TECHNISCHE UNIVERSITÄT ILMENAU
Fakultät für Informatik und Automatisierung

Master Thesis Exposé

A Review on State-of-the-art Text-To-SQL Solutions

presented by

Shahriar Yazdipour
Matrikel 62366

Supervisor:

M. Sc. Martin Hofmann
Prof. Dr.-Ing. Patrick Mäder

Ilmenau, July 23, 2022

Introduction

Data retrieval in databases is typically done using SQL (Structured Query Language). Text-to-SQL machine learning models are a recent development in state-of-the-art research. The technique is an attractive alternative for many natural language problems, including complex queries and extraction tasks. The text is converted into a SQL query that can be executed on the database. This technique can save time and effort for both developers and end-users by enabling them to interact with databases through natural language queries. With the help of machine learning and knowledge-based resources, text language to SQL conversion is facilitated.

Semantic parsing is a natural language processing that extracts the meaning from text. Text-to-SQL, a type of Semantic Parsing, is a task that converts natural language problems into SQL query statements. This is achieved using machine learning and natural language processing algorithms, and this research is conducted to study different solutions and practices which has been taken by researchers to tackle this problem.

Text-to-SQL allows the elaboration of structured data with information about the natural language text in several domains, such as healthcare, customer service, and search engines. It can be used by data analysts, data scientists, software engineers, and end users who want to explore and analyze their data without learning SQL. It can be used in a variety of ways:

1) Data analysts can use it to generate SQL queries for specific business questions, such as ”What are the top ten products sold this month?”

2) Data scientists can use it to generate SQL queries for machine learning experiments, such as ”How does the price of these products affect their sales?”

3) Businesses can use this technique to automate data extraction and improve efficiency.

4) End-users who want to explore and analyze their data without learning SQL can use it by clicking on a button on any table or chart in a user interface.

Although these models may not solve this problem entirely and perfectly, humans can still struggle with the task. For example, people involved in database migration projects often have to work on schema that they have never seen before.

This research study will review some of the most commonly used NLP technologies relevant to converting text language into Structured Query Language (SQL), and representative models and datasets in the recent solutions for this challenge and their technical implementation.

Motivation and State of the Art

Representative datasets for Text-to-SQL include the WikiSQL [**?**] dataset and the SPIDER [**?**] dataset, which contains more complex SQL queries. The former case consists of Single Table - Multiple Question and the latter case Multiple Table - Multiple Question. The top models available for Text-To-SQL will be studied, and the implementation of a couple of currently best methods will be reviewed and discussed in this study.

In this thesis, we will review the Text-to-SQL Challenges and datasets and structure of existing datasets and difference between them. Datasets to be covered are: ATIS, GeoQuery, IMDb, Advising, WikiSQL, Spider.

Datasets

ATIS (Air Travel Information System) Dataset

A relational schema is used to organize data from the official airline guide in the ATIS corpus. There are 25 tables containing information about fares, airlines, flights, cities, airports, and ground services. All questions related to this dataset can be answered using a single relational query. The relational database uses shorter tables for this dataset to answer queries intuitively.

Here is an example query from the ATIS dataset: Input is in natural language, and the output is in $\lambda calculus$.

Input description: list airport in ci0
Output λ-calculus:

```
lambda $0 e ( and ( airport $0 )
              ( loc:t $0 ci0 ) )
```

Figure 1: Example from ATIS dataset for semantic parsing

GeoQuery Dataset

United States geography is represented in the Geoquery dataset. About 800 facts are expressed in Prolog. State, city, river, and mountain information can be found in the database. Geographic and topographical attributes such as capitals and populations make up the majority of the attributes.

IMDb Dataset

The IMDb dataset contains 50K reviews from IMDb. There is a limit of 30 reviews per movie. Positive and negative reviews are equally represented in the dataset. The dataset creators considered a negative review with a score of 4 out of 10 and a positive review with a score of 7 out of 10. When creating the dataset, neural reviews are not taken into account. Furthermore, Training and testing datasets are equally divided.

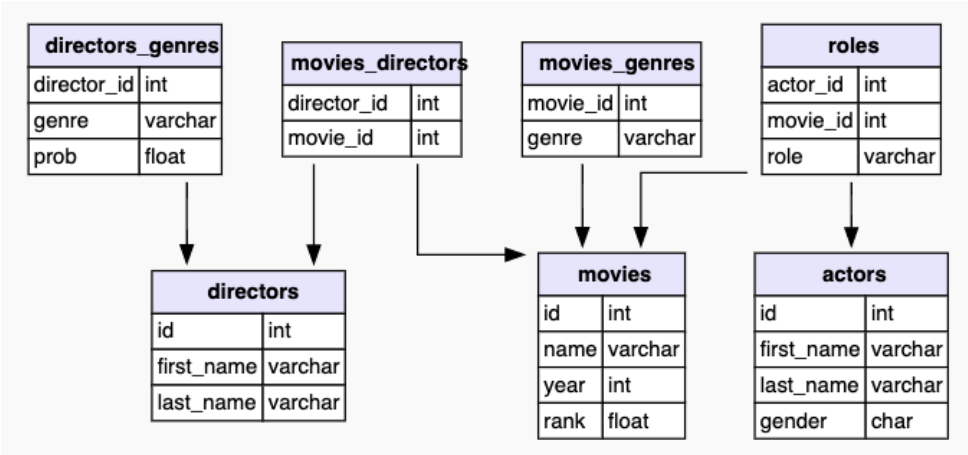


Figure 2: Database Structure of IMDb dataset

Advising Dataset

The Advising dataset was created in order to propose improvements in text2SQL systems. The creators of the dataset compare human-generated and automatically generated questions, citing properties of queries that relate to real-world applications. Dataset consists of questions from university students about courses that lead to particularly complex queries. The database contains fictional student records. The dataset includes student profile information, such as recommended courses, grades, and previous courses. In an academic advising meeting, students were asked to formulate questions they would ask if they knew the database. Many of the queries in this dataset were the same as those in ATIS, GeoQuery, and Scholar.

