add the explicit connection and context for tokens. Generally, condition related problems are dominated in the area along with dataset issues. NLP techniques are being used in various combinations along with machine learning ideas to minimize the issues. Although NLIDB systems' accuracy is not high enough for practical usage, recent work is promising enough to foresee the potential possibilities. More research insights and ideas are needed to mitigate the problems related to condition accuracy. Finally, the available datasets i.e; wikisql, Spider dataset etc are not complex enough to train the model for real-time usage and are observed to be a hurdle for advancements in the area. More ideas are required in that context also. Overall, besides all the efforts, it is still an unsolved area that is open for work. For future work, more datasets should be fine tuned with real time complex queries for the purpose of training and testing these models. Furthermore, attention mechanisms can be explored more and find the possibilities to adopt in this area as they have shown the potential to improve the overall accuracy by adding explicit context. ''Group by'' clause and ''order by'' has been the least focus, mainly because of unavailability of datasets that contain enough training example with these clauses. Therefore, working on these clauses can also bring the overall accuracy near to the real time usage.

REFERENCES

[1] 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,'' *Proc. VLDB Endowment*, vol. 9, no. 12, pp. 1209–1220, 2016.

[2] D. H. D. Warren and F. C. N. Pereira, ''An efficient easily adaptable system for interpreting natural language queries,'' *Comput. Linguistics*, vol. 8, nos. 3–4, pp. 110–122, Jul./Dec. 1982.

[3] D. A. Poetra, T. Esterina Widagdo, and F. N. Azizah, ''Natural language interface to database (NLIDB) for query with temporal aspect,'' in *Proc. Int. Conf. Data Softw. Eng. (ICoDSE)*, Nov. 2019, pp. 1–6.

[4] T. J. O'Shea and J. Hoydis, ''An introduction to deep learning for the physical layer,'' *IEEE Trans. Cogn. Commun. Netw.*, vol. 3, no. 4, pp. 563–575, Oct. 2017.

[5] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapé, ''Mobile encrypted traffic classification using deep learning: Experimental evaluation, lessons learned, and challenges,'' *IEEE Trans. Netw. Service Manag.*, vol. 16, no. 2, pp. 445–458, Feb. 2019.

[6] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapé, ''Toward effective mobile encrypted traffic classification through deep learning,'' *Neurocomputing*, vol. 409, pp. 306–315, Oct. 2020.

[7] I. Sutskever, O. Vinyals, and Q. V. Le, ''Sequence to sequence learning with neural networks,'' in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 3104–3112.

[8] V. Zhong, C. Xiong, and R. Socher, ''Seq2SQL: Generating structured queries from natural language using reinforcement learning,'' 2017, *arXiv:1709.00103*.

[9] X. Xu, C. Liu, and D. Song, ''SQLNet: Generating structured queries from natural language without reinforcement learning,'' 2017, *arXiv:1711.04436*.

[10] I. Androutsopoulos, G. Ritchie, and P. Thanisch, ''Natural language interfaces to databases–An introduction,'' *Natural Lang. Eng.*, vol. 1, no. 1, pp. 29–81, 1995.

[11] N. Yaghmazadeh, Y. Wang, I. Dillig, and T. Dillig, ''SQLizer: Query synthesis from natural language,'' *Proc. ACM Program. Lang.*, vol. 1, pp. 1–26, Oct. 2017.

[12] C. Baik, H. V. Jagadish, and Y. Li, ''Bridging the semantic gap with SQL query logs in natural language interfaces to databases,'' in *Proc. IEEE 35th Int. Conf. Data Eng. (ICDE)*, Apr. 2019, pp. 374–385.

[13] C. Wang, A. Cheung, and R. Bodik, ''Synthesizing highly expressive SQL queries from input-output examples,'' in *Proc. 38th ACM SIGPLAN Conf. Program. Lang. Design Implement.*, Jun. 2017, pp. 452–466.

[14] A. Giordani and A. Moschitti, ''Generating SQL queries using natural language syntactic dependencies and metadata,'' in *Proc. Int. Conf. Appl. Natural Lang. Inf. Syst.* Springer, 2012, pp. 164–170.

