Natural Language Interfaces for Tabular Data Querying and Visualization: A Survey

Weixu Zhang, Yifei Wang, Yuanfeng Song, Victor Junqiu Wei, Yuxing Tian, Yiyan Qi, Jonathan H. Chan, Raymond Chi-Wing Wong, and Haiqin Yang, *Senior Member, IEEE*

Abstract—The emergence of natural language processing has revolutionized the way users interact with tabular data, enabling a shift from traditional query languages and manual plotting to more intuitive, language-based interfaces. The rise of large language models (LLMs) such as ChatGPT and its successors has further advanced this field, opening new avenues for natural language processing techniques. This survey presents a comprehensive overview of natural language interfaces for tabular data querying and visualization, which allow users to interact with data using natural language queries. We introduce the fundamental concepts and techniques underlying these interfaces with a particular emphasis on semantic parsing, the key technology facilitating the translation from natural language to SQL queries or data visualization commands. We then delve into the recent advancements in Text-to-SQL and Text-to-Vis problems from the perspectives of datasets, methodologies, metrics, and system designs. This includes a deep dive into the influence of LLMs, highlighting their strengths, limitations, and potential for future improvements. Through this survey, we aim to provide a roadmap for researchers and practitioners interested in developing and applying natural language interfaces for data interaction in the era of large language models.

Index Terms—Natural Language Interface, Text-to-SQL, Text-to-Visualization, Semantic Parsing, Large Language Models

1 Introduction

TABULAR, or structured, data form the backbone of many fields in today's digital age, including business, healthcare, and scientific research [57], [81]. However, the ability to interact effectively and efficiently with vast amounts of structured data to extract valuable insights remains a crucial challenge. Traditional methods of interaction, such as querying with structured query languages or manual plotting a visualization, often require a significant degree of technical expertise, thereby limiting their accessibility to a wider user base [2].

With the emergence of natural language processing technologies, the way we interact with structured data begin to shift. These technologies enable the development of Natural Language Interfaces, making tabular data querying and visualization more intuitive and accessible. Through these interfaces, users can extract information from databases or generate visual representations of data using natural language queries and commands [47], [93]. This shift towards language-based interfaces marks a significant stride towards simplifying data interaction, making it more user-friendly and accessible to non-technical users.

The foundational technologies powering these language-based interfaces are rooted in semantic parsing tasks, which transform natural language queries into formal representations tailored for execution on structured databases [50]. While various formal languages and functional representations have been introduced for this purpose, such as Prolog, Datalog, and FunQL, two are particularly dominant in tabular data interaction: SQL for data querying and visualization specifications for data visualization. SQL has been the de facto standard for querying relational databases for decades, offering comprehensive operations to retrieve and manipulate data. Visualization specifications provide a structured way to represent complex visualizations, making them an integral part of the data visualization process. Given their importance and widespread use, this survey will primarily focus on these two types of representations, delving deep into the challenges and advancements in the tasks of translating natural language into SQL and visualization specifications. In this context, the Text-to-SQL task [133] acts as a bridge converting user queries into SQL instructions, while the Text-to-Vis task [71] facilitates the transformation from user visualization requests into visualization specifications.

The development of these two semantic parsing tasks has evolved significantly over the years, driven by advancements in machine learning and natural language processing techniques. Early approaches often rely on rule-based or template-based [1], [50] systems and shallow parsing techniques. However, these methods struggle with complex queries and visualizations and are sensitive to the specific phrasing of the user's input. Introducing neural networks and deep learning methods brings about a significant leap

W. Zhang (Xi'an Jiaotong University, email: weixu_zhang@stu.xjtu.edu.cn), Y. Wang (University of Toronto, email: yifeii.wang@mail.utoronto.ca), and Y. Tian (Xidian University, email: tianyxxx@stu.xidian.edu.cn) are interns at International Digital Economy Academy (IDEA), Shenzhen, China. Y. Song is with WeBank Co., Ltd, Shenzhen, China. Email: yfsong@webank.com. V. J. Wei is with Department of Computer Science and Engineering, Hong Kong University of Science and Technology (HKUST), Hong Kong. Email: victorwei@cse.ust.hk. Y. Qi is with IDEA, Shenzhen, China. Email: qiyiyan@idea.edu.cn. J. H. Chan is with Innovative Cognitive Computing (IC2) Research Center at School of Information Technology, King Mongkut's University of Technology Thonburi. Email: jonathan@sit.kmutt.ac.th. R. C. Wong is with Department of Computer Science and Engineering, HKUST, Hong Kong. Email: raywong@cse.ust.hk. H. Yang (corresponding author) is affiliated with IDEA. Email: hqyang@ieee.org.

in performance. These methods, often based on sequence-to-sequence models [53], can capture more complex patterns in the data and are more robust to variations in the input. However, they still require substantial amounts of training data and struggle with out-of-domain queries [14]. The rise of Pretrained Language Models (PLMs), such as BERT [16], T5 [85], GPT [79], marks a turning point in the field. With their ability to leverage pre-training on vast amounts of text data, PLMs have shown remarkable success in a wide range of natural language processing tasks, including Text-to-SQL and Text-to-Vis. Recently, the advent of Large Language Models (LLMs) such as ChatGPT and the exploration of prompt engineering techniques have opened new avenues for developing more effective and user-friendly natural language interfaces for data interaction.

The interdisciplinary research on natural language interfaces for tabular data querying and visualization incorporates multiple research aspects, such as natural language processing and data mining, with advancements often following diverse and distinct trajectories. Despite its increasing importance, no single study has comprehensively reviewed the problem of semantic parsing for both querying and visualization tasks in a systematic and unified manner. As the field continues to evolve and grow, there is an increasing need to organize the research landscape, categorize current work, and identify knowledge gaps.

While there have been several prior efforts to summarize advances in this area, they have primarily focused on the early approaches and subsequent deep learning developments in querying and visualization [1], [14], [47], [53], [93], respectively, and do not offer a consolidated view of these intertwined domains. Furthermore, to the best of our knowledge, no existing surveys cover the recent advances by LLMs in these areas. The profound influence of LLMs, such as ChatGPT and its successors, on the Natural Language Interfaces for data querying and visualization is a rapidly growing area that requires more attention and exploration.

This survey aims to fill these gaps by offering a detailed overview of natural language interfaces for tabular data querying and visualization. We source references from key journals and conferences over the past two decades, spanning Natural Language Processing, Human-Computer Interaction, Data Mining, and Visualization. Our search is guided by terms such as ”Natural Language Interface”, ”Visualization”, and ”Text-to-SQL”, and we also explore cited publications to capture foundational contributions. We set out to address a set of critical research questions that can guide our understanding of natural language interfaces for tabular data and visualization:

- How have natural language interfaces evolved over time?
- How have recent advancements, especially LLMs, influenced the field?
- What are the inherent strengths and weaknesses of the existing methods?

Through this survey, we aim to provide informed and insightful answers to these questions by drawing upon an extensive literature review and analysis. We will delve into functional representations, datasets, evaluation metrics, and system architectures, particularly emphasizing the influence of LLMs. Our aim is to present a clear and succinct overview of the current state of the art, emphasizing existing approaches’ strengths and limitations while exploring potential avenues for future enhancements.

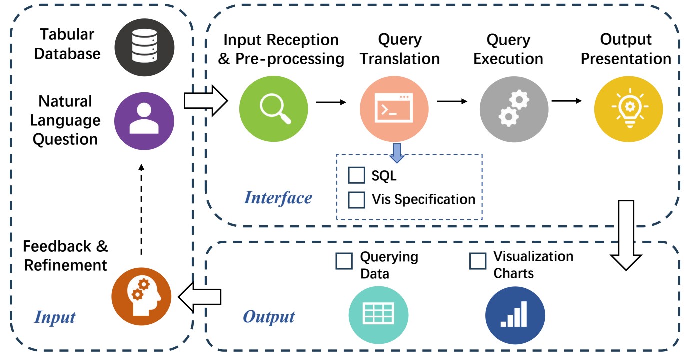

Fig. 1. Schematic representation of natural language interfaces for tabular data querying and visualization

2 Background and Framework

2.1 Context

The need for Natural Language Interfaces to process tabular data arises from the growing importance of data-driven decision-making across various industries, making it a crucial ability to interact with data efficiently and intuitively. Natural language interfaces simplify access to valuable insights by enabling a wider user base, including those without technical expertise, to query and visualize structured data [2].

Figure 1 shows the workflow of natural language interfaces for tabular data querying and visualization, where the user provides input in the form of a natural language question targeting a specific structured database. The interface pre-processes this input, translating it into functional representations, such as SQL queries for data extraction or visualization specifications for chart generation. Executing the SQL queries retrieves relevant data from the database, and the visualization specifications produce corresponding charts. The resulting output, whether raw data or visuals, is then presented to the user, who can provide feedback or further refine their query. This streamlined process enables users to extract data insights and generate visuals without diving into the complexities of databases or visualization tools merely by posing their questions.

The practical application of natural language interfaces for tabular data querying and visualization is exemplified in several existing tools. Microsoft’s Power BI [84], for instance, includes a feature called Q&A which allows users to ask natural language questions about their data and receive answers in the form of charts or tables. This feature leverages advanced natural language processing to understand the question and generate appropriate visualizations, thereby simplifying the process of data exploration for users. Similarly, Tableau [102], a popular data visualization tool, includes a feature named Ask Data. Users can type a question, and the system generates an answer through a data visualization. These applications underscore the potential and impact of natural language interfaces in enhancing the accessibility and usability of data interaction.

Table: sales data for a company

Quarter	Product Category	Sales
Q1	Electronics	3732
Q1	Clothing	3607
Q2	Electronics	1763
Q2	Clothing	2731
Q3	Electronics	1277
Q3	Clothing	2778
Q4	Electronics	4544
Q4	Clothing	3648

Tabular Database

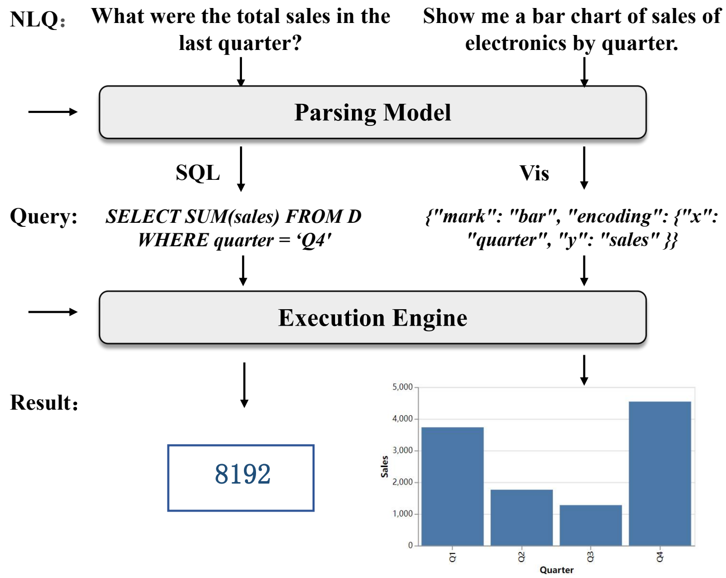

Fig. 2. Example of the process of translating natural language queries to SQL and visualization specifications on sales data. A textual query about quarterly sales is parsed into an SQL command to fetch numerical data, and a request for a sales visualization is transformed into the corresponding bar chart specification.

2.2 Problem Definition

In the context of natural language interfaces for tabular data, the central problem is to parse a natural language query into a functional representation that can be executed on a structured database. There are various formal languages designed for this purpose.

• *SQL (Structured Query Language).* SQL is a domain-specific language specifically designed for managing and querying data held in relational databases [133]. It provides a standardized protocol to retrieve, update, insert, or delete data from databases. SQL's structure allows for precise definitions of data relationships, enabling a wide range of inquiries and data manipulations.

• *Visualization Specifications.* Visualization Specifications are structured definitions that determine how data should be presented visually, often in the form of charts, graphs, or other graphical elements [93]. Common formats for visualization specifications include Vega-Lite and D3.js, which provide a high-level grammar for visualizing data. These specifications allow for a wide range of visualizations, from simple bar and line charts to more complex scatter plots and heat maps.

• *Prolog and Datalog.* Both Prolog and Datalog offer logic-based paradigms for database querying. Prolog is primarily recognized for its role in artificial intelligence and symbolic reasoning, while Datalog is a subset of Prolog tailored for database operations. Their declarative nature allows users to define what they want without explicitly detailing how to retrieve it.

• *FunQL.* FunQL (Functional Query Language) is designed to bridge the natural language and database query languages, making it particularly suited for the realm of semantic parsing. It's a functional representation that maps natural language constructs into structured queries, emphasizing the relationships between entities.

While an assortment of formal languages and functional representations have been introduced for structured data interaction, SQL and visualization specifications remain the linchpins in tabular data analysis due to their widespread adoption and comprehensive capabilities.

Formally, given a natural language query q, the task of the semantic parser P is to translate q into a functional expression e. This functional expression can take various forms depending on the task at hand. For a Text-to-SQL task, e would be an SQL query; for a Text-to-Vis task, e would be a visualization specification. Formally, this translation can be represented as:

$$P(q) \rightarrow e$$

Once the functional expression e is generated, it can be executed on the structured database D by an execution engine E to produce a result r. This execution can be represented as:

$$E(e, D) \rightarrow r$$

The overall process can therefore be seen as a translation from a natural language query q to a result r, facilitated by a semantic parser P and an execution engine E.

To provide a concrete example as shown in Fig. 2, consider a database D containing a company's sales data, a query q : "What were the total sales in the last quarter?" is processed by the semantic parser P, generating the SQL expression e : *SELECT SUM(sales) FROM D WHERE quarter = 'Q4'* . The execution engine E runs this query on D to yield the result r, representing the last quarter's total sales. Similarly, for a visualization request q : "Show me a bar chart of sales of electronics by quarter," P produces a visualization specification e like { *"mark": "bar", "encoding":* { *"x": "quarter", "y": "sales"* } }. The engine E then renders the specified bar chart, providing the visual output r.

2.3 Framework

The natural language interfaces for tabular data querying and visualization encompass a variety of components, each playing a crucial role in the technology framework, as shown in Fig 3.

