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

by: Shahriar Yazdipour

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

Submission Date: 10.05.2023
Defense Date: 16.06.2023

ABOUT ME

```
# About Me

– NAME: Shahriar Yazdipour

– RCSE Master Student @TU_Ilmenau
– Software Engineer @UBISOFT

– Matriculation Number: 62366

– https://yazdipour.com/
– shahriar.yazdipour@tu-ilmenau.de
```

CONTENTS
OUTLINE

Introduction
& Research Question

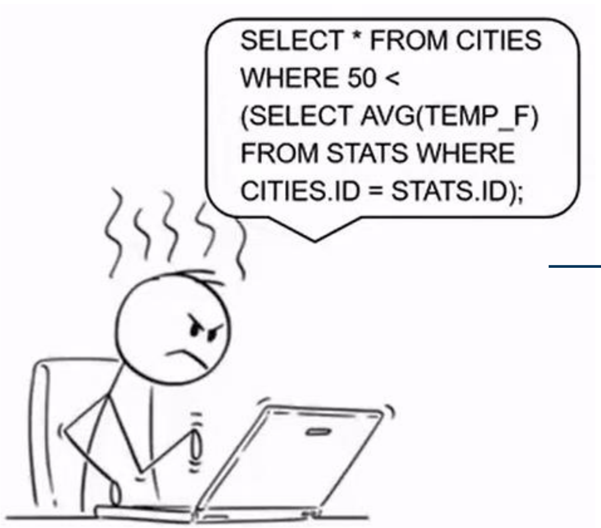
Introduction & Research Question

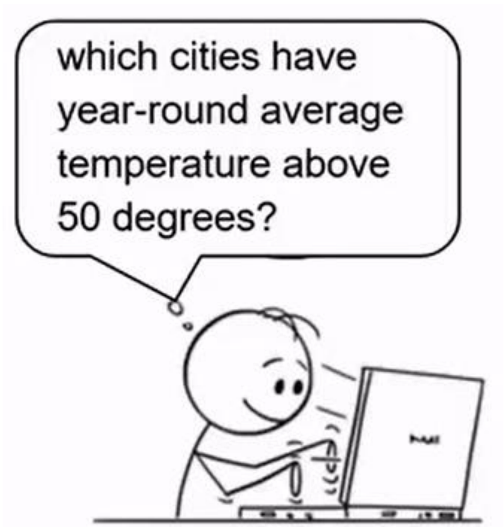
- SQL is the typical method for data retrieval in databases.

Introduction & Research Question

- SQL is the typical method for data retrieval in databases.

- Text-to-SQL allows for natural language interaction with structured data across various domains.