WikiSQL Dataset

WikiSQL consists of 80K+ natural language questions and corresponding SQL queries on 24K+ tables extracted from Wikipedia. Neither the train nor development sets contain the database in the test set. Databases and SQL queries have simplified the dataset's creators' assumptions. This dataset consists only of SQL labels covering a single SELECT column and aggregation and WHERE conditions. Furthermore, all the databases contain only one table.

The database does not include complex queries involving advanced operations like JOIN, GROUP BY, ORDER BY, etc. Prior to the release of SPIDER, this dataset was considered to be a benchmark dataset. Using WikiSQL has been the subject of a great deal of research. WikiSQL's "WHERE" clause has been recognized as one of the most challenging clauses to parse semantically, and SQLNet and SyntaxSQL were previous state-of-the-art models.

Spider Dataset

Yale University students created this dataset. The SPIDER database contains 10K questions and 5K+ complex SQL queries covering 138 different domains across 200 databases. As opposed to previous datasets (most of which used only one database), this one incorporates multiple datasets. Creating this corpus was primarily motivated by the desire to tackle complex queries and generalize across databases without requiring multiple interactions.

Creating a dataset involves three main aspects: SQL pattern coverage, SQL consistency, and question clarity. Several databases from WikiSQL are included in the dataset. The table is complex as it links

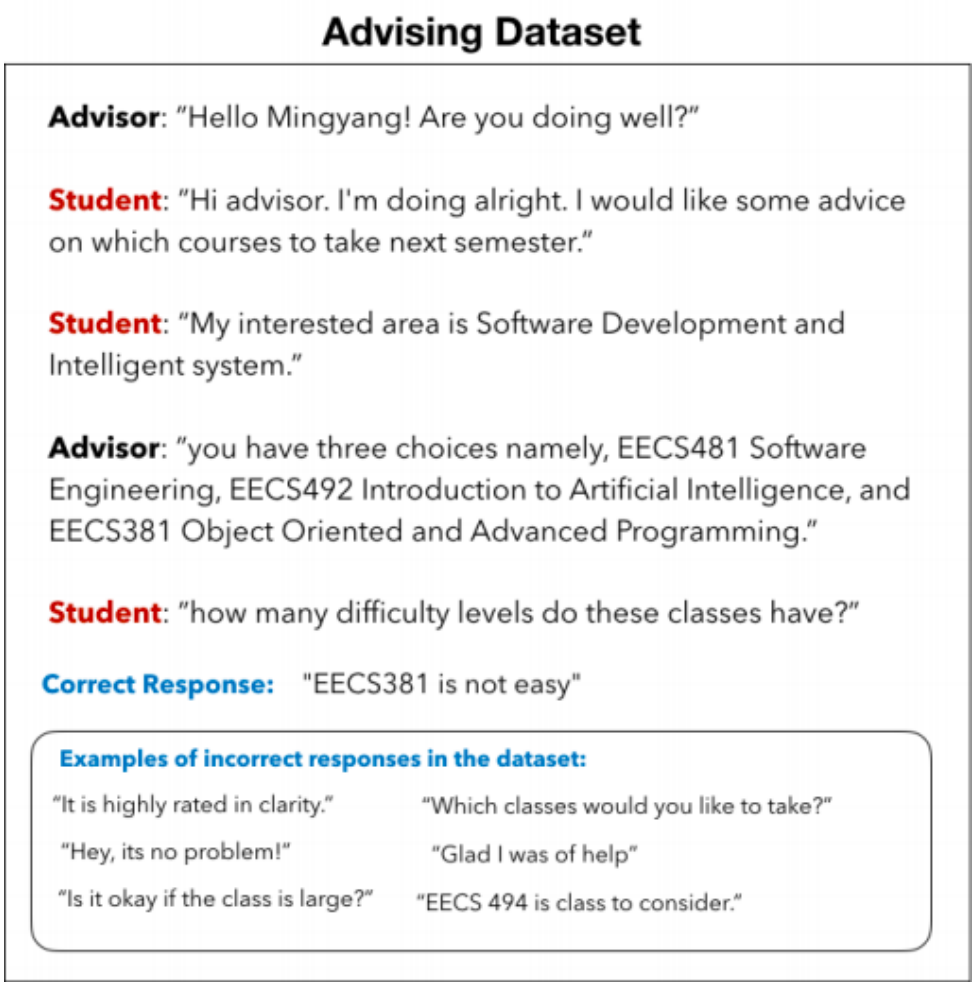


Figure 3: Example from Advising dataset

several tables with foreign keys. In SPIDER, SQL queries include: SELECT with multiple columns and aggregations, WHERE, GROUP BY, HAVING, ORDER BY, LIMIT, JOIN, INTERSECT, EXCEPT, UNION, NOT IN, OR, AND, EXISTS, LIKE.

SPIDER's exact matching accuracy was 12.4% compared to existing state-of-the-art models. As a result of its low accuracy, SPIDER presents a strong research challenge. Current SPIDER accuracy is around 66% with an exact set match without values (refers to values in the WHERE clause) and around 63% with values.

Models

The research section will assess the best state-of-the-art research in this field, the T5[**?**] Transformer with PICARD[**?**] (2021), RAT-SQL[**?**] (2019), BRIDGE v1 and v2[**?**] with BERT, HydraNet[**?**] (2020), some novel methods in recent years, and some older models like Seq2SQL[**?**] (2017) and SQLova[**?**] (2019).

- Seq2SQL and SQLNet will perform encoding using LSTM/Bi-LSTM and decode using classification and Pointer Network.
- SQLova and HydraNet encode natural language queries through a language model and decode

Table:

Player	Country	Points	Winnings ($)
Steve Stricker	United States	9000	1260000
K.J. Choi	South Korea	5400	756000
Rory Sabbatini	South Africa	3400	4760000
Mark Calcavecchia	United States	2067	289333
Ernie Els	South Africa	2067	289333

Question: What is the points of South Korea player?