• *Datasets.* Datasets play a vital role in training and evaluating the performance of these interfaces. Datasets can be single-turn, where a single query is posed without any prior context, or multi-turn, where a series of queries are posed in a conversational manner. There are also various types of datasets designed to evaluate different aspects of the systems, such as their ability to handle complex queries, out-of-domain queries, and more.

• *Approaches.* The approaches to building natural language interfaces have evolved over time. Early approaches were rule-based, using pre-defined rules to translate natural language queries into functional representations. With the advent of neural networks, sequence-to-sequence models became popular, providing more flexibility in handling diverse queries. The rise of pre-trained language models, such as BERT [16] and GPT [79], marked a significant advancement in this field. Recently, the advent of LLMs like ChatGPT, and the exploration of prompt engineering techniques, have opened new avenues for the development of more effective natural language interfaces for data interaction.

• *Evaluation Metrics.* Evaluation metrics are used to measure the performance of these interfaces. These can be string-based, comparing the generated functional representation to a ground truth, or execution-based, comparing the result of executing the generated representation on the database to the expected result. Manual evaluation is also sometimes used to assess aspects like the system's usability.

• *System Design.* System architecture is a crucial component of natural language interfaces which involves the underlying mechanisms that translate user queries into actionable outputs. The architectural paradigms, ranging from rule-based to end-to-end designs, provide varied solutions and trade-offs in terms of flexibility, interpretability, and accuracy.

Each of these components contributes to the effectiveness and usability of natural language interfaces for tabular data querying and visualization. The subsequent sections of this survey will delve into these components in more detail, discussing their role, the various methods and technologies used, and the recent advancements in each area.

3 Datasets

3.1 Text-to-SQL Datasets

3.1.1 Existing Benchmarks

Text-to-SQL datasets have evolved significantly over time, adapting to the growing complexity of the field. Early datasets were single-domain, focusing on simple, context-specific queries. As the field progressed, cross-domain datasets emerged, featuring diverse schemas and queries across multiple domains. The introduction of multi-turn conversational datasets added another layer of complexity, requiring the understanding of inter-query dependencies within a conversation. The most recent advancement is the emergence of multilingual datasets, which extend the challenge to handling queries in multiple languages. Researchers are also exploring complex scenarios such as ambiguous queries, queries requiring external knowledge, and queries involving temporal and spatial reasoning. This evolution reflects the progress and the expanding challenges in the Text-to-SQL domain. Table 1 presents a comprehensive overview of various Text-to-SQL and Text-to-Vis datasets.

Single Domain. The early phase of Text-to-SQL research was marked by single-domain datasets, which focused on handling queries within a specific context. ***Academic*** [57] and ***Advising*** [25] are examples of early single-domain datasets. The ***ATIS*** dataset [13], [83] and ***GeoQuery*** [129] are notable for their focus on flight information and U.S. geography respectively. Datasets like ***Yelp*** and ***IMDB*** [118], ***Scholar*** [48], and ***Restaurants*** [108] were also developed around this time, each catering to queries pertaining to their respective domains. In recent years, the development of single-domain datasets has continued with the introduction of ***SEDE*** [39] and ***MIMICSQL*** [112]. These datasets represent the ongoing efforts to explore and address more complex and diverse queries within specific domains.

Cross Domain. Following the single-domain datasets, the focus shifted to cross-domain datasets, which widened the scope of the Text-to-SQL task by including queries from multiple domains. A pivotal dataset marking this shift is ***WikiSQL*** [133]. It offers a rich collection of 80,654 natural language inquiries paired with SQL queries. These pairs correspond to SQL tables extracted from a vast set of 26,521 Wikipedia tables. The dataset's uniqueness lies in its extensive coverage of tables and its capacity to challenge models to adapt to novel queries and table schemas. Another monumental contribution to this arena is the ***Spider*** dataset [127]. This dataset encompasses 10,181 natural language questions from 138 varied domains. Its diversity and inclusion of intricate queries make it a tougher challenge compared to its predecessors.

The Spider dataset has inspired the creation of several variants, each designed to test specific capabilities of Text-to-SQL models. For instance, ***Spider-SYN*** [27] tweaks the original Spider questions by substituting schema-related terms with their synonyms, elevating the schema linking challenge. ***Spider-DK*** [28] infuses domain-specific knowledge into questions, probing models' domain knowledge comprehension. Variants like ***Spider-CG*** [26] and ***Spider-SSP*** [92] focus on models' generalization abilities through diverse strategies, such as sub-sentence substitutions and compositional generalization, respectively. ***Dr. Spider*** [8] serves as a diagnostic tool, introducing variations in the original Spider dataset across multiple dimensions. Lastly, ***Spider-realistic*** [15] enhances task complexity by removing direct column name mentions from questions, demanding an improved robustness from models.

Multi-turn. As the field of Text-to-SQL expanded to encompass more complex interactions, the need for datasets that could simulate multi-turn conversations became apparent. To cater to this, various datasets emphasizing context-driven Text-to-SQL interactions were developed. ***SParC*** [128] is a prominent cross-domain dataset boasting approximately 4.3k sequences of questions, which

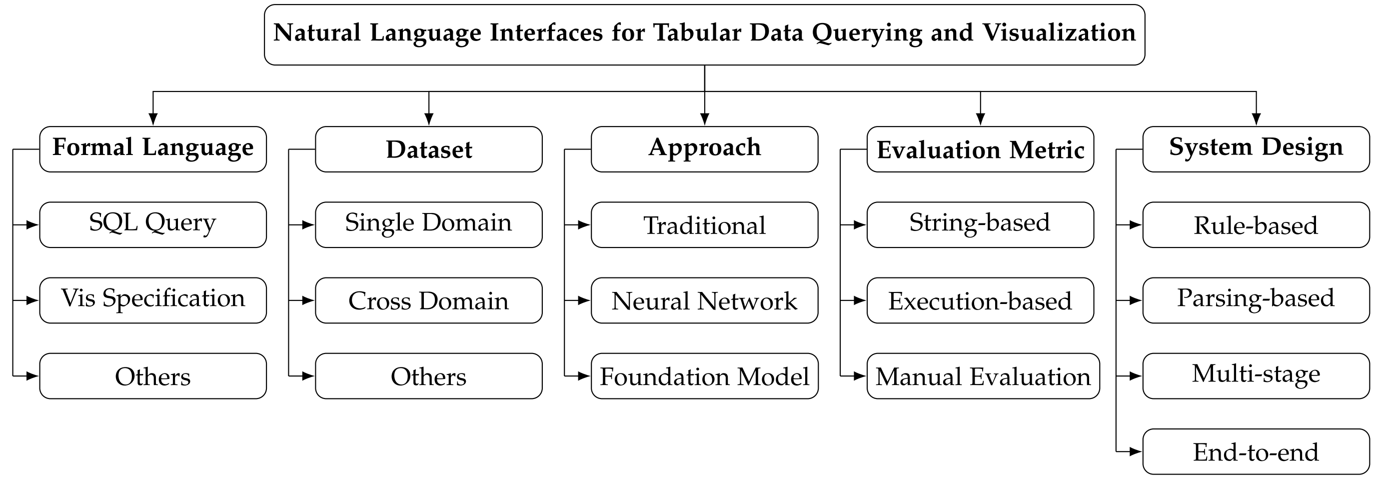

Fig. 3. Framework for Natural Language Interfaces in Tabular Data Querying and Visualization. The technology framework comprises various pivotal components: functional representation, dataset, approach, evaluation metric, and system design.

cumulatively constitute over 12k question-SQL pairings. What's unique about SParC is that each of its question sequences evolves from an original question in Spider, with subsequent questions intricately woven in. Similarly, the ***CoSQL*** dataset [126], established under the Wizard-of-Oz framework, stands out as the first large-scale, cross-domain conversational Text-to-SQL collection. It houses nearly 3k dialogues, translating to over 30k dialogue turns and 10k associated SQL queries. Through these dialogues, the dataset replicates a scenario where annotators, posing as database users, utilize natural language to extract database responses. Another noteworthy contribution is the ***CHASE*** dataset [37]. This dataset introduces a large-scale, context-sensitive Chinese Text-to-SQL collection, featuring 5,459 interconnected question sequences and 17,940 individual questions paired with SQL queries. Collectively, these datasets push the boundaries in the Text-to-SQL domain, emphasizing more fluid, dialogue-centric database interactions and offering diverse challenges for research exploration.

Multilingual. As the Text-to-SQL field expands globally, the need for multilingual datasets has become increasingly apparent. Several datasets have been developed to address this need, offering benchmarks in different languages and thereby broadening the scope of Text-to-SQL research. ***CSpider*** [74], ***TableQA*** [104] and ***DuSQL*** [111] extend the Text-to-SQL task to Chinese, introducing a new linguistic challenge. ***ViText2SQL*** [78] broadens the field further with a Vietnamese Text-to-SQL dataset, pushing models to handle the complexities of the Vietnamese language. Similarly, ***PortugueseSpider*** [49] extends the task to Portuguese, requiring models to translate Portuguese queries into SQL. These multilingual datasets represent a significant stride towards developing Text-to-SQL systems that can cater to a global, multilingual user base, thereby democratizing access to data across linguistic boundaries.

Knowledge Grounding. Recent advancements in Text-to-SQL research have seen a growing emphasis on knowledge-intensive benchmarks, reflecting the need for models that can handle real-world analysis scenarios. Such benchmarks, like ***Spider-dk*** [28], extends the Spider dataset to focus more on domain knowledge, reflecting the need for models to understand and incorporate domain-specific knowledge in their translations. Another datset, ***knowSQL*** [23], prioritize knowledge grounding or commonsense reasoning, helping experts make informed decisions. A most recent benchmark is ***BIRD*** [60], specifically tailored for expansive database-anchored Text-to-SQL tasks. What sets BIRD apart is its emphasis on the values within databases. It underscores novel hurdles, such as inconsistencies in database content, the imperative of bridging external knowledge with natural language queries and database content, as well as the efficiency of SQL, particularly when dealing with vast databases. These knowledge-intensive datasets represent a significant stride towards developing Text-to-SQL systems that can handle complex, real-world scenarios, bridging the gap between academic study and practical application.

3.1.2 Auxiliary Annotations

Auxiliary annotations in Text-to-SQL datasets provide additional information that assists models in understanding and translating natural language queries into SQL. These annotations, which include but are not limited to schema linking, context dependency, and query difficulty level, encapsulate insights vital for understanding the task. By incorporating these and other auxiliary annotations, datasets not only present a more diverse array of challenges but also offer models a richer training context, urging them to attain a more profound grasp of the task.

Schema Linking. Schema linking annotations provide explicit mappings between elements in the natural language query and entities in the database schema [37]. They essentially link the ’concepts’ in the query to the corresponding columns or tables in the database. For example, the phrase ”flights from New York” might be linked to the ’flights’ table and the ’departure_city’ column. These annotations are crucial for models to understand the semantics of the query in the context of the specific database schema, enabling them to generate accurate SQL

TABLE 1
Statistics for Text-to-SQL and Text-to-Vis datasets

Datasets	#Query	#Database	#Domain	#T/DB	Language	Main Features
Text-to-SQL Datasets						
Academic (Li and Jagadish, 2014)	196	1	1	15	English	Single Domain
ATIS (Hemphill et al., 1990; Dahl et al., 1994)	5,280	1	1	32	English	
GeoQuery (Zelle and Mooney, 1996)	877	1	1	6	English	
Scholar (Iyer et al., 2017)	817	1	1	7	English	
Restaurants (Tang and Mooney, 2000)	378	1	1	3	English	
IMDB (Yaghmazadeh et al., 2017)	131	1	1	16	English	
Yelp (Yaghmazadeh et al., 2017)	128	1	1	7	English	
Advising (Finegan-Dollak et al., 2018)	3,898	1	1	10	English	
MIMICSQL (Wang et al., 2020)	10,000	1	1	5	English	
SEDE (Hazoom et al., 2021)	12,023	1	1	29	English	
Spider (Yu et al., 2018)	10,181	200	138	5	English	Cross Domain
WikiSQL (Zhong et al., 2017)	80,654	26,521	-	1	English	
Squall (Shi et al., 2020)	11,468	1,679	-	1	English	
KaggleDBQA (Lee et al., 2021)	272	8	8	2	English	
SParC (Yu et al., 2019)	12,726	200	138	5.1	English	Multi-turn
CoSQL (Yu et al., 2019)	15,598	200	138	5.1	English	
CHASE (Guo et al., 2021)	17,940	280	-	4.6	Chinese	
Spider-SYN (Gan et al., 2021)	7,990	166	-	5	English	Robustness
Spider-CG (Gan et al., 2022)	45,599	-	-	5	English	
Spider-SSP (Shaw et al., 2021)	3,282	-	-	5	English	
Spider-realistic (Deng et al., 2021)	508	-	-	5	English	
Dr. Spider (Chang et al., 2023)	-	166	-	5	English	
CSpider (Min et al., 2019)	10,181	200	138	5	Chinese	Multilingual
DuSQL (Wang et al., 2020)	23,797	200	-	4.1	Chinese	
TableQA (Sun et al., 2020)	64,891	6,029	-	1	Chinese	
ViText2SQL (Nguyen et al., 2020)	9,691	166	-	5	Vietnamese	
PortugueseSpider (Archanjo Jose et al., 2021)	9,691	166	-	5	Portuguese	
PAUQ (Bakshandaeva et al., 2022)	9,691	166	-	5	Russian	
Spider-DK (Gan et al., 2021)	535	10	-	5	English	Knowledge Grounding
knowSQL (Dou et al., 2022)	25,888	200	3	-	Chinese	
BIRD (Li et al., 2023)	12,751	95	-	7	English	
Text-to-Vis Datasets						
Gao et al., 2015	10	3	-	1	English	Single Domain
Kumar et al., 2016	490	1	-	-	English	
Srinivasan et al., 2021	893	3	-	1	English	
nvBench (Luo et al., 2021)	25750	153	105	5	English	Cross Domain
ChartDialogs (Shao et al., 2020)	3284	-	-	-	English	Multi-turn
Dial-NVBench (Song et al., 2023)	4495	-	-	-	English	

queries.

Context Dependency. In multi-turn conversation datasets, context dependency annotations [37] provide information about the relationships between consecutive queries. They indicate whether the interpretation of a query depends on previous queries or responses in the conversation. For example, in a conversation like ”Show me flights from New York.” - ”What about to Chicago?”, the second query is context-dependent, as its meaning relies on the first query. The dependency types can be ”Coreference”, which involves referring back to an entity or concept from a previous query without directly stating it, or ”Ellipsis”, where certain parts of a query are omitted because they can be inferred from the previous context.