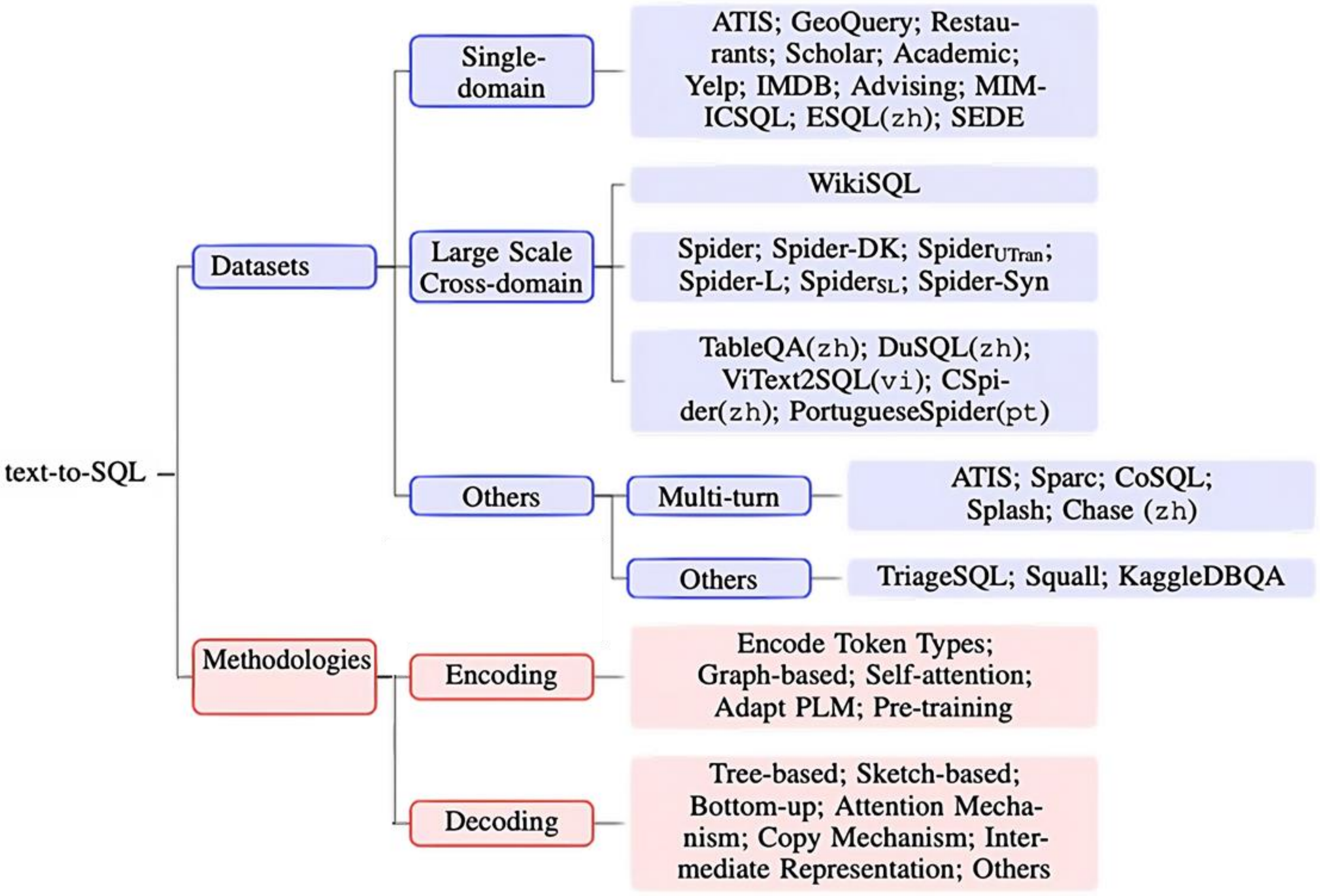


Introduction & Research Question

- SQL is the typical method for data retrieval in databases.
- Text-to-SQL allows for natural language interaction with structured data across various domains.
- We explores various NLP technologies used in Text-To-SQL and study "How can we effectively apply NLP technologies to convert natural language text into SQL, overcoming the existing challenges in the field?"

Challenges

- Encoding: Extract the meaning of NL
- Translation: Transform the extracted meaning into another expression which is pragmatically equivalent to the NL meaning
- Decoding: Produce the corresponding SQL queries


text-to-SQL
Datasets
Single-domain
ATIS; GeoQuery; Restaurants; Scholar; Academic; Yelp; IMDB; Advising; MIMICSQL; ESQL(zh); SEDE
Large Scale Cross-domain
WikiSQL
Spider; Spider-DK; SpiderUTran; Spider-L; SpiderSL; Spider-Syn
TableQA(zh); DuSQL(zh); ViText2SQL(vi); CSpider(zh); PortugueseSpider(pt)
Others
Multi-turn
ATIS; Sparc; CoSQL; Splash; Chase (zh)
Others
TriageSQL; Squall; KaggleDBQA
Methodologies
Encoding
Encode Token Types; Graph-based; Self-attention; Adapt PLM; Pre-training
Decoding
Tree-based; Sketch-based; Bottom-up; Attention Mechanism; Copy Mechanism; Intermediate Representation; Others

Datasets

Datasets

- Contain mainly: List of Utterances with their equivalent SQL queries
- Provided by companies, universities or communities