SQL: `SELECT Points WHERE Country = South Korea`

Answer: 5400

Figure 4: Example from WikiSQL dataset

them through the Natural Language-to-SQL layer, converting them to SQL grammars. Going a little further, SQLova was the first to use a language model as an encoder, encode questions and columns, and then predict queries.

- HydraNet uses BERT's token to rank columns one by one to fill in the slots of the SQL Query statement. Brief description of the BERT will be given in our final report.
- Since the SPIDER dataset used for RAT-SQL and BRIDGE is a Multi-Table, it is necessary to understand the Schema's relationship to the question and its internal relationships, for which Schema Linking and Encoding are applied, and decoders such as SemQL are utilized.
- RAT SQL reflects the schema information into the encoding and decoding with SemQL, including the relationships extracted from the question-scheme contextualized graph in Self-Attention.
- BRIDGE achieves Schema Linking by including the Schema's Table and Column Name and Value in the Encoding and utilizes the Pointer Generator Network as the Decoder.

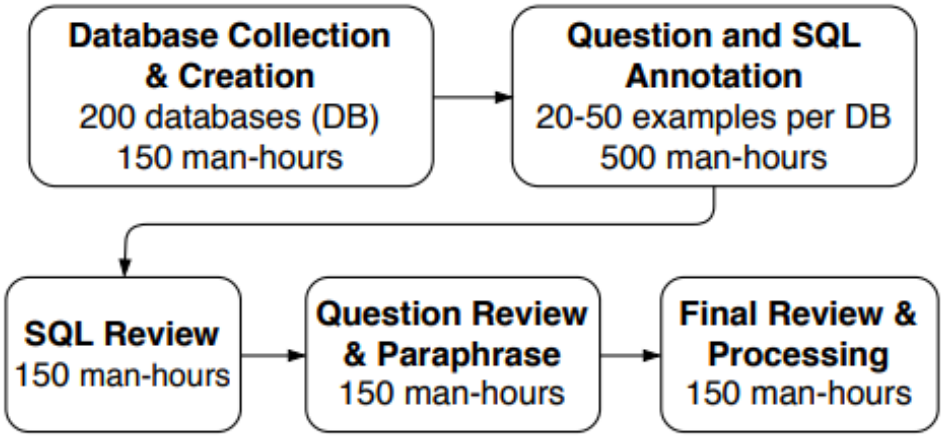


Figure 5: Example from Spider dataset

Complex question

What are the name and budget of the departments with average instructor salary greater than the overall average?

Complex SQL

```
SELECT T2.name, T2.budget
FROM instructor as T1 JOIN department as
T2 ON T1.department_id = T2.id
GROUP BY T1.department_id
HAVING avg(T1.salary) >
    (SELECT avg(salary) FROM instructor)
```

Figure 6: Example of Question-Query set from SPIDER

- PICARD proposes a new method for simple and effective constrained decoding with large pre-trained language models. On both the SPIDER cross-domain and cross-database Text-to-SQL dataset and the CoSQL SQL-grounded dialog state tracking dataset, we find that the PICARD decoding method not only significantly improves the performance of fine-tuned unmodified T5 models but it also lifts a T5-3B model to state-of-the-art results on the established exact-match and execution accuracy metrics.

After reviewing research papers of these models, we will study implementation steps of these models. And at last, we will use our private dataset to evaluate the accuracy of each models for our data and if they are usable and reliable enough for our usage.

Most of these studies have excellent documentation regarding their implantation. Execution of these studies will be documented and published on Github. Nonetheless, In case of old and impractical implementation instructions, we will skip the implementation and continue with the top models available.

SQLNet

- The model was designed to demonstrate that reinforcement learning should be limited in Text2SQL tasks.
- Until SQLNet, all previous models used reinforcement learning to improve the decoder results when it generated appropriate serializations.
- In cases where order is irrelevant, SQLNet avoids the seq2seq structure.
- For making predictions, the model uses a sketch-based approach consisting of a dependency graph that allows previous predictions to be taken into account.
- To improve the results, the model also incorporates column attention (weights assigned to significant words and phrases in sentences).
- According to the flowchart below, SQLNet employs three phases to generate SQL queries for WikiSQL tasks.

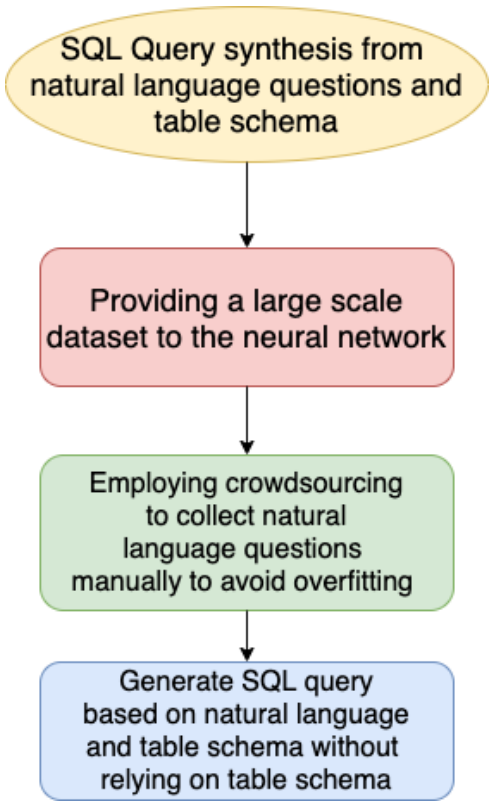
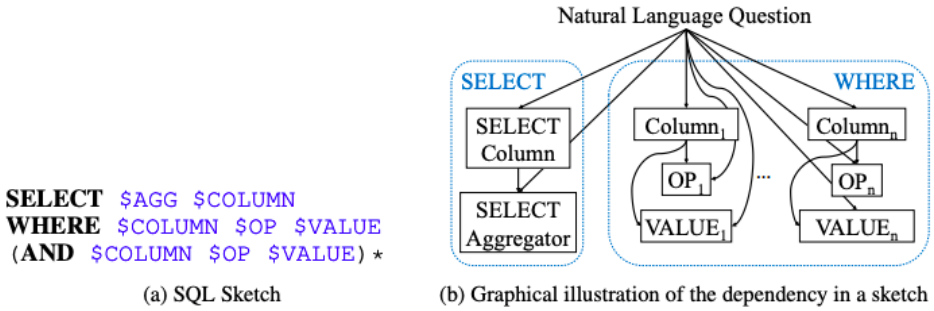


Figure 7: SQLNet

Sketch-based query synthesis

The token with the $ sign represents an empty slot, and the token name represents the type of prediction. Tokens in bold represent SQL keywords such as SELECT, WHERE, etc. $AGG can be filled with either an empty token or one of the aggregation operators, such as SUM or MAX. Fill in the $COLUMN and $VALUE slots with the column name and substring of the question, respectively. The $OP slot can be a value between {=, }. The notion ...* uses a regular expression to indicate zero or more AND clauses.