Difficulty Level. Difficulty level [127] annotations categorize queries based on their complexity. They provide a measure of the sophistication required to translate the natural language query into SQL. Factors that might influence the difficulty level include the complexity of the SQL query (e.g., the number of tables or joins involved, the presence of subqueries, etc.), the complexity of the natural language query (e.g., the length of the query, the complexity of the sentence structure, etc.), and the degree of schema understanding required. These annotations enable a more nuanced evaluation of model performance, highlighting their ability to handle varied queries.

3.2 Text-to-Vis Datasets

3.2.1 *Existing Benchmarks*

Text-to-Vis datasets generally follow the same format as Text-to-SQL datasets with a set of tabular data and (NLQ, Vis) pairs for each database. The progression was similarly a transition from single to cross-domain datasets though several Text-to-Vis datasets piggybacked Text-to-SQL benchmarks. Currently, a few public datasets are available for the Text-to-Vis task, as shown in Table. 1.

Single Domain. During early development stages of Text-to-Vis interfaces, datasets are generally used as a proof of concept. These datasets are each concentrated on one domain that only involves queries within a set range of context. The dataset by Gao et al [55] was developed by asking test subjects to pose natural utterances by looking at several human-generated visualizations with the goal to gain certain information. The dataset by Kumar et al. [55] focused on crime data and the queries are to gain insight to police force allocations.

Cross Domain. As the field progressed, there was a need for cross-domain datasets with natural language queries of several different concepts. Srinivasan et al. [100] collected queries across 3 datasets. They provided thorough analyses on the classification of Text-to-Vis natural language queries. ***nvBench*** [69] is the largest and most used Text-to-Vis benchmark, containing 25,750 natural language and visualization pairs from 750 tables over 105 domains. It is synthesized from Text-to-SQL benchmark Spider [127] to support cross-domain Text-to-Vis task.

Multi-turn. Due to the large amount of information needed to produce accurate visualizations, it is apparent that not all information may be provided in just one round of natural language query. To tackle this issue, multi-turn datasets have been introduced to make several rounds of modifications on the output visualization. ***ChartDialogs*** [91] contains 3,284 dialogues and is curated for plotting using matplotlib. Building on the cross-domain dataset nvBench, ***Dial-NVBench*** [97] was created to target dialogue inputs. The dataset contains 4,495 dialogue sessions and each is aimed to contain enough information so the system can output a suitable visualization.

4 APPROACHES

4.1 Text-to-SQL Parsing

The approaches to the Text-to-SQL task have evolved significantly over time, mirroring the broader developments in natural language processing, as illustrated in the timeline in Fig 4. Early efforts were focused on rule-based approaches, where queries were translated into SQL based on a predefined set of rules and patterns. The emergence of neural networks and the sequence-to-sequence paradigm marked a turning point in Text-to-SQL research. Neural network approaches, which translate a source sequence (the natural language query) into a target sequence (the SQL query), showed a greater capacity to handle the intricacies of natural language and the diversity of SQL queries. In recent years, the advent of foundation language models like BERT [16] and GPT [79] has opened up new possibilities for the Text-to-SQL task. This evolution in approaches reflects the ongoing efforts to develop models that can accurately and efficiently translate natural language queries into SQL, handling the challenges presented by the variability of natural language and the complexity of SQL. Table 2 provides a comparative analysis of notable approaches in the Text-to-SQL and Text-to-Vis domains.

4.1.1 Traditional Stage

Text-to-SQL research began with rule-based approaches, which were the primary method of handling this task for several decades. Surveys like [54], [81] have presented the work of this stage in more detailed ways. Early rule-based methods like ***TEAM*** [32] and ***CHAT-80*** [113] used intermediate logical representations, translating natural language queries into logical queries that were independent of the database schema, and then converting these logical queries into SQL. However, these methods relied heavily on handcrafted mapping rules.

In the early 2000s, more advanced rule-based methods were developed. ***PRECISE*** [80] utilized an off-the-shelf natural language parser to translate queries, but its coverage was limited due to the assumption of a one-to-one correspondence between words in the query and database elements. To address this, methods like ***NaLIR*** [57], ***ATHENA*** [88], and ***SQLizer*** [118] adopted a ranking-based approach, finding multiple candidate mappings and ranking them based on a score. NaLIR further improved performance by involving user interaction, while ATHENA leveraged a domain-specific ontology for richer semantic information. SQLizer used an iterative process to refine the logical form of the query. ***Templar*** [3] offered an optimization technique for mapping and joint path generation using a query log. Despite their significant improvements, these methods still relied on manually-defined rules, which limit their ability to handle many variations in natural language.

4.1.2 Neural Network Stage

The advent of neural networks and the sequence-to-sequence (Seq2Seq) paradigm marked a turning point in the field of Text-to-SQL. Originally for machine translation, Seq2Seq models can learn intricate data mappings, accommodating diverse queries and complex SQL structures. Such a model typically uses an encoder to process the natural language query and a decoder to generate the corresponding SQL query. For a deeper dive into neural network-based approaches, readers are encouraged to consult prior surveys like [14], [52], [53].

Encoder. Encoders in the Text-to-SQL context determine how the natural language query and the database schema are jointly transformed into a continuous representation that the model can work with. They can broadly be classified into two categories: sequence-based encoders and graph-based encoders.

- *Sequence-based Encoder.* Sequence-based encoders form the foundation of many Text-to-SQL systems. They are often based on Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTM) networks, Gated Recurrent Units (GRUs), or Transformer architectures.

Bi-directional Long Short-Term Memory (bi-LSTM) based models have been widely used in early Text-to-SQL systems due to their capability to capture dependencies

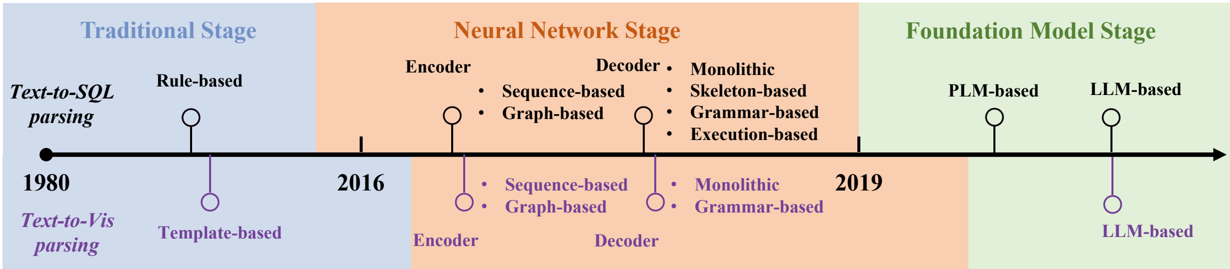

Fig. 4. Evolution of Text-to-SQL and Text-to-Vis approaches over time. We briefly divide the timeline into traditional, neural network, and foundation language model stages.

in both directions of a sequence. Notable work includes ***TypeSQL*** [123], which assigns a type to each word in the question, with a word being an entity from the knowledge graph, a column, or a number. The model then concatenates word embeddings and the corresponding type embeddings as input to the bi-LSTM, which helps it better encode keywords in questions. ***Seq2SQL*** [133], ***SQLNet*** [117], and ***IncSQL*** [95] employ a bi-LSTM to produce a hidden state representation for each word in the natural language query. For column headers, a bi-LSTM is also used for each column name, with the final hidden state used as the initial representation for the column. For example, ***EditSQL*** [131] also utilizes two separate Bi-LSTMs for encoding the natural language questions and the table schema, and then applies a dot-product attention layer to integrate the two encodings.

With the advent of the Transformer architecture, self-attention models have gained popularity in the Text-to-SQL task. The original self-attention mechanism is the building block of the Transformer structure, and models like those developed by He et al. [40], Hwang et al. [46], and Xie et al. [116], have incorporated this mechanism. These models leverage the Transformer's ability to capture dependencies regardless of their distance in the sequence, which is especially useful for handling complex, non-local dependencies often present in the Text-to-SQL task.

• *Graph-based Encoder.* Graphs are an effective way to capture complex structures, making them particularly suitable for encoding database (DB) schemas, which are rich in structural information. Bogin et al. [4] were pioneers in using graph representations for DB schemas. They employed nodes for tables and columns and edges to depict table-column relationships, such as table compositions, and primary and foreign key constraints. These graph structures were then encoded using graph neural networks (GNNs). In a follow-up study, Bogin et al. introduced ***Global-GNN*** [5], emphasizing global reasoning to encode the schema, integrating question token representations between question terms and schema entities. ***RAT-SQL*** [109] combined global reasoning, structured reasoning, and relation-aware self-attention for schema entities and question terms.

Graphs have also been employed to concurrently encode questions and DB schemas. Cao et al. put forward the ***LGESQL*** [7] model to unearth multi-hop relational attributes and significant meta-paths. ***S^2SQL*** [45] explored question token syntax's role in Text-to-SQL encoders and introduced a versatile and resilient injection technique. To strengthen graph method's generalization for unfamiliar domains, ***SADGA*** [6] crafted both question and schema graphs based on the dependency structure of natural language queries and schema layout, respectively. ***ShawdowGNN*** [11] countered domain information influence by disregarding table or column names and employing abstract schemas for delexicalized representations. Lastly, Hui et al. 2021 [44] designed a dynamic graph framework to capture interactions among utterances, tokens, and database schemas, leveraging both context-independent and dependent parsing.

Decoder. The decoder is a crucial component of the sequence-to-sequence paradigm, responsible for generating the SQL query from the encoded representation of the natural language query and database schema. Broadly, these decoders can be classified into four categories: monolithic decoders, skeleton-based decoders, grammar-based decoders, and execution-guided decoders.

• *Monolithic Decoder.* The Monolithic decoder, influenced by advancements in machine translation, primarily utilizes RNNs for the sequential generation of SQL commands. Early implementations of this method relied on RNNs to compute the probability of each SQL token, considering both the prior context and previously generated tokens [48]. The context from the input is encoded, often using mechanisms like soft-attention, which emphasizes the most pertinent input components for each token generation. As for representing previously generated tokens, a common method is to use hidden state from the prior decoder step.

• *Skeleton-based Decoder.* Skeleton-based decoders tackle the Text-to-SQL problem by first generating a template or skeleton of the SQL query, which is then populated with specific details from the input. This approach can help manage the complexity of SQL queries by breaking down the generation process into more manageable steps. For example, ***SQLNet*** [117] introduced the approach that focuses on filling in the slots in a SQL sketch, aligning with the SQL grammar, rather than predicting both the output grammar and the content. This approach captures the dependency of the predictions, where the prediction of one slot is conditioned only on the slots it depends on. ***HydraNet*** [72] uses a multi-headed selection network for simultaneous generation of different parts of the SQL query. ***IE-SQL*** [40] and ***TypeSQL*** [123] also use a slot-filling approach, where a pre-defined SQL template is filled in based on the input. ***COARSE2FINE*** [20] and IRNet [38] adopt a two-step coarse-to-fine generation process, where an initial rough sketch is

TABLE 2
Comparative Analysis of Representative Approaches in Text-to-SQL and Text-to-Vis

Approach		SQL Example	Vis Example	Advantages	Disadvantages
Traditional Stage					
Rule-based		SQLizer	-	Strong interpretability	Limited adaptability and complexity
Template-based		TEAM	DataTone	Fast, good for well-defined scenarios	Struggles with novel queries
Neural Network Stage					
Encoder	Sequence	Seq2SQL	ncNet	Flexible to diverse inputs	Requires substantial training data
	Graph	LGESQL	RGVisNet	Captures relational data structures	Complexity increases with graph size
Decoder	Monolithic	Seq2Tree	Seq2Vis	Unified architecture	May not capture nuances
	Skeleton	SQLNet	-	Better at capturing SQL structure	Limited generalization
	Grammar	PICARD	RGVisNet	Aligns with formal language structure	Limited complexity
	Execution	SQLova	-	Validates through actual execution	Might be slower in real-time
Foundation Language Model Stage					
PLM-based		RESDSQL	-	Strong performance	Hard to interpret and constrain
LLM-based		DIN-SQL	Chat2Vis	Strong performance and generalization	Requires prompt engineering

generated and subsequently refined with low-level details conditioned on the question and the sketch. ***RYANSQL*** [12] takes a recursive approach to yield SELECT statements and employs a sketch-based slot filling for each of the SELECT statements. This approach effectively handles complex SQL queries with nested structures.

• *Grammar-based Decoder.* Grammar-based decoders generate the SQL query directly from the encoded representation of the input, often utilizing SQL grammar rules, intermediate representations, or incorporating constraints in the decoding process to ensure the generation of valid SQL queries.

Decoders utilizing rules aim to reduce the chances of generating out-of-place tokens or syntactically incorrect queries. By generating a sequence of grammar rules instead of simple tokens, these models ensure the syntactical correctness of the generated SQL queries. For example, ***Seq2Tree*** [19] employs a top-down decoding strategy, generating logical forms that respect the hierarchical structure of SQL syntax. ***Seq2AST*** [119] takes this idea further by using an abstract syntax tree (AST) for decoding. ***SyntaxSQLNet*** [125] adapts this approach to SQL-specific syntax. It employs a tree-based decoder that recursively calls modules to predict different SQL components, providing a structured approach to SQL query generation. ***SmBoP*** [87] stands out for its bottom-up decoding mechanism. Given a set of trees, it scores and selects trees based on SQL grammar, ensuring that the generated queries are both syntactically valid and semantically aligned with the input. ***Bridge*** [61] uses an LSTM-based pointer-generator with multi-head attention and a copy mechanism as the decoder. This model is capable of generating a token from the vocabulary, copying a token from the question, or copying a schema component from the database schema at each decoding step, providing a flexible approach to SQL query generation.

Some other decoders generate an intermediate representation (IR) of the SQL query first, simplifying the SQL generation task by breaking it down into more manageable steps. Typical models include ***IncSQL*** [95] which defines distinct actions for different SQL components and lets the decoder predict these actions instead of directly generating SQL queries, effectively simplifying the generation task. ***IRNet*** [38] introduces SemQL, an intermediate representation for SQL queries. SemQL can cover a wide range of SQL queries, making it a versatile tool for SQL query generation. ***NatSQL*** [29] builds on the idea of SemQL by removing set operators, streamlining the IR and making it easier to handle.