- Vary in
  - Complexity
  - Size
  - Annotation
- They provide a standardized testbed for evaluating the performance of Text-to-SQL models

Classified into three categories:
- Single domain datasets
- Cross domain datasets
- Others

Benchmark Dataset: Single domain datasets

- Valuable specially for Real-life single domain knowledge
- Early Datasets are limited in size

Datasets	#Size	#DB	#D	#T/DB	Issues addressed	Sources for data
IMDB (Yaghmazadeh et al., 2017)	131	1	1	16	-	Internet Movie Database
Yelp (Yaghmazadeh et al., 2017)	128	1	1	7	-	Yelp website
Advising (Finegan-Dollak et al., 2018)	3,898	1	1	10	-	University of Michigan course information
MIMICSQL (Wang et al., 2020d)	10,000	1	1	5	-	Healthcare domain
SEDE (Hazoom et al., 2021)	12,023	1	1	29	SQL template diversity	Stack Exchange

Benchmark Dataset:
Cross domain datasets

- NL-SQL pairs span across multiple databases and domains.
- Usually large in size
- Test the generalizability of models
- Some are generated datasets

Datasets	#Size	#DB	#D	#T/DB	Issues addressed	Sources for data
Spider (Yu et al., 2018c)	10,181	200	138	5.1	Domain generalization	College courses, DabaseAnswers, WikiSQL
WikiSQL (Zhong et al., 2017)	80,654	26,521	-	1	Data size	Wikipedia

Benchmark Dataset: Other

- Multilingual
  - e.g: Cspider
- Conversational Datasets:
  - e.g: CoSQL

U_1: List the name of the teachers and the courses assigned for them to teach.

S_1 :
```
SELECT T3.Name, T2.Course FROM course_arrange AS T1
JOIN course AS T2 ON T1.Course_ID = T2.Course_ID
JOIN teacher AS T3 ON T1.Teacher_ID = T3.Teacher_ID
```

U_2: Arrange this list with the teachers name in ascending order.

S_2 :
```
SELECT T3.Name, T2.Course FROM course_arrange AS T1
JOIN course AS T2 ON T1.Course_ID = T2.Course_ID
JOIN teacher AS T3 ON T1.Teacher_ID = T3.Teacher_ID
ORDER BY T3.Name
```

U_3: Include teachers id in the same list.

S_3 :
```
SELECT T3.Name, T2.Course, T1.teacher_ID FROM course_arrange AS T1
JOIN course AS T2 ON T1.Course_ID = T2.Course_ID
JOIN teacher AS T3 ON T1.Teacher_ID = T3.Teacher_ID
ORDER BY T3.Name
```

Datasets

Dataset	Year	DBs	Tables	Utterances	Queries	Domain
ATIS	1994	1	32	5280	947	Air Travel Information
GeoQuery	2001	1	6	877	247	US geography database
Academic	2014	1	15	196	185	MicrosoftAcademicSearch
IMDB	2015	1	16	131	89	Internet Movie Database
Scholar	2017	1	7	817	193	Academic Publications
Yelp	2017	1	7	128	110	Yelp Movie Website
WikiSQL	2017	26,521	26,521	80,654	77,840	Wikipedia
Advising	2018	1	10	3,898	208	Student Course Info
Spider	2018	200	1,020	10,181	5,693	138 Different Domains
SEDE	2021	1	29	12,023	11,767	Stack Exchange
SEOSS	2022	1	13	1,162	116	Project ITS and VSC

Why not WikiSQL?

- WikiSQL: A crowdsourced dataset containing 18,000 natural language and SQL pairs, gathered from Wikipedia tables

- Issues with WikiSQL:
  - Simplicity
  - Many mistakes
  - Research suggests that the upper bound has been reached
  - Human accuracy estimated at 88%

SPIDER

- 10,000 questions
- 5,000+ complex SQL queries
- 138 different domains
- 200 databases
- It is unique in that it incorporates multiple datasets, unlike previous datasets that mostly used only one database
- Include SELECT, WHERE, GROUP BY, HAVING, ORDER BY, LIMIT, JOIN, INTERSECT, EXCEPT, UNION, NOT IN, OR, AND, EXISTS, and LIKE.