(a) SQL Sketch

(b) Graphical illustration of the dependency in a sketch

Figure 8: Sketch-based query synthesis

Column attention for sequence-to-set prediction

Instead of producing a sequence of column names, sequence-to-set prediction predicts the names of the columns of interest. Based on column names, column attention is part of the generic attention mechanism for computing the feature attention map on a question.

Predicting WHERE and SELECT clause

- One of the most challenging tasks in Text2SQL is predicting the WHERE clause.
- According to SQL sketch, SQLNet finds the columns that appear in the WHERE clause and predicts the OP slots and value for each column.
- It is predicted that the OP slot will be filled with one of the three classes ¡,¿,=, and the VALUE slot will be filled with the substring from the natural language question.
- In SELECT clauses, columns are named, and aggregator functions are specified, such as count, sum, max, etc. There is only one difference between SELECT and WHERE: the column name. There is only one column selected in SELECT.

  In the WikiSQL test set, SQLNet accuracy is 64.4

Table

Player	No.	Nationality	Position	Years in Toronto	School/Club Team
Antonio Lang	21	United States	Guard-Forward	1999-2000	Duke
Voshon Lenard	2	United States	Guard	2002-03	Minnesota
Martin Lewis	32, 44	United States	Guard-Forward	1996-97	Butler CC (KS)
Brad Lohaus	33	United States	Forward-Center	1996	Iowa
Art Long	42	United States	Forward-Center	2002-03	Cincinnati

Question:

Who is the player that wears number 42?

SQL:

SELECT player
WHERE no. = 42

Result:

Art Long

Figure 9: An example of a query executed by SQLNet on WikiSQL

SyntaxSQLNet

- The main goal of developing the SyntaxSQLNet model was to generate complex SQL queries with multiple clauses and generalize them to new databases.
- The model is based on a syntax tree network to address complex and cross-domain queries. The encoders are table-aware, and the decoders have a history of the SQL generation path.
- With a massive 7.3% improvement in accuracy, SyntaxSQLNet outperformed previous models, such as SQLNet, on the SPIDER dataset.

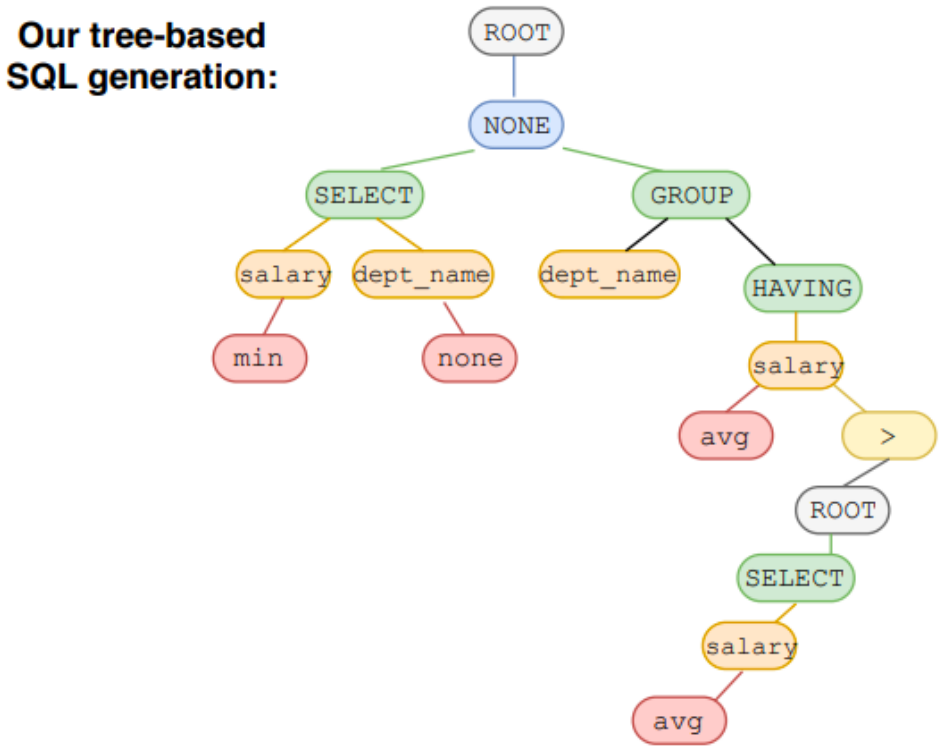


Figure 10: Tree-based SQL generator in SyntaxSQLNet

- A cross-domain data augmentation technique further improves accuracy by generating more variance during training.
- Below is a chart showing the various modules and their functions.

SQL Grammar and Attention Mechanism

- In order to enable the decoder to handle complex queries, SQL grammar is used. At each step of recursive decoding, it determines which module to invoke.
- Predicting the next SQL token is also based on the history of SQL path generation and current SQL tokens.
- The attention mechanism is also used to encode the question representation. Attention also applies to SQL path history encoding.