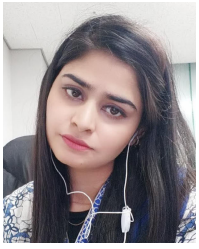
[15] N. Yaghmazadeh, Y. Wang, I. Dillig, and T. Dillig, ''Type- and content-driven synthesis of SQL queries from natural language,'' 2017, *arXiv:1702.01168*.

[16] F. Li and H. V. Jagadish, ''NaLIR: An interactive natural language interface for querying relational databases,'' in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2014, pp. 709–712.

[17] W. Wang, Y. Tian, H. Xiong, H. Wang, and W.-S. Ku, ''A transfer-learnable natural language interface for databases,'' 2018, *arXiv:1809.02649*.

[18] I. A. Hosu, R. C. A. Iacob, F. Brad, S. Ruseti, and T. Rebedea, "Natural language interface for databases using a dual-encoder model," in *Proc. 27th Int. Conf. Comput. Linguistics*, 2018, pp. 514–524.
[19] S. Yavuz, I. Gur, Y. Su, and X. Yan, "What it takes to achieve 100% condition accuracy on wikisql," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 1702–1711.
[20] S. A. Bukhari, H. S. Dar, M. I. Lali, F. Keshtkar, K. M. Malik, and S. Kadry, "Frameworks for querying databases using natural language: A literature review–NLP-to-DB querying frameworks," *Int. J. Data Warehousing Mining*, vol. 17, no. 2, pp. 21–38, 2021.
[21] E. U. Reshma and P. C. Remya, "A review of different approaches in natural language interfaces to databases," in *Proc. Int. Conf. Intell. Sustain. Syst. (ICISS)*, Dec. 2017, pp. 801–804.
[22] K. Affolter, K. Stockinger, and A. Bernstein, "A comparative survey of recent natural language interfaces for databases," *VLDB J.*, vol. 28, no. 5, pp. 793–819, Oct. 2019.
[23] Z. Dong, S. Sun, H. Liu, J.-G. Lou, and D. Zhang, "Data-anonymous encoding for text-to-SQL generation," in *Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP)*, 2019, pp. 5408–5417.
[24] D. Lee, "Clause-wise and recursive decoding for complex and cross-domain text-to-SQL generation," in *Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP)*, 2019, pp. 6045–6051.
[25] A. Kelkar, R. Relan, V. Bhardwaj, S. Vaichal, C. Khatri, and P. Relan, "Bertrand-DR: Improving text-to-SQL using a discriminative re-ranker," 2020, *arXiv:2002.00557*.
[26] C. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. Bethard, and D. McClosky, "The Stanford CoreNLP natural language processing toolkit," in *Proc. 52nd Annu. Meeting Assoc. Comput. Linguistics: Syst. Demonstrations*, 2014, pp. 55–60.
[27] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2014, pp. 1532–1543.
[28] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, vol. 1, Jun. 2019, pp. 4171–4186.
[29] C. Finegan-Dollak, J. K. Kummerfeld, L. Zhang, K. Ramanathan, S. Sadasivam, R. Zhang, and D. Radev, "Improving text-to-SQL evaluation methodology," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, 2018, pp. 351–360.
[30] P. Wang, T. Shi, and C. K. Reddy, "Text-to-SQL generation for question answering on electronic medical records," in *Proc. Web Conf.*, Apr. 2020, pp. 350–361.
[31] A. Elgohary, S. Hosseini, and A. Hassan Awadallah, "Speak to your parser: Interactive text-to-SQL with natural language feedback," in *Proc. 58th Annu. Meeting Assoc. Comput. Linguistics*, 2020, pp. 2065–2077.