79 Sep 20, 2018	SQLNet *Shanghai Jiao Tong University (modified by Yale)* (Xu et al., '18) code	12.4
80 Sep 20, 2018	TypeSQL *Yale University* (Yu et al., NAACL '18) code	8.2
81 Sep 20, 2018	Seq2Seq + attention *University of Edinburgh (modified by Yale)* (Dong and Lapata, ACL '16) code	4.8

SPIDER

Easy

What is the number of cars with more than 4 cylinders?

```
SELECT COUNT(*)
FROM cars_data
WHERE cylinders > 4
```

Meidum

For each stadium, how many concerts are there?

```
SELECT T2.name, COUNT(*)
FROM concert AS T1 JOIN stadium AS T2
ON T1.stadium_id = T2.stadium_id
GROUP BY T1.stadium_id
```

Hard

Which countries in Europe have at least 3 car manufacturers?

```
SELECT T1.country_name
FROM countries AS T1 JOIN continents
AS T2 ON T1.continent = T2.cont_id
JOIN car_makers AS T3 ON
T1.country_id = T3.country
WHERE T2.continent = 'Europe'
GROUP BY T1.country_name
HAVING COUNT(*) >= 3
```

Extra Hard

What is the average life expectancy in the countries where English is not the official language?

```
SELECT AVG(life_expectancy)
FROM country
WHERE name NOT IN
   (SELECT T1.name
    FROM country AS T1 JOIN
    country_language AS T2
    ON T1.code = T2.country_code
    WHERE T2.language = "English"
      AND T2.is_official = "T")
```

SEOSS (Software Engineering in Open-Source Systems)

- Rich collection of NL expressions and corresponding Query.
- 166 uniquely phrased questions.
- Based on real scenarios from Apache Pig project.
- Expressions adapted from literature to match Apache Pig's ITS and VCS data.

- Dataset is divided into two categories:
    - 'Development' (81 queries)
    - 'Research' (63 queries)

- Enhanced with 22 records from stakeholder comments across 33 Apache projects.

- By Mr. Tomova and Mr. Hofmann and Prof. Mäder in TU Ilmenau

Methods

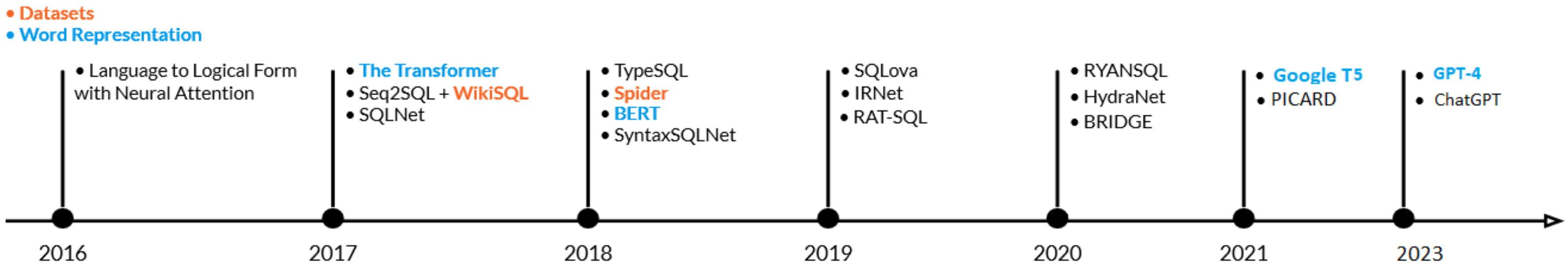
Methods

- **Early Text-to-SQL approaches** relied on rule-based and template-based methods.
  - Limitations: inability to handle complex queries and variations in natural language inputs.