Data Augmentation

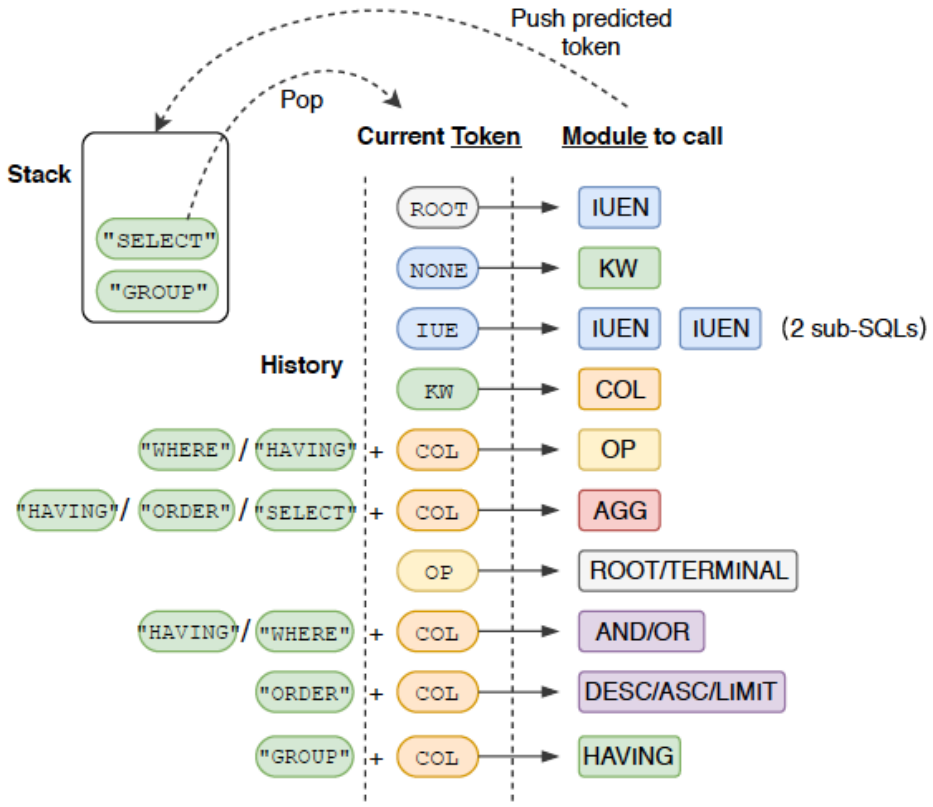


Figure 11: Modules defined in SyntaxSQLNet model

- Despite SPIDER's large dataset, it lacks complex queries.
- For proper generalization, cross-domain datasets are used for data augmentation.
- Various training databases of the SPIDER dataset are used to prepare a list of patterns for natural language questions and corresponding SQL queries.

The SPIDER model using syntaxSQLNet decoding history reaches 27.2% accuracy.

Compared to previous models, such as SQLNet, the accuracy increased by 15%.

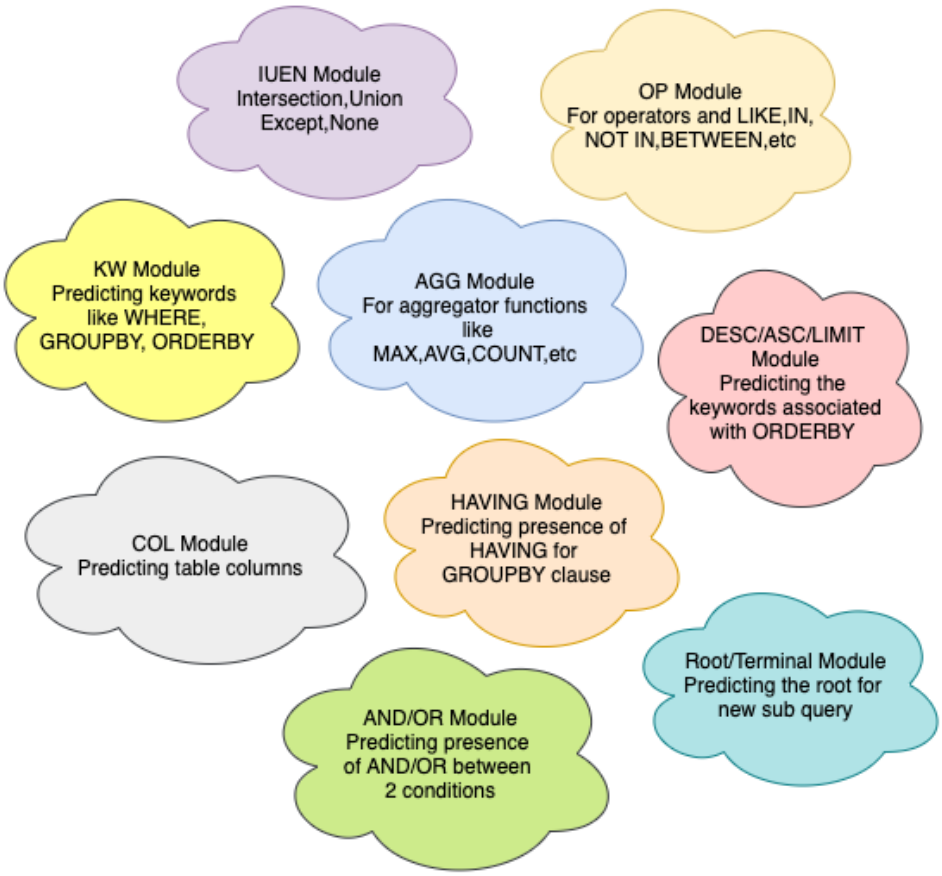


Figure 12: Modules and SQL Grammar used in the decoding process

GrammarSQL

Sequence-to-sequence models for neural text-to-SQL typically perform token-level decoding and do not consider generating SQL hierarchically.

3 proposes a grammar-based model for reducing the complexity of text2SQL tasks involving hierarchical grammars.

– The authors introduce schema-dependent grammar with minimal over-generation.

– The grammar developed in [3] covers 98

SQL Grammar

– The shallow parsing expression grammar aims to capture as little SQL as possible to cover most instances in the dataset.

– In order to ensure consistency of table, column, and value references in SQL, the authors added non-terminals to context-free grammar.

– They use runtime constraints during decoding to ensure that only valid programs can be used to join different tables in DB together using a foreign key.