[32] T. Yu, R. Zhang, K. Yang, M. Yasunaga, D. Wang, Z. Li, J. Ma, I. Li, Q. Yao, S. Roman, Z. Zhang, and D. Radev, "Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-SQL task," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 3911–3921.
[33] T. Charles Hemphill, J. John Godfrey, and R. George Doddington, "The ATIS spoken language systems pilot corpus," in *Proc. Speech Natural Lang.: Workshop Held Hidden Valley, Pennsylvania*, Jun. 1990, pp. 1–6.
[34] A. D. Deborah, B. Madeleine, B. Michael, F. William, H.-S. Kate, P. David, P. Christine, R. Alexander, and S. Elizabeth, "Expanding the scope of the ATIS task: The ATIS-3 corpus," in *Proc. Workshop Hum. Lang. Tech. (HLT)*, Stroudsburg, PA, USA, 1994, pp. 1–6.
[35] T. Yu, R. Zhang, M. Yasunaga, Y. Chern Tan, X. Victoria Lin, S. Li, H. Er, I. Li, B. Pang, T. Chen, E. Ji, S. Dixit, D. Proctor, S. Shim, J. Kraft, V. Zhang, C. Xiong, R. Socher, and D. Radev, "SParC: Cross-domain semantic parsing in context," 2019, *arXiv:1906.02285*.
[36] U. Brunner and K. Stockinger, "ValueNet: A neural text-to-SQL architecture incorporating values," *Proc. VLDB Endowment*, pp. 1–14, May 2020.
[37] L. Dong and M. Lapata, "Coarse-to-fine decoding for neural semantic parsing," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, Jul. 2018, pp. 731–742.
[38] Z. Yao, Y. Su, H. Sun, and W.-T. Yih, "Model-based interactive semantic parsing: A unified framework and a text-to-SQL case study," in *Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP)*, 2019, pp. 5447–5458.
[39] G. Huilin, G. Tong, W. Fan, and M. Chao, "Bidirectional attention for SQL generation," in *Proc. IEEE 4th Int. Conf. Cloud Comput. Big Data Anal. (ICCCBDA)*, Apr. 2019, pp. 676–682.
[40] Y. Sun, D. Tang, N. Duan, J. Ji, G. Cao, and X. Feng, "Semantic parsing with syntax-and table-aware SQL generation," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, Jul. 2018, pp. 361–372.
[41] C. Wang, M. Brockschmidt, and R. Singh, "Pointing out SQL queries from text," *Microsoft Res.*, pp. 1–12, Aug. 2018.
[42] W. Wang, "A cross-domain natural language interface to databases using adversarial text method," in *Proc. CEUR Workshop*, vol. 2399, 2019, pp. 1–4.
[43] J. Zeng, X. V. Lin, S. C. H. Hoi, R. Socher, C. Xiong, M. Lyu, and I. King, "Photon: A robust cross-domain text-to-SQL system," in *Proc. 58th Annu. Meeting Assoc. Comput. Linguistics: Syst. Demonstrations*, 2020, pp. 204–214.
[44] T. Shi, K. Tatwawadi, K. Chakrabarti, Y. Mao, O. Polozov, and W. Chen, "IncSQL: Training incremental text-to-SQL parsers with non-deterministic oracles," 2018, *arXiv:1809.05054*.
[45] R. Zhang, T. Yu, H. Er, S. Shim, E. Xue, X. V. Lin, T. Shi, C. Xiong, R. Socher, and D. Radev, "Editing-based SQL query generation for cross-domain context-dependent questions," in *Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP)*, 2019, pp. 5338–5349.
[46] B. Wang, R. Shin, X. Liu, O. Polozov, and M. Richardson, "RAT-SQL: Relation-aware schema encoding and linking for text-to-SQL parsers," in *Proc. 58th Annu. Meeting Assoc. Comput. Linguistics*, 2020, pp. 757–7567.