There are also constrained decoding-based decoders which incorporate constraints into the decoding process to guide the SQL query generation. Models like ***PICARD*** [89] and ***UniSAr*** [24] use reinforcement learning and rule-based systems, respectively, to incorporate constraints into the decoding process. These constraints guide the model towards generating valid SQL queries, contributing to the accuracy and reliability of these models.

• *Execution-based Decoder.* Execution-based decoders offer a unique approach to the Text-to-SQL task, utilizing an off-the-shelf SQL executor such as SQLite to verify the validity and correctness of the generated SQL queries during the decoding process. This methodology ensures both syntactic and semantic accuracy of the produced SQL queries. Wang et al. [110] leverages a SQL executor to check the partially generated SQL queries during the decoding process. Queries that raise errors are discarded, and the model continues to refine the generation until a valid SQL query is produced. Suhr et al. [103] follow a similar approach, but they avoid altering the decoder's structure. Instead, they examine the executability of each candidate SQL query. Only the queries that can be successfully executed are considered valid, which helps in maintaining the grammatical correctness of the generated SQL queries. In another approach, ***SQLova*** [46] incorporate an execution-guided decoding mechanism that filters out non-executable partial SQL queries from the output candidates. This methodology ensures the generation of SQL queries that are not only syntactically correct but also executable.

4.1.3 Foundation Language Model Stage

The recent upsurge in the performance of NLP tasks is significantly attributed to the advancement of foundation language models (FMs) such as BERT, T5, and GPT. These models, trained on large corpora, capture rich semantic and syntactic features of languages and have been successful across a variety of tasks.

We categorize the FM-based approaches in Text-to-SQL into two categories based on the language models they incorporate: Pretrained Language Models(PLMs) and Large Language Models(LLMs). PLMs, representing the earlier evolution like BERT and initial GPT versions, capture detailed linguistic nuances through extensive training. They are often refined for specific tasks via methods like fine-tuning. LLMs represent an advancement, characterized by their vast scale. By amplifying model parameters or training data, these models exhibit enhanced "emergent abilities" [114]. A prime example is ChatGPT, an adaptation of the GPT architecture that excels in dialogue interactions. LLM-based Text-to-SQL methods leverage prompts, utilizing in-context learning [21] and chain-of-thought [115] reasoning to produce apt SQL queries.

PLM-based. Early PLM-based approaches directly utilize and fine-tune pre-trained language models, refining them specifically for the Text-to-SQL task. These models can be broadly categorized into encoder-only language models and encoder-decoder language models.

- *Encoder-only Language Models.* Models like BERT and RoBERTa [66] serve as foundational encoder-only PLMs in various Text-to-SQL models, transforming input sequences into context-sensitive numerical representations. ***IRNet*** [38], for example, harnesses BERT to craft a specialized input sequence. ***BRIDGE*** [61] fuses BERT's prowess with schema-consistency guided decoding in a seq-to-seq architecture, enhancing the schema linking ability. ***HydraNet*** [72] and ***SQLova*** [46] process questions and columns separately, predicting for each column individually with BERT, notably excelling on the WikiSQL benchmark. ***X-SQL*** [40] makes a novel modification to BERT by replacing segment embeddings with column type embeddings. This model also encodes additional feature vectors for matching question tokens with table cells and column names, and concatenates them with BERT embeddings of questions and database schemas.

- *Encoder-decoder Language Models.* Unlike encoder-only models, encoder-decoder models like T5 [85] and BART [56] are end-to-end models designed for seq-to-seq tasks. These models take a sequence of textual input and generate a sequence of textual output. They have been adapted and fine-tuned for the Text-to-SQL task, resulting in innovative and effective models. ***UnifiedSKG*** [116], for example, finetunes T5 on Text-to-SQL task with PICARD [89] decoding. By combining the advantages of T5's powerful language understanding capabilities with the benefits of a sketch-based approach, it captures both the structural aspects of SQL and the semantic nuances of natural language questions. ***Graphix-T5*** [59] leverages the robust contextual encoding intrinsic of T5 to enhance domain generalization by modeling relational structures. Its GRAPHIX layer encodes both semantic and structural insights, marking a pioneering step in infusing graphs into Text-to-SQL translation. ***RESD-SQL*** [58] also taps into the T5 model to craft the SQL query, utilizing a fusion of the question and schema sequences, where various T5 variants are adapted to generate skeletons derived from questions.

- *Additional Pretraining.* Apart from finetuning from general pretrained language models, there are some approaches involving additional pretraining of language models with Text-to-SQL data. Rather than directly employing off-the-shelf PLMs, these methods construct a new model using architectures like BERT or BART, and train these models using Text-to-SQL data (tabular data and text-to-SQL pairs) with specially designed objectives that are related to SQL generation.

For instance, ***TaBERT*** [120] enhances BERT by training on tabular data, focusing on predicting concealed column names and restoring cell values. This equips the model with insights into database tables' structure and content, which is crucial for accurate SQL query generation. ***Grappa*** [124] finetunes BERT by generating question-SQL pairs over tables. The training targets objectives like masked language modeling (MLM), column prediction, and SQL operation prediction, honing the model's ability to produce SQL queries aligned with the natural language intent. ***GAP*** [94] follows a parallel strategy, pretraining BART on combined Text-to-SQL and tabular datasets. The training focuses on objectives like MLM, predicting columns, restoring columns, and crafting SQL. Integrating these goals, GAP ensures that the model comprehends subtle differences in the database tables and the posed questions, improving the precision of generated SQL queries.

LLM-based. LLM-based methods mark the latest trend in Text-to-SQL, combining the power of large language models with the art of prompt engineering. These approaches use carefully designed prompts to steer the models towards generating accurate SQL queries, with two main categories: zero-shot prompting and few-shot prompting.

- *Zero-shot Prompting.* In zero-shot prompting, the LLM receives a specific prompt without any additional training examples, banking on the extensive knowledge it has gained during the pre-training phase. Rajkumar et al. [86] first embarked on an empirical exploration of zero-shot Text-to-SQL capabilities on Codex [9]. After ChatGPT came out, Liu et al. [63] conducted an extensive evaluation of zero-shot Text-to-SQL ability on it across an array of benchmark datasets. Building on this, the method ***C3*** [22] based on ChatGPT emerged as a leading zero-shot Text-to-SQL solution on the Spider Challenge. The essence of C3 lies in its three foundational prompting components: Clear Prompting, Calibration with Hints, and Consistent Output. ***ZERoNL2SQL*** [34] has merged the strengths of both PLMs and LLMs to foster zero-shot Text-to-SQL capabilities. The approach leverages PLMs for the generation of an SQL sketch through schema alignment and subsequently employs LLMs to infuse the missing details via intricate reasoning. A distinctive feature of their method is the predicate calibration, designed to align the generated SQL queries closely with specific database instances.

- *Few-shot Prompting.* Few-shot prompting in Text-to-SQL presents a fascinating landscape where models are guided to achieve complex tasks with minimal examples. The strategies of in-context learning (ICL) and chain-of-thought (CoT) reasoning play pivotal roles in these approaches, enabling models to extract knowledge from a handful of demonstrations and reason through intricate SQL generation processes.

A notable work in this area is ***DIN-SQL*** [82], which showcases how breaking down SQL query generation into constituent problems can significantly improve the performance of LLMs. This is achieved through a four-module strategy: schema linking, query classification and decomposition, SQL generation, and a novel self-correction mechanism. Similary, Liu et al. [65] brought forth the *Divide-and-Prompt* paradigm which decomposes the primary task into simpler sub-tasks, tackling each through a CoT approach, thereby enhancing the reasoning abilities of LLMs for the Text-to-SQL task. Gu et al. [33] presented a unique Divide-and-Conquer framework which steers LLMs to generate structured SQL queries and subsequently populates them with concrete values, ensuring both validity and accuracy.

In a comprehensive study, Nan et al. [76] explored various prompt design strategies to enhance Text-to-SQL models. The research probes into different demonstration selection methods and optimal instruction formats, revealing that a balance between diversity and similarity in demonstration selection combined with database-related knowledge augmentations can lead to superior outcomes. Tai et al. [106] proposed a systematic investigation of enhancing LLM's reasoning abilities for text-to-SQL parsing through various chain-of-thought style promptings. The research found that avoiding excessive detail in reasoning steps and improving multi-step reasoning can lead to superior results.

More recently, ***SQL-PaLM*** [105], an LLM-based approach grounded in PaLM-2, is proposed employing an execution-based self-consistency prompting approach tailored for Text-to-SQL. Guo et al. [35] proposed a retrieval-augmented prompting method that integrates sample-aware demonstrations and a dynamic revision chain. This approach aims to generate executable and accurate SQLs by iteratively adapting feedback from previously generated SQL, ensuring accuracy without human intervention.

4.2 Text-to-Vis Parsing

Currently, there are several models specifically handling the Text-to-Vis problem. They typically accept a natural language query and tabular data, producing a self-defined visual language query (VQL), a SQL-like pseudo syntax for combining database querying with visualization directives, which is then hard-coded to visual specification code. Similar to Text-to-SQL, Text-to-Vis parsing approaches have transitioned through three evolutionary stages: traditional, neural network, and foundation language model, as illustrated in Fig. 4.

4.2.1 Traditional Stage

During this stage, the main focus was to improve accuracy by using different parsing methods, key words, and grammar rules. Between 2015 and 2020, the works mostly explored the effects of different semantic parsing techniques. Notable works include ***DataTone*** [30], ***Eviza*** [90], ***Evizeon*** [41], ***VisFlow*** [121], ***FlowSense*** [122], ***Orko*** [101], ***Valletto*** [51], and ***InChorus*** [99]. The survey by Shen et al. [93] gave a thorough walk-through of the different methods. Stemming from the method in DataTone, several works in 2020 and 2021 deployed a more structured VQL template. The VQLs for each system were defined slightly differently but they generally follow the SQL style and include additional visualization attributes. ***ADVISor*** [64] developed an automatic pipeline to generate visualization with annotations. The input is a set of table headers and a NLQ and the output is a set of aggregations in a SQL-like format. ***NL4DV*** [77] was a python package that takes an NLQ and the associated tabular dataset as input and outputs visualization recommendations in the form of a JSON object that can help users generate visualizations.

4.2.2 Neural Network Stage

The emergences of deep neural networks, especially attention mechanisms, brought a shift towards encoder-decoder-based models. As discussed earlier, the template approach can be easily converted to a neural network model. In some models, visualization specifications are directly produced, bypassing the intermediate VQL sequence step. This section delves into various models leveraging the encoder-decoder architecture.

Encoder. Sequence-based encoders like LSTMs and transformers excel at managing sequential data's long-term dependencies, while graph-based encoders grasp non-linear relationships, comprehensively depicting the input. Their capability to represent complex data structures establish their significance in crafting efficient Text-to-Vis systems.

• *Sequence-based Encoder.* Sequence-based encoders like LSTM, attention mechanisms, and transformers have become essential to Text-to-Vis. While LSTMs are great at managing sequential long-term dependencies, they are restricted in modeling complex interactions between distant words. This limitation is addressed by the attention mechanism and is further enhanced by the Transformer architecture.

Seq2Vis [70], evolving from ***Data2Vis*** [17], employs a seq2seq model, enhancing it with pre-trained global word embeddings for richer input understanding. Combined with LSTM encoders, attention, and LSTM decoders, Seq2Vis adeptly translates natural language queries into visualizations. Similarly, MMCoVisNet [97] leverages an LSTM-based encoder for text-to-Vis dialogues. Conversely, ***ncNet*** [71] transitions to a Transformer-based model. Its multi-self-attention design eliminates recurrent computations, heightening efficiency. In ncNet, tokenized inputs from three sources are sequenced and merged. Each word is tokenized, masked tokens are populated, and boundary-indicating tokens are added. These tokens undergo vectorization using various embeddings, establishing ncNet as a state-of-the-art in Text-to-Vis, proficiently converting queries into visualization codes.

• *Graph-based Encoder.* As the field of Text-to-Vis progresses, there is a notable shift towards leveraging more complex and efficient encoding methods for input data. Unlike sequence-based methods that process input data in a linear manner, graph-based encoders can capture non-linear relationships within the data, thus offering a richer and more contextually accurate representation of the input.

A notable work in this direction is ***RGVisNet*** [98]. It merges sequence and graph-based encoding in a novel retrieval-generation approach. The input natural language query(NLQ) is parsed to extract relevant VQL from its codebase, achieved by retrieving schemas in the NLQ, performing schema linking, and locating similar VQLs from the codebase. The NLQ is embedded through an LSTM encoder, while the candidate VQLs are processed through a Graph Neural Network (GNN) encoder using an abstract syntax tree (AST) representation. The relevance between NLQ and VQL embeddings is assessed using cosine similarity, with the embeddings then funneled into a Transformer encoder to ascertain relationships and yield the final output.

Decoder. Decoders in Text-to-Vis systems translate encoded textual input into coherent visualizations. Existing approaches have incorporated LSTM, transformer, and grammar-based decoders.

• *Monolithic Decoder.* In the context of Text-to-Vis tasks, monolithic decoders utilize a single, end-to-end model, often based on RNNs, LSTMs, or Transformer architectures, to transform a natural language description into a complete and coherent visual representation by sequentially generating components of a visualization, conditioned on an encoded representation of the input text.

Seq2Vis [70] uses an LSTM decoder within its architecture to generate visual queries. The attention mechanism it incorporates enables dynamic consideration of the input sequence's segments during output generation. Conversely, ***ncNet*** [71] employs a Transformer-based encoder-decoder approach. Both its encoder and decoder are built using self-attention blocks, optimizing inter-token relationship processing. This design provides flexibility in sequence translation, with the auto-regressive decoder ensuring coherent and logically sequenced outputs.

• *Grammar-based Decoder.* ***RGVisNet*** [98] introduces a grammar-aware decoder tailored for VQL revision. Given VQL's strict and defined grammar, similar to programming languages, leveraging this structure becomes essential. This approach mirrors text-to-SQL tasks, where integrating grammar as inherent knowledge effectively guides code generation. RGVisNet adapts the SemSQL grammar to support DV queries. The core decoder in RGVisNet adapts an LSTM-based structure underpinned by the formation of a context-free grammar tree. As the model traverses this tree, it leverages an LSTM model at every step to opt for the most likely branch, based on prior routes.