Few details on the proposed model

– As an input, the proposed model takes an utterance of natural language, a database, and grammar about that utterance.

– String matching heuristics are applied after taking the input to link words in the input to identifiers or tokens in the database.

– Afterward, the bidirectional LSTM receives a concatenated string of the learned word and the link embeddings for each token.

– Using the attention mechanism, the decoder builds up the SQL query iteratively on the input sequence.

– Database identifiers in natural language questions and SQL queries are also anonymized.

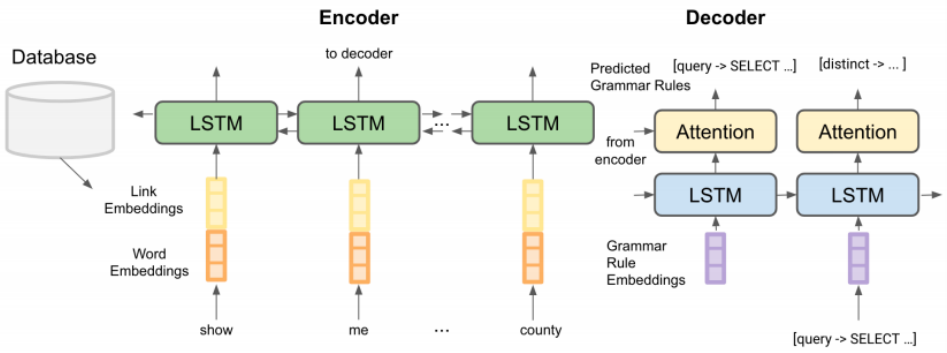


Figure 13: Structure of the proposed model

On ATIS and SPIDER datasets, the GrammarSQL model was evaluated. By 14

IRNet

- In Text2SQL tasks, the Intermediate Representation Network (IRNet) addresses two main challenges.
- Among the challenges are mismatches between natural language intents and predicting columns resulting from a more significant number of out-of-domain words.
- Instead of synthesizing SQL queries end-to-end, IRNet decomposes natural language into three phases.
- Schema linking is performed over a database schema and a question during the first phase.
- IRNet uses SemQL to bridge the gap between SQL and natural language.
- It includes a Natural Language (NL) encoder, a Schema Encoder, and a Decoder.

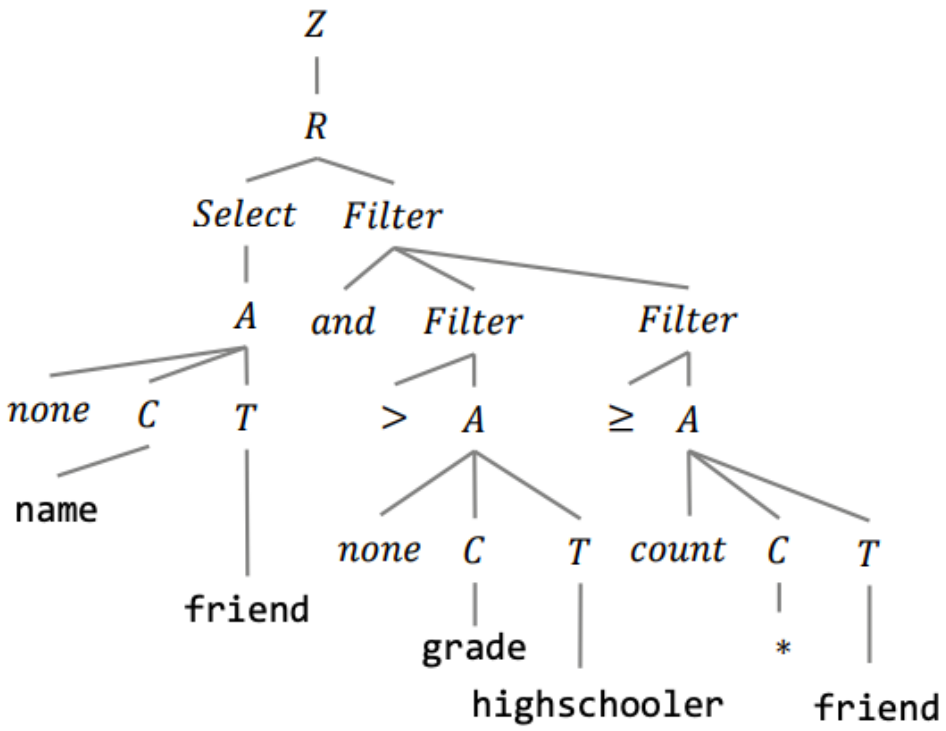


Figure 14: An illustrative example of SemSQL from [1]

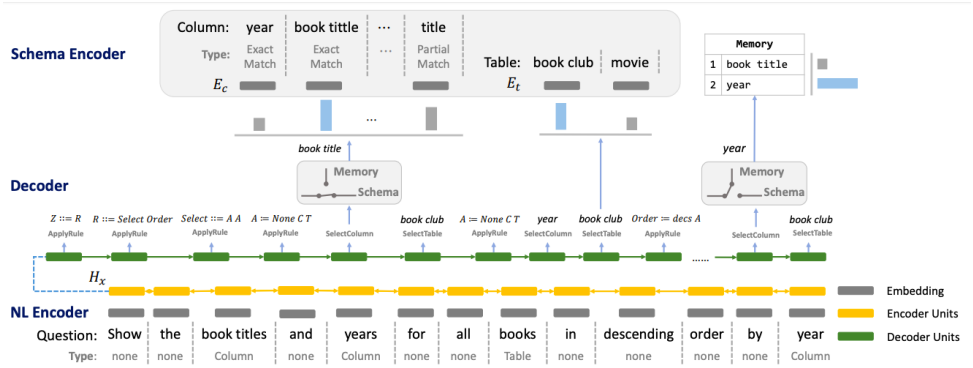


Figure 15: An overview of the neural model proposed in [1]

– The model provides different functions to accomplish Text2SQL tasks.
– Natural language is encoded into an embedding vector by the NL encoder. By using a bi-directional LSTM, these embedding vectors are used to construct hidden states.
– A schema encoder takes a database schema as input and outputs representations for columns and tables.
– Using a context-free grammar, the decoder synthesizes SemQL queries.
– On the SPIDER dataset, IRNet performs 46.7% better than previous benchmark models by 19
– The accuracy of 54.7% is achieved by combining IRNet with BERT.