[47] P. He, Y. Mao, K. Chakrabarti, and W. Chen, "X-SQL: Reinforce schema representation with context," 2019, *arXiv:1908.08113*.
[48] Z. Lu, H. Li, and B. Kao, "Neural enquirer: Learning to query tables in natural language," *IEEE Data Eng. Bull.*, vol. 39, no. 3, pp. 63–73, Dec. 2016.
[49] C. Lawrence and S. Riezler, "Nlmaps: A natural language interface to query openstreetmap," in *Proc. 26th Int. Conf. Comput. Linguistics: Syst. Demonstrations (COLING)*, 2016, pp. 6–10.
[50] T. Yu, Z. Li, Z. Zhang, R. Zhang, and D. Radev, "TypeSQL: Knowledge-based type-aware neural text-to-SQL generation," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol. (NAACL-HLT)*, New Orleans, LA, USA, vol. 2, Jun. 2018, pp. 588–594.
[51] H. Xiong and R. Sun, "Transferable natural language interface to structured queries aided by adversarial generation," in *Proc. IEEE 13th Int. Conf. Semantic Comput. (ICSC)*, Jan. 2019, pp. 255–262.
[52] P.-S. Huang, C. Wang, R. Singh, W.-T. Yih, and X. He, "Natural language to structured query generation via meta-learning," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol.*, vol. 2, Jun. 2018, pp. 732–738.
[53] T. Guo and H. Gao, "Using database rule for weak supervised text-to-SQL generation," 2019, *arXiv:1907.00620*.
[54] T. Guo and H. Gao, "Content enhanced BERT-based text-to-SQL generation," 2019, *arXiv:1910.07179*.
[55] K. Lin, B. Bogin, M. Neumann, J. Berant, and M. Gardner, "Grammar-based neural text-to-SQL generation," 2019, *arXiv:1905.13326*.
[56] T. Yu, M. Yasunaga, K. Yang, R. Zhang, D. Wang, Z. Li, and D. Radev, "SyntaxSQLNet: Syntax tree networks for complex and cross-domaintext-to-SQL task," 2018, *arXiv:1810.05237*.
[57] D. Choi, M. C. Shin, E. Kim, and D. R. Shin, "RYANSQL: Recursively applying sketch-based slot fillings for complex text-to-SQL in cross-domain databases," *Comput. Linguistics*, vol. 47, no. 2, pp. 1–24, May 2021.
[58] J. Guo, Z. Zhan, Y. Gao, Y. Xiao, J.-G. Lou, T. Liu, and D. Zhang, "Towards complex text-to-SQL in cross-domain database with intermediate representation," in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, 2019, pp. 4524–4535.
[59] R. Shin, "Encoding database schemas with relation-aware self-attention for text-to-SQL parsers," 2019, *arXiv:1906.11790*.
[60] B. Bogin, J. Berant, and M. Gardner, "Representing schema structure with graph neural networks for text-to-SQL parsing," in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, 2019, pp. 4560–4565.
[61] B. Bogin, M. Gardner, and J. Berant, "Global reasoning over database structures for text-to-SQL parsing," in *9th Int. Joint Conf. Natural Lang. Process.*, Nov. 2019, pp. 3659–3664.
[62] D. Lee, J. Yoon, J. Song, S. Lee, and S. Yoon, "One-shot learning for text-to-SQL generation," 2019, *arXiv:1905.11499*.
[63] W. Hwang, J. Yim, S. Park, and M. Seo, "A comprehensive exploration on wikisql with table-aware word contextualization," in *Proc. 33rd Conf. Neural Inf. Process. Syst. (NeurIPS)*, 2019, pp. 1–34.