4.2.3 Foundation Language Model Stage

Foundation language models (FMs), especially large language models such as CodeX [9] and GPT-3, have revolutionized natural language processing with their ability to generate contextually accurate text. This is leveraged to advance the field of Text-to-Vis towards a new set of approaches.

• *Zero-Shot Prompting.* Zero-shot prompts refers to the use of untrained prompts to guide LLMs in generating visualization codes straight from textual or spoken queries. Leveraging LLMs' natural language understanding capabilities, zero-shot prompting in text-to-visualization systems employs carefully crafted prompts as guiding instructions, steering the models to generate specific and contextually appropriate visualizations based on user input. Mitra et al. [75] developed a prototype web application by prompting CodeX. ***Chat2VIS*** [73] also chose the model CodeX and specifically included a code prompt component to guide the LLM. These two methods both output visualization specification code directly.

• *Few-Shot Prompting.* Few-shot methods employ limited examples to guide LLMs towards desired outputs. ***NL2INTERFACE*** [10] utilizes CodeX by first preparing examples that translate natural language queries into a specific VQL format named SPS. This step forms a suitable prompt for in-context learning by CodeX. Subsequently, given the natural language queries and a database catalog, Codex predicts the corresponding VQL. Finally, NL2INTERFACE maps these SPS representations to generate interactive interfaces, following a procedure similar to PI2 based on a predefined and extensible cost model.

5 Evaluation Metrics

Evaluation metrics play a pivotal role in assessing the performance and robustness of semantic parsers for both Text-to-SQL and Text-to-Vis tasks. As the ultimate goal is to generate formal queries or visualization commands that accurately reflect the user's intent, the choice of evaluation metric is crucial to ensure the models make semantically and syntactically correct predictions. Typically, three types of metrics are used to assess the performance of these models: string-based matching, execution-based matching, and manual evaluation. A detailed comparison of these evaluation metrics can be found in Table 3.

String-based matching metrics evaluate the exact textual match between the generated output and the ground truth. They can be strict measures, ensuring both structural and semantic correctness, but might overlook minor variations that lead to equivalent outputs. Rather than relying on textual similarity, execution-based matching metrics assess the equivalence of outputs based on their execution results. They allow for flexibility in representation, focusing more on the functional correctness of the generated queries or visual commands. Lastly, manual evaluation involves human evaluators judging the quality of the generated queries. While this can be more subjective and labor-intensive, it can also provide more nuanced insights into the performance of the models. Table 3 provides a comparative analysis of different evaluation metrics.

5.1 String-based Matching

5.1.1 Exact String Match

Exact String Match [25] is the strictest form of string-based evaluation, where the generated functional representation query must be exactly identical to the target query down to every character. That means the order of elements, the choice of synonyms, and even the formatting must match exactly. The strengths of Exact String Match lie in its high efficiency, automatic judgment capabilities, and low implementation and execution complexities. Additionally, it

TABLE 3
Comparative Analysis of Evaluation Metrics

Type	Method	Advantages	Disadvantages
String-based Matching	Exact String Match	High efficiency, wide applicability	Cannot handle alias expressions
	Fuzzy Match	Suitable for complex queries	Insufficient precision
	Component Match	Can handle simple alias expressions	Needs to be customized
Execution-based Matching	Naive Execution Match	Convenient, robust to alias expressions	Prone to false positives
	Test Suite Match	Can handle semantically close expressions	Needs to be customized
Manual Evaluation	Manual Evaluation	Precise, flexible	High cost, low efficiency

places no restrictions on the formal language, making it broadly applicable. However, while this metric can be useful in some cases, it is often seen as too rigid because it can overlook semantically equivalent outputs that differ slightly in syntax or structure.

5.1.2 Fuzzy Match

To address the rigidity of the exact string match, Fuzzy Match allows for approximate matching between the predicted output and the reference [119]. Unlike Exact String Match, which necessitates a perfect character-by-character alignment, Fuzzy Match provides a degree of flexibility. This method quantifies similarity by assigning scores based on the closeness of the two strings, with BLEU being the most prevalent among such indicators [18]. This is particularly useful in scenarios where slight variations in phrasing or syntax can lead to semantically equivalent outputs. While Fuzzy Match offers a more nuanced assessment, especially in cases with minor discrepancies, it may sometimes be overly lenient, potentially overlooking significant errors in the predicted outputs [62].

5.1.3 Component Match

Component matching provides a more granular evaluation approach, focusing on individual components or segments of the predicted output rather than the entire string. In scenarios where the semantic expression is composed of multiple distinct parts or components, an exact or fuzzy string match might be too strict or lenient. Component match aims to assess the correctness of each segment independently, ensuring that each part of the output adheres to the expected reference. Component Match is widely adopted in Text-to-SQL evaluation, recognized as Exact Set Match proposed by Yu et al. [127] in the Spider dataset. Instead of assessing the SQL query as a whole, they determine the correctness by comparing individual sub-clauses of the SQL queries. This approach ensures that even if the sequence order of a generated query is different, it is still considered correct as long as all the necessary components or sub-clauses are present. The component match is also used in Text-to-Vis evaluation. RGVisNet [98] measures the accuracy of the three components in a Vega-lite specification (vis type, axis, and data). Another model, Seq2Vis, also measures whether the final result and its components are correctly matched, offering a detailed and multi-faceted approach to visualization evaluation.

5.2 Execution-based Matching

5.2.1 Execution Match

Unlike methods that directly compare the structural components or strings of the output, execution match evaluates the correctness of a semantic expression based on its actual execution results. If the results are the same, the generated query is considered correct, regardless of its syntactic differences with the reference query. This metric is particularly beneficial in contexts where distinct semantic expressions can lead to the same desired output, circumventing the strictness of string-based metrics that might yield false negatives. It is typically more robust in the Text-to-SQL task [127], given that SQL, as a language designed for database operations, allows numerous queries to fetch the same result from a database, even if their structures differ. Similarly, in the Text-to-Vis domain, execution match can be employed to determine if a generated visualization specification correctly visualizes the desired data insights, regardless of its exact structure or components.

5.2.2 Test Suite Match

Evaluating the correctness of semantic expressions based purely on execution results can be misleading. This is because different semantic expressions might produce identical results, leading to false positives and potentially inflating the perceived performance of a semantic parser. Addressing this challenge, Zhong et al. proposed a refined metric inspired by the concept of Distinguishing Testing in software engineering [132]. Central to this method is creating multiple knowledge base variants specifically designed to differentiate between the predicted SQL query and the reference. These variants are crafted by altering the stored values and their order in the original knowledge base. For an expression to be considered correct, its execution results must align with those of the reference expression across all these variants. This criterion ensures that the validation isn't merely based on matching outcomes in a single context but remains consistent across a spectrum of scenarios.

5.3 Manual Evaluation

5.3.1 Human Evaluation of SQL

Manual evaluation is a crucial component of Text-to-SQL task assessment. A pivotal role of human evaluation in Text-to-SQL is discerning the subtle nuances of semantic equivalence in contexts where execution results of two expressions might differ, but both are still valid in real-world scenarios. For instance, in film queries, both *SELECT rating FROM film WHERE title = "Titanic"* and *SELECT id, rating FROM film WHERE title = "Titanic"* are valid responses to a request for the rating of "Titanic" despite producing different outputs. Dahl et al. introduced an approach where an execution result is deemed correct if it falls within a predefined interval. However, determining this interval often requires human judgment, as seen in their work where the correct answer range was manually established [13]. While manual evaluation offers depth, it is labor-intensive and time-consuming, and the subjectivity of the evaluators can influence the results. Therefore, manual evaluation is typically used in conjunction with automatic evaluation metrics to provide a more rounded evaluation of a Text-to-SQL model's performance.

5.3.2 User Study of Vis

Unlike Text-to-SQL, a User Study provides a practical evaluation of Text-to-Vis models, focusing on user experience, model effectiveness, and potential areas of improvement. For example, ncNet colleced user feedback, highlighting its practicality across various domains. Such studies often assess system user-friendliness and efficiency, capturing user feedback on system speed and ease of use. They also collect user preferences and suggestions to direct subsequent refinements. Essentially, user studies offer valuable insights into Text-to-Vis models' real-world applicability and success.

5.4 Multi-turn Evaluation

In addition to the above-mentioned classification criteria for evaluation metrics, in dialogue-based multi-turn scenarios, evaluation metrics can also be divided into Question Match accuracy and Interaction Match accuracy [128].

Question Match focuses on individual questions within an interaction, evaluating how closely the model's predictions align with the ground truth in each set of question-query pairs. This metric is computed by determining the ratio of questions with matches to the total number of questions in the interaction. According to different task objectives, the evaluation metrics mentioned above, such as string-based matching, execution-based matching, and manual evaluation, can all be used as evaluation indicators for evaluating the accuracy of Question Match. Interaction Match, in contrast, adopts a more holistic lens, treating the entire interaction as an indivisible entity. An interaction is deemed perfectly matched, receiving a score of 1, if every single question within the interaction exhibits a match. The final score is then deduced by dividing the number of perfectly matched interactions by the total count of interactions.

6 System Design

System architecture is crucial in shaping the capabilities of natural Language interfaces for tabular data querying and visualization. Various architectural paradigms have emerged as the field has evolved, each tailored to specific challenges and needs. While in-depth analyses and comparisons of earlier systems can be found in surveys like [1], [31], [54], this section will categorize these systems into four main architectural types: rule-based systems, parsing-based systems, multi-stage systems, and end-to-end systems. Table 4 presents a comprehensive overview of various Text-to-SQL and Text-to-Vis systems.

6.1 Rule-based System

Rule-based systems stand as foundational architectures for natural language interfaces to databases. These systems leverage a set of predefined rules, mapping natural language inputs directly to database queries or visualizations. For Text-to-SQL, systems like PRECISE [80] and NaLIR [57] employ rule-based strategies, translating linguistic patterns into SQL queries. In the Text-to-Vis context, DataTone [30] represents this approach, converting user language into visualization specifications via established patterns. While precise, rule-based systems can face challenges in scalability and adaptability to diverse linguistic constructs.

6.2 Parsing-based System

Parsing-based systems primarily focus on understanding the inherent grammatical structure of the input question. Drawing inspiration from traditional linguistic parsing, these systems convert natural language questions into syntactic structures or logical forms. In the field of Text-to-SQL, systems such as SQLova [46] and Seq2Tree [19] utilize semantic parsers to bridge the gap between natural language and structured database queries. For Text-to-Vis, systems ncNet [71] process user queries through semantic parsing, transforming them into Visualization Query Languages (VQL). Parsing-based systems prioritize linguistic structure and semantics, offering depth in understanding, but might struggle with the variability and ambiguity inherent to natural language.

6.3 Multi-stage System

Multi-stage systems in natural language interfaces for tabular data operate through sequenced processing pipelines. These systems dissect the overarching task into distinct stages, each addressing a particular sub-task. This layered approach allows for focused improvements at every juncture. Within the Text-to-SQL domain, the DIN-SQL system [82] exemplifies this architecture, segmenting SQL generation into stages for schema linking, query classification and decomposition, SQL generation, and self-correction. In the Text-to-Vis sphere, DeepEye [68] emerges as a notable multi-stage system to discern the quality of visualizations, rank them, and optimally select the top-k visualizations from a dataset. By segmenting the process, multi-stage systems can apply tailored techniques to each segment, enhancing accuracy. However, the modular approach demands careful orchestration between stages to ensure coherency in the final output and can potentially bring higher computational cost.

6.4 End-to-end System

End-to-end systems represent a holistic approach to natural language interfaces for tabular data. Rather than relying on intermediate representations or multi-phase processing,

TABLE 4
Comparison of Different Systems of Text-to-SQL and Text-to-Vis

Type	SQL system example	Vis system example	Advantages	Disadvantages
Rule-based	NaLIR, PRECISE	DataTone	Robustness and consistency for familiar queries	Limited adaptability
Parsing-based	SQLova, Seq2Tree	ncNet	Grasps deeper language structures	Struggles with ambiguity
Multi-stage	DIN-SQL	DeepEye	Enhanced accuracy and flexibility	Synchronization challenges
End-to-end	Photon, VoiceQuerySystem	Sevi, DeepTrack	High adaptability, unified training process	Difficult to interpret and debug

these systems process input questions and directly generate the desired output in one cohesive step. For example, Photon [130] offers a modular framework tailored for industrial applications of Text-to-SQL system. It takes a user's question and a database schema, directly generating SQL and executing it to produce the desired result, with its core strength lying in its SQL parser and a unique confusion detection mechanism. Another exemplar is VoiceQuerySystem [96], which elevates the user experience by converting voice-based queries directly into SQL, bypassing the need for text as an intermediary. Similarly, in the Text-to-Vis domain, Sevi [107] stands out as an end-to-end visualization assistant. It empowers novices to craft visualizations using either natural language or voice commands. Furthermore, DeepTrack [67] integrates data preparation, visualization selection, and intuitive interactions within a singular framework, exemplifying the comprehensive capabilities of end-to-end systems.

7 Future Research Direction

As the field of natural language interfaces for tabular data querying and visualization continues to evolve, new challenges and opportunities emerge, leading to exciting future research directions. While the advancements brought about by semantic parsing techniques and Large Language Models have significantly improved the capabilities of such interfaces, there remain areas that have not been fully explored or addressed. This section highlights six pivotal areas that promise to shape the domain's future, emphasizing the ongoing research evolution and its potential. Table 5 compares Text-to-SQL and Text-to-Vis tasks across these research directions.

7.1 Advancing Neural Models and Approaches

The landscape of Natural Language Interfaces for Tabular Data has seen impressive strides, especially with the advent of neural models in the text-to-SQL domain. However, there remains substantial room for improvement and innovation. While plenty of models have been proposed for text-to-SQL tasks, continual refinement is essential to handle more complex queries, multi-turn interactions, and domain-specific problems [14]. Concurrently, the text-to-visualization domain hasn't witnessed the same influx of neural network-based models. The challenges here are multifold: from generating diverse visualizations based on user intent to ensuring those visualizations maintain both accuracy and aesthetic appeal [93]. For both domains, it's vital to push the boundaries of current neural architectures. This could involve exploring deeper networks, advanced attention mechanisms, or hybrid models combining rule-based logic with neural insights. Leveraging external knowledge bases, transfer learning, and multi-modal strategies could further optimize the interpretation and translation of user intent into SQL queries or visual representations.