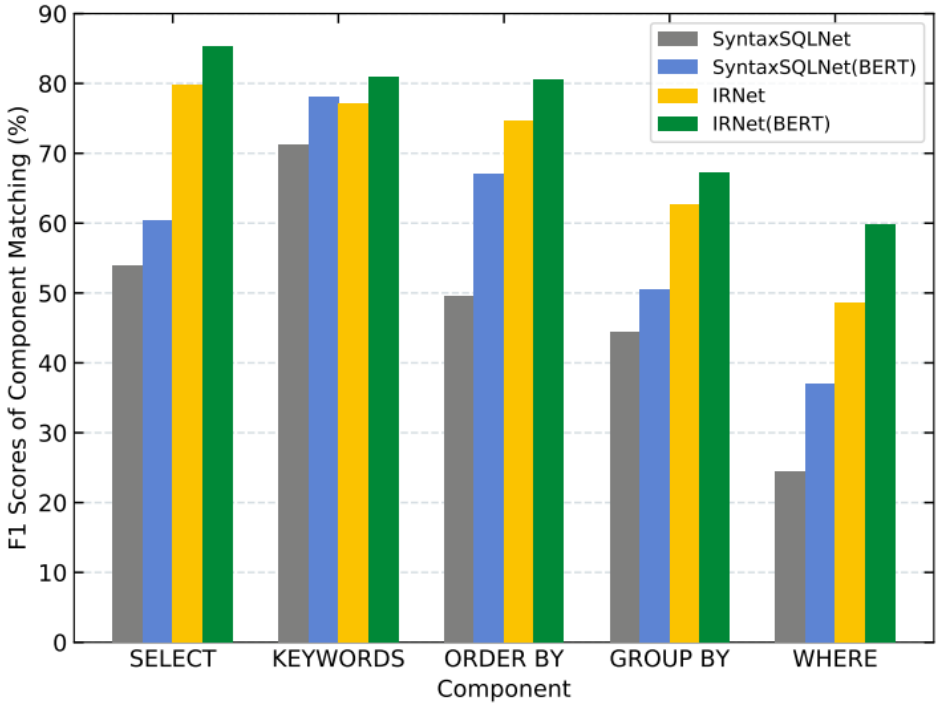


Figure 16: F1 scores of component matching of SyntaxSQLNet, SyntaxSQLNet(BERT), IRNet and IRNet(BERT) on the test set from [1]

EditSQL

- EditSQL focuses on text-to-SQL tasks that are context-dependent across domains.
- It exploits the fact that adjacent natural language questions are dependent on one another and that corresponding SQL queries overlap.
- To improve the generation quality, they edit the previously predicted query.
- The editing mechanism reuses generation results at the token level based on SQL input sequences.
- An utterance-table encoder and a table-aware decoder are utilized to incorporate the context of the natural language and the schema when dealing with complicated tables in different domains.

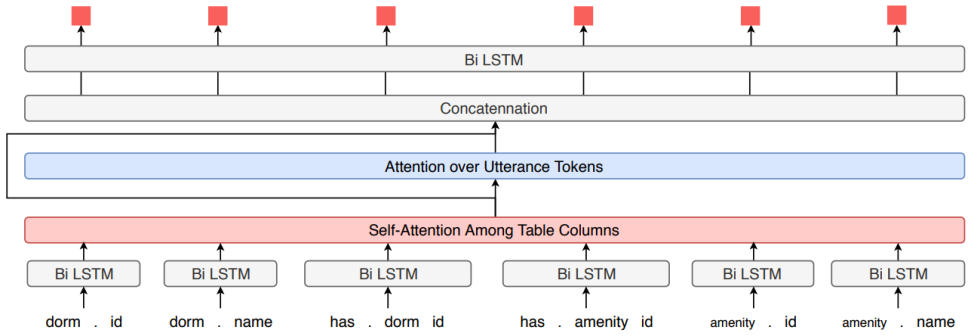


Figure 17: The model architecture of EditSQL from [2]

- User utterances and table schemas are encoded by the utterance-table encoder. Tokens of utterances are encoded using a bi-LSTM.
- To determine the most relevant columns, Attention weighed an average of column header embedding is applied to each token.

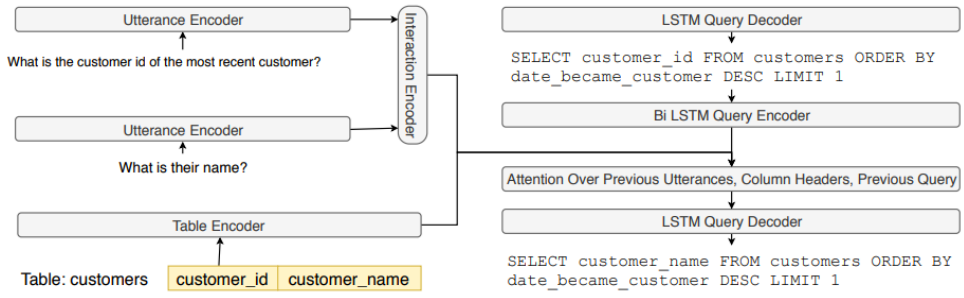


Figure 18: An example of user utterance and column headers and Utterance Encoder from [2]

- To capture the relationship between table schema and utterance, an attention layer is incorporated.
- The utterance-level encoder is built on top of an interaction-level decoder in order to capture information across utterances.
- LSTM decoding is used to generate SQL queries by incorporating interaction history, table schema, and user utterances.