[64] C. Wang, K. Tatwawadi, M. Brockschmidt, P.-S. Huang, Y. Mao, O. Polozov, and R. Singh, "Robust text-to-SQL generation with execution-guided decoding," 2018, *arXiv:1807.03100*.
[65] I. Gur, S. Yavuz, Y. Su, and X. Yan, "DialSQL: Dialogue based structured query generation," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, 2018, pp. 1339–1349.
[66] Q. Lyu, K. Chakrabarti, S. Hathi, S. Kundu, J. Zhang, and Z. Chen, "Hybrid ranking network for text-to-SQL," 2020, *arXiv:2008.04759*.
[67] H. Liu, L. Fang, Q. Liu, B. Chen, J.-G. Lou, and Z. Li, "Leveraging adjective-noun phrasing knowledge for comparison relation prediction in text-to-SQL," in *Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP)*, 2019, pp. 3515–3520.
[68] Z. Bai, B. Yu, B. Wu, Z. Wang, and B. Wang, "Learning to generate structured queries from natural language with indirect supervision," *Comput. Speech Lang.*, vol. 67, May 2021, Art. no. 101185.



SHANZA ABBAS received the B.S. degree in information technology from the University of the Punjab, Pakistan, and the M.S. degree in software engineering from the National University of Science and Technology, Pakistan. She is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering, Hanyang University, South Korea, funded by the Higher Education Commission of Pakistan, in 2018. Her research interest includes natural language interface to database systems with machine learning.

MUHAMMAD UMAIR KHAN received the B.S. degree in computer engineering from the Balochistan University of Information Technology and Management Science, Pakistan, and the M.S. degree in computer engineering from the Lahore University of Management Sciences, Pakistan. He is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering, Hanyang University, South Korea, funded by the Higher Education Commission of Pakistan. His research interests include android development, program analysis, and machine learning.

SCOTT UK-JIN LEE (Member, IEEE) received the B.S. degree in software engineering and the Ph.D. degree in computer science from The University of Auckland, New Zealand. After the Ph.D. degree, he was a Postdoctoral Research Fellow with the Commissariat à l'Énergie Atomique et aux Énergies Alternatives, France. He has been with the Department of Computer Science and Engineering, Hanyang University, ERICA Campus, South Korea, since 2011. He is currently working as an Associate Professor with the Department of Computer Science and Engineering, Major in Bio Artificial Intelligence. His research interests include software engineering, formal methods, and quality assurance. He is also a member of the Korean Institute of Information Scientists and Engineers and the Korean Society of Computer and Information. He has served as an editor, the technical chair, and a committee member for several journals and conferences.

ASAD ABBAS received the B.S. degree in information technology from the University of the Punjab, Pakistan, in 2011, the M.S. degree leading to the Ph.D. degree from the Department of Computer Science and Engineering, Hanyang University ERICA Campus, Ansan, South Korea, funded by the Higher Education Commission, Pakistan, in 2014, and the Ph.D. degree in computer science and engineering from Hanyang University, in August 2018. He served as an Assistant Professor for the Department of Software Engineering, The University of Lahore, Pakistan. He is currently serving as an Assistant Professor for the Faculty of Information and Technology, University of Central Punjab (UCP), Lahore, Pakistan. His research interests include software product line, the IoT systems and applications and embedded software applications, big data analytics, and machine learning.

ALI KASHIF BASHIR (Senior Member, IEEE) received the B.Sc. degree (Hons.) in computer forensics and security from the Department of Computing and Mathematics, Manchester Metropolitan University, U.K., and the Ph.D. degree in computer science and engineering from Korea University, South Korea. His past assignments include an Associate Professor of ICT with the University of the Faroe Islands, Denmark; Osaka University, Japan; Nara College, National Institute of Technology, Japan; the National Fusion Research Institute, South Korea; Southern Power Company Ltd., South Korea; and Seoul Metropolitan Government, South Korea. He is currently a Senior Lecturer/an Associate Professor and the Program Leader of the Department of Computing and Mathematics, Manchester Metropolitan University. He is also with the School of Electrical Engineering and Computer Science, National University of Science and Technology (NUST), Islamabad, as an Adjunct Professor, and the School of Information and Communication Engineering, University of Electronics Science and Technology of China (UESTC) as an Affiliated Professor and the Chief Advisor of the Visual Intelligence Research Center. He has worked on several research and industrial projects of South Korean, Japanese and European agencies and Government ministries. In his career, he has received over 2.5 Million USD funding. He has authored over 180 research articles, received funding as the PI and the Co-PI from research bodies of South Korea, Japan, EU, U.K., and Middle East, and supervising/co-supervising several graduate (M.S. and Ph.D.) students. His research interests include the Internet of Things, wireless networks, distributed systems, network/cyber security, network function virtualization, and machine learning. Dr. Bashir is a member of the IEEE Industrial Electronic Society and ACM. He is leading many conferences as the chair (program, publicity, and track) and had organized workshops in flagship conferences, like IEEE Infocom, IEEE Globecom, and IEEE Mobicom. He is serving as the Editor in-Chief for the IEEE Future Directions Newsletter. He is also serving as an Area Editor for *KSII Transactions on Internet and Information Systems*, and an Associate Editor for *IEEE Internet of Things Magazine*, IEEE Access, *Computer Science*, *IET Quantum Communication*, and *Journal of Plant Diseases and Protection*. He is a Distinguished Speaker of ACM.