7.2 Harnessing Potential of Large Language Models

Large Language Models (LLMs) like ChatGPT have revolutionized various Natural Language Processing domains with their profound text understanding and generation capabilities. Despite this, exploring LLMs in the context of natural language interfaces for databases remains relatively nascent. While preliminary efforts have begun integrating LLMs into text-to-SQL and text-to-visualization systems [10], [82], the vast potential of LLMs has not been fully harnessed. Their ability to capture context, understand nuances, and generalize from limited examples could be invaluable in understanding and translating complex user queries. However, merely deploying LLMs without customization might not be optimal. Future research should focus on tailoring these models to the specific challenges of querying and visualization. This might involve adapting LLMs on domain-specific datasets, integrating them with existing architectures, or developing novel prompting strategies to better align them with the tasks at hand.

7.3 Exploring Advanced Learning Methods

The heavy reliance of traditional supervised learning on large labeled datasets poses challenges for evolving natural language interfaces for tabular data. This underscores the need for alternative learning approaches. Semi-supervised and weakly supervised methods, which capitalize on unlabeled data or weak supervision signals, present viable solutions [36]. For example, implicit user interactions might offer weak guidance for model refinement. Additionally, parameter-efficient training methods like Adapter [42] and LoRA [43] have demonstrated superior data efficiency, especially in low-resource settings, compared to traditional fine-tuning methods. Fusing large pre-trained models with these parameter-efficient techniques hints at a promising future for data-efficient semantic parsing.

7.4 Constructing Large-Scale and Diverse Datasets

The potency of natural language interfaces for databases depends on high-quality, diverse datasets. While several datasets are tailored for text-to-SQL and text-to-visualization tasks, there's a pressing need for even larger-scale, more varied datasets. Such datasets foster better

TABLE 5
Comparison between Text-to-SQL and Text-to-Vis Research

Aspects	Text-to-SQL	Text-to-Vis
Neural Models and Approaches	Notable advancements with a variety of models	Still in nascent stages with limited models
Integration of LLMs	Preliminary efforts with room for more exploration	Limited work, significant potential
Learning Methods	Mainly supervised; early exploration of semi-supervised methods	Predominantly supervised
Datasets	Several large-scale datasets available	Fewer datasets; need for more diversity
Robustness and Generalizability	Increased focus, especially for complex queries	Emerging focus; essential for diverse visualizations
Advanced Applications	Integration with chatbots, recommendation systems	Potential for multimodal systems, dynamic visualizations

generalization and robustness to a broad spectrum of user queries, spanning various domains and complexities. Moreover, the current dataset landscape is predominantly English-centric, overlooking the global spectrum of data user [37]. Embracing multilingual or under-represented language datasets can amplify the reach and inclusivity of these interfaces.

7.5 Advancing Robustness and Generalizability

As natural language interfaces for tabular data become more integral in various applications, the robustness and generalizability of the underlying models and systems are central. It's not just about achieving high performance on benchmark datasets; real-world scenarios demand models that can reliably handle diverse, unexpected, and sometimes adversarial inputs.

• *Robustness Against Adversarial and Out-of-Distribution Perturbations.* As with many machine learning models, adversarial attacks or unexpected inputs can pose significant challenges. There's a need for models that can gracefully handle and respond to such inputs without compromising on accuracy or reliability. This involves developing models inherently resistant to such perturbations and creating datasets that can effectively train and test such robustness [8].

• *Compositional Generalization.* The ability for models to understand and combine known concepts in novel ways is vital. For instance, if a model understands two separate queries, it should ideally be able to handle a composite query that combines elements of both. This capability ensures that models can effectively tackle unseen queries by leveraging their understanding of underlying concepts.

• *Domain Generalization.* As these interfaces permeate various sectors, models should adapt across domains and incorporate domain-specific knowledge. This ensures that, while retaining versatility, models are attuned to the nuances of diverse queries, from finance to healthcare and beyond [28]. Future research should prioritize these aspects of robustness and generalizability.

7.6 Pioneering Advanced Applications in the LLM Era

With the dawn of the Large Language Models era, there's an unprecedented opportunity to revolutionize the applications and systems of natural language interfaces for databases. Leveraging the depth and breadth of LLMs paves the way for more sophisticated, intuitive, and versatile applications.

• *Multimodal Systems.* Combining the power of LLMs with other modalities, such as visual or auditory inputs, can lead to the creation of truly multi-modal systems. Imagine querying a database not just with text, but with images, voice commands, or even gestures. Such systems can cater to a broader audience and offer more dynamic and natural interactions.

• *Integrated Systems.* As LLMs continue to excel in various tasks, there's potential for integrating natural language interfaces with other functionalities, like document summarization, recommendation systems, or even chatbots. This can result in comprehensive systems where users can query data, get summaries, seek recommendations, and more, all within a unified, language-centric interface.

• *User-Centric Design.* The LLM era emphasizes user interaction. There's a need for applications prioritizing user experience, offering intuitive interfaces, interactive feedback, and personalized responses. By harnessing the capabilities of these models and focusing on creating holistic, user-centric applications, we can set the stage for a future where data interaction is both efficient and delightful.

8 Conclusion

In this survey, we explore Natural Language Interfaces for Tabular Data Querying and Visualization in-depth, delving into the intricacies of the field, its evolution, and the challenges it addresses. We trace its evolution from foundational problem definitions to state-of-the-art approaches. We highlight the significance of diverse datasets fueling these interfaces and discuss the metrics that gauge their efficacy. By exploring system architectures, we examine the differences of distinct system designs. Lastly, our gaze turns toward the horizon, pointing to promising research avenues in the era of Large Language Models. As this dynamic field evolves, our exploration offers a concise snapshot of its current state, challenges, and potential.

References

[1] K. Affolter, K. Stockinger, and A. Bernstein. A comparative survey of recent natural language interfaces for databases. *VLDB J.*, 28(5):793–819, 2019.

[2] I. Androutsopoulos, G. D. Ritchie, and P. Thanisch. Natural language interfaces to databases - an introduction. *Nat. Lang. Eng.*, 1(1):29–81, 1995.

[3] C. Baik, H. V. Jagadish, and Y. Li. Bridging the semantic gap with SQL query logs in natural language interfaces to databases. In *35th IEEE International Conference on Data Engineering, ICDE 2019, Macao, China, April 8-11, 2019*, pages 374–385. IEEE, 2019.

[4] B. Bogin, J. Berant, and M. Gardner. Representing schema structure with graph neural networks for text-to-sql parsing. In A. Korhonen, D. R. Traum, and L. Màrquez, editors, *Proceedings of ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4560–4565, 2019.

[5] B. Bogin, M. Gardner, and J. Berant. Global reasoning over database structures for text-to-sql parsing. In K. Inui, J. Jiang, V. Ng, and X. Wan, editors, *Proceedings of EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3657–3662, 2019.

[6] R. Cai, J. Yuan, B. Xu, and Z. Hao. SADGA: structure-aware dual graph aggregation network for text-to-sql. In M. Ranzato, A. Beygelzimer, Y. N. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 7664–7676, 2021.

[7] R. Cao, L. Chen, Z. Chen, Y. Zhao, S. Zhu, and K. Yu. LGESQL: line graph enhanced text-to-sql model with mixed local and non-local relations. In C. Zong, F. Xia, W. Li, and R. Navigli, editors, *Proceedings of ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 2541–2555, 2021.

[8] S. Chang, J. Wang, M. Dong, L. Pan, H. Zhu, A. H. Li, W. Lan, S. Zhang, J. Jiang, J. Lilien, S. Ash, W. Y. Wang, Z. Wang, V. Castelli, P. Ng, and B. Xiang. Dr.spider: A diagnostic evaluation benchmark towards text-to-sql robustness. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*, 2023.

[9] M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. de Oliveira Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman, A. Ray, R. Puri, G. Krueger, M. Petrov, H. Khlaaf, G. Sastry, P. Mishkin, B. Chan, S. Gray, N. Ryder, M. Pavlov, A. Power, L. Kaiser, M. Bavarian, C. Winter, P. Tillet, F. P. Such, D. Cummings, M. Plappert, F. Chantzis, E. Barnes, A. Herbert-Voss, W. H. Guss, A. Nichol, A. Paino, N. Tezak, J. Tang, I. Babuschkin, S. Balaji, S. Jain, W. Saunders, C. Hesse, A. N. Carr, J. Leike, J. Achiam, V. Misra, E. Morikawa, A. Radford, M. Knight, M. Brundage, M. Murati, K. Mayer, P. Welinder, B. McGrew, D. Amodei, S. McCandlish, I. Sutskever, and W. Zaremba. Evaluating large language models trained on code. *CoRR*, abs/2107.03374, 2021.

[10] Y. Chen, R. Li, A. Mac, T. Xie, T. Yu, and E. Wu. NL2INTERFACE: interactive visualization interface generation from natural language queries. *CoRR*, abs/2209.08834, 2022.

[11] Z. Chen, L. Chen, Y. Zhao, R. Cao, Z. Xu, S. Zhu, and K. Yu. Shadowgnn: Graph projection neural network for text-to-sql parser. In K. Toutanova, A. Rumshisky, L. Zettlemoyer, D. Hakkani-Tür, I. Beltagy, S. Bethard, R. Cotterell, T. Chakraborty, and Y. Zhou, editors, *Proceedings of NAACL-HLT 2021, Online, June 6-11, 2021*, pages 5567–5577, 2021.

[12] D. Choi, M. 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*, 47(2):309–332, 2021.

[13] D. A. Dahl, M. Bates, M. Brown, W. M. Fisher, K. Hunicke-Smith, D. S. Pallett, C. Pao, A. I. Rudnicky, and E. Shriberg. Expanding the scope of the ATIS task: The ATIS-3 corpus. In *Human Language Technology, Proceedings of a Workshop held at Plainsboro, New Jerey, USA, March 8-11, 1994*, 1994.

[14] N. Deng, Y. Chen, and Y. Zhang. Recent advances in text-to-sql: A survey of what we have and what we expect. In N. Calzolari, C. Huang, H. Kim, J. Pustejovsky, L. Wanner, K. Choi, P. Ryu, H. Chen, L. Donatelli, H. Ji, S. Kurohashi, P. Paggio, N. Xue, S. Kim, Y. Hahm, Z. He, T. K. Lee, E. Santus, F. Bond, and S. Na, editors, *Proceedings of COLING 2022, Gyeongju, Republic of Korea, October 12-17, 2022*, pages 2166–2187, 2022.

[15] X. Deng, A. H. Awadallah, C. Meek, O. Polozov, H. Sun, and M. Richardson. Structure-grounded pretraining for text-to-sql. In K. Toutanova, A. Rumshisky, L. Zettlemoyer, D. Hakkani-Tür, I. Beltagy, S. Bethard, R. Cotterell, T. Chakraborty, and Y. Zhou, editors, *Proceedings of NAACL-HLT 2021, Online, June 6-11, 2021*, pages 1337–1350, 2021.

[16] J. Devlin, M. Chang, K. Lee, and K. Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In J. Burstein, C. Doran, and T. Solorio, editors, *Proceedings of NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019.

[17] V. Dibia and Ç. Demiralp. Data2vis: Automatic generation of data visualizations using sequence-to-sequence recurrent neural networks. *IEEE Computer Graphics and Applications*, 39(5):33–46, 2019.

[18] G. Doddington. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the Second International Conference on Human Language Technology Research*, HLT '02, page 138–145, San Francisco, CA, USA, 2002.

[19] L. Dong and M. Lapata. Language to logical form with neural attention. In *Proceedings of ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*, 2016.

[20] L. Dong and M. Lapata. Coarse-to-fine decoding for neural semantic parsing. In I. Gurevych and Y. Miyao, editors, *Proceedings of ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 731–742, 2018.

[21] Q. Dong, L. Li, D. Dai, C. Zheng, Z. Wu, B. Chang, X. Sun, J. Xu, L. Li, and Z. Sui. A survey on in-context learning, 2023.

[22] X. Dong, C. Zhang, Y. Ge, Y. Mao, Y. Gao, L. Chen, J. Lin, and D. Lou. C3: zero-shot text-to-sql with chatgpt. *CoRR*, abs/2307.07306, 2023.

[23] L. Dou, Y. Gao, X. Liu, M. Pan, D. Wang, W. Che, D. Zhan, M. Kan, and J. Lou. Towards knowledge-intensive text-to-sql semantic parsing with formulaic knowledge. In Y. Goldberg, Z. Kozareva, and Y. Zhang, editors, *Proceedings of EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 5240–5253, 2022.

[24] L. Dou, Y. Gao, M. Pan, D. Wang, J. Lou, W. Che, and D. Zhan. Unisar: A unified structure-aware autoregressive language model for text-to-sql. *CoRR*, abs/2203.07781, 2022.

[25] C. Finegan-Dollak, J. K. Kummerfeld, L. Zhang, K. Ramanathan, S. Sadasivam, R. Zhang, and D. R. Radev. Improving text-to-sql evaluation methodology. In I. Gurevych and Y. Miyao, editors, *Proceedings of ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 351–360, 2018.

[26] Y. Gan, X. Chen, Q. Huang, and M. Purver. Measuring and improving compositional generalization in text-to-sql via component alignment. In M. Carpuat, M. de Marneffe, and I. V. M. Ruíz, editors, *Findings of the Association for Computational Linguistics: NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 831–843, 2022.

[27] Y. Gan, X. Chen, Q. Huang, M. Purver, J. R. Woodward, J. Xie, and P. Huang. Towards robustness of text-to-sql models against synonym substitution. In C. Zong, F. Xia, W. Li, and R. Navigli, editors, *Proceedings of ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 2505–2515, 2021.

[28] Y. Gan, X. Chen, and M. Purver. Exploring underexplored limitations of cross-domain text-to-sql generalization. In M. Moens, X. Huang, L. Specia, and S. W. Yih, editors, *Proceedings of EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 8926–8931, 2021.

[29] Y. Gan, X. Chen, J. Xie, M. Purver, J. R. Woodward, J. H. Drake, and Q. Zhang. Natural SQL: making SQL easier to infer from natural language specifications. In M. Moens, X. Huang, L. Specia, and S. W. Yih, editors, *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*, pages 2030–2042, 2021.