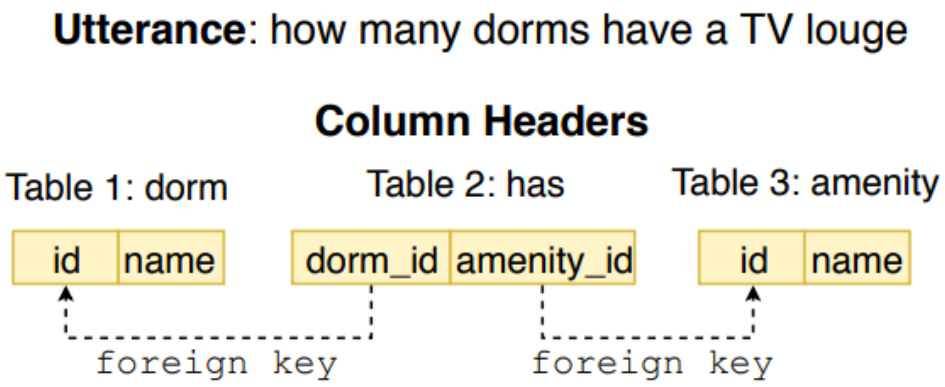


Figure 19: Table Encoder from [2]

- The model is evaluated on the SParC dataset, a large cross-domain context-dependent semantic parsing dataset derived from SPIDER.
- In both SPIDER and SParC, the model outperforms the previous state of the art model, IRNet.
- In cross-domain text2SQL generation, the model achieves 32.9% accuracy. A 53.4% improvement in accuracy can be achieved by using BERT embedding.

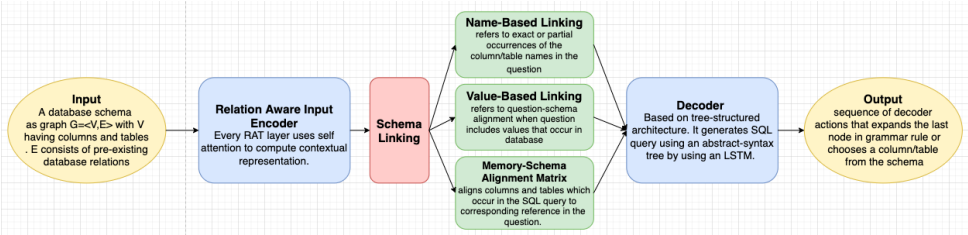


Figure 20: A flow chart of RAT-SQL model (Source: Author)

- A major challenge in translating natural language queries into SQL queries is generalizing them to unknown database schemas.
- As part of the generalisation, it is necessary to encode database relations in an accessible way and model alignment between relevant database columns in the query.
- Within a text2SQL encoder, the proposed framework leverages the relation-aware self-attention mechanism to encode address schemas, represent features, and link schemas.
- Check out the flow chart below for an overview of RAT-SQL's encoder-decoder structure.

Model	Dev	Test
IRNet (Guo et al., 2019)	53.2	46.7
Global-GNN (Bogin et al., 2019b)	52.7	47.4
IRNet V2 (Guo et al., 2019)	55.4	48.5
RAT-SQL (ours)	**62.7**	**57.2**
With BERT:		
EditSQL + BERT (Zhang et al., 2019)	57.6	53.4
GNN + Bertrand-DR (Kelkar et al., 2020)	57.9	54.6
IRNet V2 + BERT (Guo et al., 2019)	63.9	55.0
RYANSQL V2 + BERT (Choi et al., 2020)	**70.6**	60.6
RAT-SQL + BERT (ours)	69.7	**65.6**

Figure 21: Accuracy on the Spider development and test sets, compared to the other approaches at the top of the dataset leaderboard as of May 1st, 2020 from [3]

On the SPIDER dataset, RAT-SQL achieves 57.2% accuracy, an improvement of 8.7% over previous benchmark models.

With RAT-SQL, 65.6% accuracy can be achieved by combining BERT with RAT-SQL.

Split	Easy	Medium	Hard	Extra Hard	All
RAT-SQL					
Dev	80.4	63.9	55.7	40.6	62.7
Test	74.8	60.7	53.6	31.5	57.2
RAT-SQL + BERT					
Dev	86.4	73.6	62.1	42.9	69.7
Test	83.0	71.3	58.3	38.4	65.6

Figure 22: Accuracy on the Spider development and test sets, by difficulty from [3]

Outline

1. Introduction
    - Description and Motivation
    - Applications
    - Basic Research
2. Related Works and Background
    - Theoretical background and the review of world literature
    - Deep Learning and other approaches
3. Dataset, Implementation, and Results
    - Datasets and Challenges
    - Study Different Researches in Text-to-SQL SPIDER Challenge
        - Seq2SQL
        - RATSQL
        - T5 - PICARD
        - ...
    - Implementation Details
4. Summary and Future Work
    - Implement and test on private dataset
    - Discussion of the Results
    - Effect on other researches, like Text-to-SPARQL
    - Conclusions
5. Bibliography

Timeline

The timeline for the project will be broken down into four phases: literature review, establishing the theoretical framework and research implementation, using some datasets on a state-of-the-art model, and finally, documenting our study.

Phase One: Literature Review The first phase will be a review of any literature considerations. It includes examining how other researchers have approached the topic and a discussion of challenges around this research problem. Many considerations need to be addressed in this phase, such as the goal, the data, and how researchers find solutions for such a task.

Phase Two: Theoretical Framework and Research implementation The second phase is where A theoretical framework of a model will be established and implemented. In order to do this, a variety of different approaches will be explored, which are possible to implement with our hardware and legal license limitations.

Phase Three: Using private data on our model Here, the previously collected dataset will be used to test our model and validate the accuracy of the final result.

Phase Four: Documenting our study. It will be planned to finish a draft regarding our study and our outcome in a month, and getting prepared for thesis defense.

The following timeline is a rough draft of what I would like to do for this project. It is not a complete timeline, and it is subject to change.

Week Number	Month	Task Detail
22-23	October 2022	Lorem ipsum dolor sit amet, consectetuer adipiscing elit.
12	November 2022	Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis.
12	December 2022	Curabitur dictum gravidamauris.
12	January 2023	Curabitur dictum gravidamauris.
12	February 2023	Master's Thesis Defense