[30] T. Gao, M. Dontcheva, E. Adar, Z. Liu, and K. G. Karahalios. Datatone: Managing ambiguity in natural language interfaces for data visualization. In C. Latulipe, B. Hartmann, and T. Grossman, editors, *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology, UIST 2015, Charlotte, NC, USA, November 8-11, 2015*, pages 489–500. ACM, 2015.

[31] O. Gkini, T. Belmpas, G. Koutrika, and Y. E. Ioannidis. An in-depth benchmarking of text-to-sql systems. In G. Li, Z. Li, S. Idreos, and D. Srivastava, editors, *SIGMOD '21: International Conference on Management of Data, Virtual Event, China, June 20-25, 2021*, pages 632–644. ACM, 2021.

[32] B. J. Grosz. TEAM: A transportable natural-language interface system. In *1st Applied Natural Language Processing Conference, ANLP 1983, Miramar-Sheraton Hotel, Santa Monica, California, USA, February 1-3, 1983*, pages 39–45, 1983.

[33] Z. Gu, J. Fan, N. Tang, L. Cao, B. Jia, S. Madden, and X. Du. Few-shot text-to-sql translation using structure and content prompt learning. *Proc. ACM Manag. Data*, 1(2):147:1–147:28, 2023.

[34] Z. Gu, J. Fan, N. Tang, S. Zhang, Y. Zhang, Z. Chen, L. Cao, G. Li, S. Madden, and X. Du. Interleaving pre-trained language models and large language models for zero-shot NL2SQL generation. *CoRR*, abs/2306.08891, 2023.

[35] C. Guo, Z. Tian, J. Tang, S. Li, Z. Wen, K. Wang, and T. Wang. Retrieval-augmented gpt-3.5-based text-to-sql framework with sample-aware prompting and dynamic revision chain. *CoRR*, abs/2307.05074, 2023.

[36] J. Guo, J. Lou, T. Liu, and D. Zhang. Weakly supervised semantic parsing by learning from mistakes. In M. Moens, X. Huang, L. Specia, and S. W. Yih, editors, *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*, pages 2603–2617, 2021.

[37] J. Guo, Z. Si, Y. Wang, Q. Liu, M. Fan, J. Lou, Z. Yang, and T. Liu. Chase: A large-scale and pragmatic chinese dataset for cross-database context-dependent text-to-sql. In C. Zong, F. Xia, W. Li, and R. Navigli, editors, *Proceedings of ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 2316–2331, 2021.

[38] J. Guo, Z. Zhan, Y. Gao, Y. Xiao, J. Lou, T. Liu, and D. Zhang. Towards complex text-to-sql in cross-domain database with intermediate representation. In A. Korhonen, D. R. Traum, and L. Màrquez, editors, *Proceedings of ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4524–4535, 2019.

[39] M. Hazoom, V. Malik, and B. Bogin. Text-to-sql in the wild: A naturally-occurring dataset based on stack exchange data. *CoRR*, abs/2106.05006, 2021.

[40] P. He, Y. Mao, K. Chakrabarti, and W. Chen. X-SQL: reinforce schema representation with context. *CoRR*, abs/1908.08113, 2019.

[41] E. Hoque, V. Setlur, M. Tory, and I. Dykeman. Applying pragmatics principles for interaction with visual analytics. *IEEE Trans. Vis. Comput. Graph.*, 24(1):309–318, 2018.

[42] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. de Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly. Parameter-efficient transfer learning for NLP. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799, 2019.

[43] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*, 2022.

[44] B. Hui, R. Geng, Q. Ren, B. Li, Y. Li, J. Sun, F. Huang, L. Si, P. Zhu, and X. Zhu. Dynamic hybrid relation exploration network for cross-domain context-dependent semantic parsing. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 13116–13124, 2021.

[45] B. Hui, R. Geng, L. Wang, B. Qin, Y. Li, B. Li, J. Sun, and Y. Li. S^2sql: Injecting syntax to question-schema interaction graph encoder for text-to-sql parsers. In S. Muresan, P. Nakov, and A. Villavicencio, editors, *Findings of the Association for Computational Linguistics: ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 1254–1262, 2022.

[46] W. Hwang, J. Yim, S. Park, and M. Seo. A comprehensive exploration on wikisql with table-aware word contextualization. *CoRR*, abs/1902.01069, 2019.

[47] R. C. A. Iacob, F. Brad, E. S. Apostol, C. Truica, I. Hosu, and T. Rebedea. Neural approaches for natural language interfaces to databases: A survey. In D. Scott, N. Bel, and C. Zong, editors, *Proceedings of COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 381–395, 2020.

[48] S. Iyer, I. Konstas, A. Cheung, J. Krishnamurthy, and L. Zettlemoyer. Learning a neural semantic parser from user feedback. In R. Barzilay and M. Kan, editors, *Proceedings of ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 963–973, 2017.

[49] M. A. José and F. G. Cozman. mrat-sql+gap: A portuguese text-to-sql transformer. In A. Britto and K. V. Delgado, editors, *Intelligent Systems - 10th Brazilian Conference, BRACIS 2021, Virtual Event, November 29 - December 3, 2021, Proceedings, Part II*, volume 13074 of *Lecture Notes in Computer Science*, pages 511–525, 2021.

[50] A. Kamath and R. Das. A survey on semantic parsing. In *1st Conference on Automated Knowledge Base Construction, AKBC 2019, Amherst, MA, USA, May 20-22, 2019*, 2019.

[51] J. Kassel and M. Rohs. Valletto: A multimodal interface for ubiquitous visual analytics. In R. L. Mandryk, M. Hancock, M. Perry, and A. L. Cox, editors, *Extended Abstracts of the 2018 CHI Conference on Human Factors in Computing Systems, CHI 2018, Montreal, QC, Canada, April 21-26, 2018*. ACM, 2018.

[52] G. Katsogiannis-Meimarakis and G. Koutrika. A deep dive into deep learning approaches for text-to-sql systems. In G. Li, Z. Li, S. Idreos, and D. Srivastava, editors, *SIGMOD '21: International Conference on Management of Data, Virtual Event, China, June 20-25, 2021*, pages 2846–2851, 2021.

[53] G. Katsogiannis-Meimarakis and G. Koutrika. A survey on deep learning approaches for text-to-sql. *VLDB J.*, 32(4):905–936, 2023.

[54] H. Kim, B. So, W. Han, and H. Lee. Natural language to SQL: where are we today? *Proc. VLDB Endow.*, 13(10):1737–1750, 2020.

[55] A. Kumar, J. Aurisano, B. D. Eugenio, A. E. Johnson, A. Gonzalez, and J. Leigh. Towards a dialogue system that supports rich visualizations of data. In *Proceedings of the SIGDIAL 2016 Conference, The 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue, 13-15 September 2016, Los Angeles, CA, USA*, pages 304–309, 2016.

[56] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In D. Jurafsky, J. Chai, N. Schluter, and J. R. Tetreault, editors, *Proceedings of ACL 2020, Online, July 5-10, 2020*, pages 7871–7880, 2020.

[57] F. Li and H. V. Jagadish. Constructing an interactive natural language interface for relational databases. *Proc. VLDB Endow.*, 8(1):73–84, 2014.

[58] H. Li, J. Zhang, C. Li, and H. Chen. RESDSQL: decoupling schema linking and skeleton parsing for text-to-sql. In B. Williams, Y. Chen, and J. Neville, editors, *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence, IAAI 2023, Thirteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2023, Washington, DC, USA, February 7-14, 2023*, pages 13067–13075, 2023.

[59] J. Li, B. Hui, R. Cheng, B. Qin, C. Ma, N. Huo, F. Huang, W. Du, L. Si, and Y. Li. Graphix-t5: Mixing pre-trained transformers with graph-aware layers for text-to-sql parsing. In B. Williams, Y. Chen, and J. Neville, editors, *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence, IAAI 2023, Thirteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2023, Washington, DC, USA, February 7-14, 2023*, pages 13076–13084, 2023.

[60] J. Li, B. Hui, G. Qu, B. Li, J. Yang, B. Li, B. Wang, B. Qin, R. Cao, R. Geng, N. Huo, X. Zhou, C. Ma, G. Li, K. C. Chang, F. Huang, R. Cheng, and Y. Li. Can LLM already serve as A database interface? A big bench for large-scale database grounded text-to-sqls. *CoRR*, abs/2305.03111, 2023.

[61] X. V. Lin, R. Socher, and C. Xiong. Bridging textual and tabular data for cross-domain text-to-sql semantic parsing. *CoRR*, abs/2012.12627, 2020.

[62] X. V. Lin, C. Wang, L. Zettlemoyer, and M. D. Ernst. Nl2bash: A corpus and semantic parser for natural language interface to the linux operating system. In N. Calzolari, K. Choukri, C. Cieri, T. Declerck, S. Goggi, K. Hasida, H. Isahara, B. Maegaard, J. Mariani, H. Mazo, A. Moreno, J. Odijk, S. Piperidis, and T. Tokunaga, editors, *Proceedings of the Eleventh International Conference on Language Resources and Evaluation, LREC 2018, Miyazaki, Japan, May 7-12, 2018*, 2018.

[63] A. Liu, X. Hu, L. Wen, and P. S. Yu. A comprehensive evaluation of chatgpt's zero-shot text-to-sql capability. *CoRR*, abs/2303.13547, 2023.

[64] C. Liu, Y. Han, R. Jiang, and X. Yuan. Advisor: Automatic visualization answer for natural-language question on tabular data. In *14th IEEE Pacific Visualization Symposium, PacificVis 2021, Tianjin, China, April 19-21, 2021*, pages 11–20. IEEE, 2021.

[65] X. Liu and Z. Tan. Divide and prompt: Chain of thought prompting for text-to-sql. *CoRR*, abs/2304.11556, 2023.

[66] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019.

[67] Y. Luo, W. Li, T. Zhao, X. Yu, L. Zhang, G. Li, and N. Tang. Deeptrack: Monitoring and exploring spatio-temporal data - A case of tracking COVID-19 -. *Proc. VLDB Endow.*, 13(12):2841–2844, 2020.

[68] Y. Luo, X. Qin, N. Tang, G. Li, and X. Wang. Deepeye: Creating good data visualizations by keyword search. In G. Das, C. M. Jermaine, and P. A. Bernstein, editors, *Proceedings of the 2018 International Conference on Management of Data, SIGMOD Conference 2018, Houston, TX, USA, June 10-15, 2018*, pages 1733–1736. ACM, 2018.

[69] Y. Luo, J. Tang, and G. Li. nvbench: A large-scale synthesized dataset for cross-domain natural language to visualization task. *CoRR*, abs/2112.12926, 2021.

[70] Y. Luo, N. Tang, G. Li, C. Chai, W. Li, and X. Qin. Synthesizing natural language to visualization (NL2VIS) benchmarks from NL2SQL benchmarks. In G. Li, Z. Li, S. Idreos, and D. Srivastava, editors, *SIGMOD '21: International Conference on Management of Data, Virtual Event, China, June 20-25, 2021*, pages 1235–1247. ACM, 2021.

[71] Y. Luo, N. Tang, G. Li, J. Tang, C. Chai, and X. Qin. Natural language to visualization by neural machine translation. *IEEE Trans. Vis. Comput. Graph.*, 28(1):217–226, 2022.

[72] Q. Lyu, K. Chakrabarti, S. Hathi, S. Kundu, J. Zhang, and Z. Chen. Hybrid ranking network for text-to-sql. *CoRR*, abs/2008.04759, 2020.

[73] P. Maddigan and T. Susnjak. Chat2vis: Generating data visualisations via natural language using chatgpt, codex and GPT-3 large language models. *CoRR*, abs/2302.02094, 2023.

[74] Q. Min, Y. Shi, and Y. Zhang. A pilot study for chinese SQL semantic parsing. In K. Inui, J. Jiang, V. Ng, and X. Wan, editors, *Proceedings of EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3650–3656, 2019.

[75] R. Mitra, A. Narechania, A. Endert, and J. T. Stasko. Facilitating conversational interaction in natural language interfaces for visualization. *CoRR*, abs/2207.00189, 2022.

[76] L. Nan, Y. Zhao, W. Zou, N. Ri, J. Tae, E. Zhang, A. Cohan, and D. Radev. Enhancing few-shot text-to-sql capabilities of large language models: A study on prompt design strategies. *CoRR*, abs/2305.12586, 2023.

[77] A. Narechania, A. Srinivasan, and J. T. Stasko. NL4DV: A toolkit for generating analytic specifications for data visualization from natural language queries. *IEEE Trans. Vis. Comput. Graph.*, 27(2):369–379, 2021.

[78] A. T. Nguyen, M. H. Dao, and D. Q. Nguyen. A pilot study of text-to-sql semantic parsing for vietnamese. In T. Cohn, Y. He, and Y. Liu, editors, *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 4079–4085, 2020.

[79] A. Patel, B. Li, M. S. Rasooli, N. Constant, C. Raffel, and C. Callison-Burch. Bidirectional language models are also few-shot learners. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*, 2023.

[80] A. Popescu, A. Armanasu, O. Etzioni, D. Ko, and A. Yates. Modern natural language interfaces to databases: Composing statistical parsing with semantic tractability. In *COLING 2004, 20th International Conference on Computational Linguistics, Proceedings of the Conference, 23-27 August 2004, Geneva, Switzerland*, 2004.

[81] A. Popescu, O. Etzioni, and H. A. Kautz. Towards a theory of natural language interfaces to databases. In D. B. Leake, W. L. Johnson, and E. André, editors, *Proceedings of IUI 2003, Miami, FL, USA, January 12-15, 2003*, pages 149–157, 2003.

[82] M. Pourreza and D. Rafiei. DIN-SQL: decomposed in-context learning of text-to-sql with self-correction. *CoRR*, abs/2304.11015, 2023.

[83] P. J. Price. Evaluation of spoken language systems: the ATIS domain. In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, USA, June 24-27, 1990*, 1990.

[84] X. Qin, Y. Luo, N. Tang, and G. Li. Making data visualization more efficient and effective: a survey. *VLDB J.*, 29(1):93–117, 2020.

[85] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67, 2020.

[86] N. Rajkumar, R. Li, and D. Bahdanau. Evaluating the text-to-sql capabilities of large language models. *CoRR*, abs/2204.00498, 2022.

[87] O. Rubin and J. Berant. Smbop: Semi-autoregressive bottom-up semantic parsing. In Z. Kozareva, S. Ravi, A. Vlachos, P. Agrawal, and A. F. T. Martins, editors, *Proceedings of SPNLP@ACL-IJCNLP 2021, Online, August 6, 2021*, pages 12–21, 2021.

[88] 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 Endow.*, 9(12):1209–1220, 2016.

[89] T. Scholak, N. Schucher, and D. Bahdanau. PICARD: parsing incrementally for constrained auto-regressive decoding from language models. In M. Moens, X. Huang, L. Specia, and S. W. Yih, editors, *Proceedings of EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 9895–9901, 2021.

[90] V. Setlur, S. E. Battersby, M. Tory, R. Gossweiler, and A. X. Chang. Eviza: A natural language interface for visual analysis. In J. Rekimoto, T. Igarashi, J. O. Wobbrock, and D. Avrahami, editors, *Proceedings of the 29th Annual Symposium on User Interface Software and Technology, UIST 2016, Tokyo, Japan, October 16-19, 2016*, pages 365–377. ACM, 2016.

[91] Y. Shao and N. Nakashole. Chartdialogs: Plotting from natural language instructions. In D. Jurafsky, J. Chai, N. Schluter, and J. R. Tetreault, editors, *Proceedings of ACL 2020, Online, July 5-10, 2020*, pages 3559–3574, 2020.

[92] P. Shaw, M. Chang, P. Pasupat, and K. Toutanova. Compositional generalization and natural language variation: Can a semantic parsing approach handle both? In C. Zong, F. Xia, W. Li, and R. Navigli, editors, *Proceedings of ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 922–938, 2021.

[93] L. Shen, E. Shen, Y. Luo, X. Yang, X. Hu, X. Zhang, Z. Tai, and J. Wang. Towards natural language interfaces for data visualization: A survey. *IEEE Trans. Vis. Comput. Graph.*, 29(6):3121–3144, 2023.

[94] P. Shi, P. Ng, Z. Wang, H. Zhu, A. H. Li, J. Wang, C. N. dos Santos, and B. Xiang. Learning contextual representations for semantic parsing with generation-augmented pre-training. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 13806–13814, 2021.

[95] T. Shi, K. Tatwawadi, K. Chakrabarti, Y. Mao, O. Polozov, and W. Chen. Incsql: Training incremental text-to-sql parsers with non-deterministic oracles. *CoRR*, abs/1809.05054, 2018.

[96] Y. Song, R. C. Wong, X. Zhao, and D. Jiang. Voicequerysystem: A voice-driven database querying system using natural language questions. In Z. G. Ives, A. Bonifati, and A. E. Abbadi, editors, *SIGMOD '22: International Conference on Management of Data, Philadelphia, PA, USA, June 12 - 17, 2022*, pages 2385–2388. ACM, 2022.

[97] Y. Song, X. Zhao, and R. C. Wong. Marrying dialogue systems with data visualization: Interactive data visualization generation from natural language conversations. *CoRR*, abs/2307.16013, 2023.

[98] Y. Song, X. Zhao, R. C. Wong, and D. Jiang. Rgvisnet: A hybrid retrieval-generation neural framework towards automatic data visualization generation. In A. Zhang and H. Rangwala, editors, *KDD '22: The 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, August 14 - 18, 2022*, pages 1646–1655. ACM, 2022.

[99] A. Srinivasan, B. Lee, N. H. Riche, S. M. Drucker, and K. Hinckley. Inchorus: Designing consistent multimodal interactions for data visualization on tablet devices. In R. Bernhaupt, F. F. Mueller, D. Verweij, J. Andres, J. McGrenere, A. Cockburn, I. Avellino, A. Goguey, P. Bjøn, S. Zhao, B. P. Samson, and R. Kocielnik, editors, *CHI '20: CHI Conference on Human Factors in Computing Systems, Honolulu, HI, USA, April 25-30, 2020*, pages 1–13. ACM, 2020.

[100] A. Srinivasan, N. Nyapathy, B. Lee, S. M. Drucker, and J. T. Stasko. Collecting and characterizing natural language utterances for specifying data visualizations. In Y. Kitamura, A. Quigley, K. Isbister, T. Igarashi, P. Bjørn, and S. M. Drucker, editors, *CHI '21: CHI Conference on Human Factors in Computing Systems, Virtual Event / Yokohama, Japan, May 8-13, 2021*, pages 464:1–464:10, 2021.

[101] A. Srinivasan and J. T. Stasko. Orko: Facilitating multimodal interaction for visual exploration and analysis of networks. *IEEE Trans. Vis. Comput. Graph.*, 24(1):511–521, 2018.

[102] A. Stöckl. Natural language interface for data visualization with deep learning based language models. In E. Banissi, A. Ursyn, M. W. M. Bannatyne, J. M. Pires, N. Datia, K. Nazemi, B. Kovalerchuk, R. Andonie, M. Nakayama, F. Sciarrone, W. Huang, Q. V. Nguyen, M. S. Mabakane, A. Rusu, M. Temperini, U. Cvek, M. Trutschl, H. Mueller, H. Siirtola, W. L. Woo, R. Francese, V. Rossano, T. D. Mascio, F. Bouali, G. Venturini, S. Kernbach, D. Malandrino, R. Zaccagnin, J. J. Zhang, X. Yang, and V. Geroimenko, editors, *26th International Conference Information Visualisation, IV 2022, Vienna, Austria, July 19-22, 2022*, pages 142–148. IEEE, 2022.

[103] A. Suhr, M. Chang, P. Shaw, and K. Lee. Exploring unexplored generalization challenges for cross-database semantic parsing. In D. Jurafsky, J. Chai, N. Schluter, and J. R. Tetreault, editors, *Proceedings of ACL 2020, Online, July 5-10, 2020*, pages 8372–8388, 2020.

[104] N. Sun, X. Yang, and Y. Liu. Tableqa: a large-scale chinese text-to-sql dataset for table-aware SQL generation. *CoRR*, abs/2006.06434, 2020.

[105] R. Sun, S. Ö. Arik, H. Nakhost, H. Dai, R. Sinha, P. Yin, and T. Pfister. Sql-palm: Improved large language model adaptation for text-to-sql. *CoRR*, abs/2306.00739, 2023.

[106] C. Tai, Z. Chen, T. Zhang, X. Deng, and H. Sun. Exploring chain-of-thought style prompting for text-to-sql. *CoRR*, abs/2305.14215, 2023.

[107] J. Tang, Y. Luo, M. Ouzzani, G. Li, and H. Chen. Sevi: Speech-to-visualization through neural machine translation. In Z. G. Ives, A. Bonifati, and A. E. Abbadi, editors, *SIGMOD '22: International Conference on Management of Data, Philadelphia, PA, USA, June 12 - 17, 2022*, pages 2353–2356. ACM, 2022.

[108] L. R. Tang and R. J. Mooney. Automated construction of database interfaces: Intergrating statistical and relational learning for semantic parsing. In H. Schütze and K. Su, editors, *Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora, EMNLP 2000, Hong Kong, October 7-8, 2000*, pages 133–141, 2000.

[109] 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 D. Jurafsky, J. Chai, N. Schluter, and J. R. Tetreault, editors, *Proceedings of ACL 2020, Online, July 5-10, 2020*, pages 7567–7578, 2020.

[110] C. Wang, P. Huang, A. Polozov, M. Brockschmidt, and R. Singh. Execution-guided neural program decoding. *CoRR*, abs/1807.03100, 2018.

[111] L. Wang, A. Zhang, K. Wu, K. Sun, Z. Li, H. Wu, M. Zhang, and H. Wang. Dusql: A large-scale and pragmatic chinese text-to-sql dataset. In B. Webber, T. Cohn, Y. He, and Y. Liu, editors, *Proceedings of EMNLP 2020, Online, November 16-20, 2020*, pages 6923–6935, 2020.

[112] P. Wang, T. Shi, and C. K. Reddy. Text-to-sql generation for question answering on electronic medical records. In Y. Huang, I. King, T. Liu, and M. van Steen, editors, *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, pages 350–361, 2020.

[113] D. H. D. Warren and F. C. N. Pereira. An efficient easily adaptable system for interpreting natural language queries. *Am. J. Comput. Linguistics*, 8(3-4):110–122, 1982.

[114] J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, D. Yogatama, M. Bosma, D. Zhou, D. Metzler, E. H. Chi, T. Hashimoto, O. Vinyals, P. Liang, J. Dean, and W. Fedus. Emergent abilities of large language models. *Trans. Mach. Learn. Res.*, 2022, 2022.

[115] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. H. Chi, Q. V. Le, and D. Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*, 2022.

[116] T. Xie, C. H. Wu, P. Shi, R. Zhong, T. Scholak, M. Yasunaga, C. Wu, M. Zhong, P. Yin, S. I. Wang, V. Zhong, B. Wang, C. Li, C. Boyle, A. Ni, Z. Yao, D. Radev, C. Xiong, L. Kong, R. Zhang, N. A. Smith, L. Zettlemoyer, and T. Yu. Unifiedskg: Unifying and multi-tasking structured knowledge grounding with text-to-text language models. In Y. Goldberg, Z. Kozareva, and Y. Zhang, editors, *Proceedings of EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 602–631, 2022.

[117] X. Xu, C. Liu, and D. Song. Sqlnet: Generating structured queries from natural language without reinforcement learning. *CoRR*, abs/1711.04436, 2017.

[118] N. Yaghmazadeh, Y. Wang, I. Dillig, and T. Dillig. Sqlizer: query synthesis from natural language. *Proc. ACM Program. Lang.*, 1(OOPSLA):63:1–63:26, 2017.

[119] P. Yin and G. Neubig. A syntactic neural model for general-purpose code generation. In R. Barzilay and M. Kan, editors, *Proceedings of ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 440–450, 2017.

[120] P. Yin, G. Neubig, W. Yih, and S. Riedel. Tabert: Pretraining for joint understanding of textual and tabular data. In D. Jurafsky, J. Chai, N. Schluter, and J. R. Tetreault, editors, *Proceedings of ACL 2020, Online, July 5-10, 2020*, pages 8413–8426, 2020.

[121] B. Yu and C. T. Silva. Visflow - web-based visualization framework for tabular data with a subset flow model. *IEEE Trans. Vis. Comput. Graph.*, 23(1):251–260, 2017.

[122] B. Yu and C. T. Silva. Flowsense: A natural language interface for visual data exploration within a dataflow system. *IEEE Trans. Vis. Comput. Graph.*, 26(1):1–11, 2020.

[123] T. Yu, Z. Li, Z. Zhang, R. Zhang, and D. R. Radev. Typesql: Knowledge-based type-aware neural text-to-sql generation. In M. A. Walker, H. Ji, and A. Stent, editors, *Proceedings of NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 2 (Short Papers)*, pages 588–594, 2018.

[124] T. Yu, C. Wu, X. V. Lin, B. Wang, Y. C. Tan, X. Yang, D. R. Radev, R. Socher, and C. Xiong. Grappa: Grammar-augmented pre-training for table semantic parsing. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*, 2021.

[125] T. Yu, M. Yasunaga, K. Yang, R. Zhang, D. Wang, Z. Li, and D. R. Radev. Syntaxsqlnet: Syntax tree networks for complex and cross-domain text-to-sql task. In E. Riloff, D. Chiang, J. Hockenmaier, and J. Tsujii, editors, *Proceedings of EMNLP 2018, Belgium, October 31 - November 4, 2018*, pages 1653–1663, 2018.

[126] T. Yu, R. Zhang, H. Er, S. Li, E. Xue, B. Pang, X. V. Lin, Y. C. Tan, T. Shi, Z. Li, Y. Jiang, M. Yasunaga, S. Shim, T. Chen, A. R. Fabbri, Z. Li, L. Chen, Y. Zhang, S. Dixit, V. Zhang, C. Xiong, R. Socher, W. S. Lasecki, and D. R. Radev. Cosql: A conversational text-to-sql challenge towards cross-domain natural language interfaces to databases. In K. Inui, J. Jiang, V. Ng, and X. Wan, editors, *Proceedings of EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 1962–1979, 2019.

[127] T. Yu, R. Zhang, K. Yang, M. Yasunaga, D. Wang, Z. Li, J. Ma, I. Li, Q. Yao, S. Roman, Z. Zhang, and D. R. Radev. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. In E. Riloff, D. Chiang, J. Hockenmaier, and J. Tsujii, editors, *Proceedings of EMNLP 2018, Brussels, Belgium, October 31 - November 4, 2018*, pages 3911–3921, 2018.

[128] T. Yu, R. Zhang, M. Yasunaga, Y. C. Tan, X. V. 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. R. Radev. Sparc: Cross-domain semantic parsing in context. In A. Korhonen, D. R. Traum, and L. Màrquez, editors, *Proceedings of ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4511–4523, 2019.

[129] J. M. Zelle and R. J. Mooney. Learning to parse database queries using inductive logic programming. In W. J. Clancey and D. S. Weld, editors, *Proceedings of the Thirteenth National Conference on Artificial Intelligence and Eighth Innovative Applications of Artificial Intelligence Conference, AAAI 96, IAAI 96, Portland, Oregon, USA, August 4-8, 1996, Volume 2*, pages 1050–1055, 1996.

[130] J. Zeng, X. V. Lin, S. C. H. Hoi, R. Socher, C. Xiong, M. R. Lyu, and I. King. Photon: A robust cross-domain text-to-sql system. In A. Celikyilmaz and T. Wen, editors, *Proceedings of ACL 2020, Online, July 5-10, 2020*, pages 204–214, 2020.

[131] R. Zhang, T. Yu, H. Er, S. Shim, E. Xue, X. V. Lin, T. Shi, C. Xiong, R. Socher, and D. R. Radev. Editing-based SQL query generation for cross-domain context-dependent questions. In K. Inui, J. Jiang, V. Ng, and X. Wan, editors, *Proceedings of EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 5337–5348, 2019.

[132] R. Zhong, T. Yu, and D. Klein. Semantic evaluation for text-to-sql with distilled test suites. In B. Webber, T. Cohn, Y. He, and Y. Liu, editors, *Proceedings of EMNLP 2020, Online, November 16-20, 2020*, pages 396–411, 2020.

[133] V. Zhong, C. Xiong, and R. Socher. Seq2sql: Generating structured queries from natural language using reinforcement learning. *CoRR*, abs/1709.00103, 2017.