- **Recent approaches** to Text-to-SQL have focused on using neural networks and machine learning techniques.
  - Needs large amounts of training data to learn the relationship between natural language and SQL.
  - Leverage pre-trained models.

Natural Language Representation

- Text Processing in NLP involves several steps:
  - Tokenization: Byte Pair Encoding (BPE), …
  - Embedding: GloVe, Word-Piece Embedding, …
  - Prediction
- Transformers
- The rise of Language Models
  - BERT
  - TaBERT
  - ….



T5-PICARD

T5 (Text-To-Text Transfer Transformer)

- Leverages the BERT encoder-decoder architecture.
- Using C4 Dataset
- Pre-training: Text corruption by randomly dropping out words, replaced with unique sentinel tokens.
- Highly adaptable, can be fine-tuned for a range of tasks

PICARD (Parsing Incrementally for Constrained Auto-Regressive Decoding)

- Challenges in Text-to-SQL:
  1. SQL must be valid
  2. SQL must correctly represent the meaning of the question

- Common solution for 1st challenge:

  Special constrained decoder architecture that always produces valid SQL which it needs to be trained from scratch and it limits generality, at odds with 2nd challenge

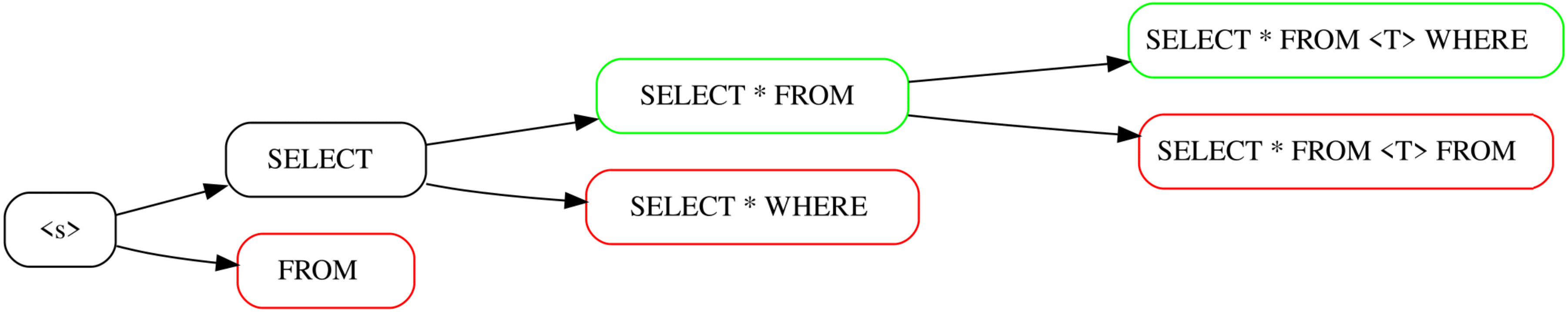
- PICARD solution:
  - Constrain decoder to produce valid SQL
  - Use existing pre-trained model
  - Incremental Parsing enforces syntactic and semantic constraints

Constrained Decoding

- Improves quality of outputs by imposing constraints or using auxiliary models during decoding.

- PICARD uses
  - Constraints to avoid generating invalid tokens.
  - Using Attoparsec parsing library that supports incremental input.
  - If the parsing is successful, it will return the final AST. If not, it will return a failure value.

Incremental Constrained Beam Search



PICARD on Spider

System	Development EM%	EX%
T5-Base (ours)	57.2	57.9
T5-Base+PICARD	65.8	68.4
T5-Large	65.3	67.2
T5-Large+PICARD	69.1	72.9
T5-3B (ours)	69.9	71.4
T5-3B+PICARD	74.1	76.3

Using T5-3B
Without PICARD: 12% invalid SQL predictions on Spider
With PICARD: 2% invalid predictions

Evaluation

Exact Set Matching

- Compares predicted SQL queries with reference queries, disregarding the order of elements.
- If the set of predicted queries is entirely in the reference query, the score is 1.0 else it's 0.0.
- Unlike "Exact Matching", which requires a perfect match in terms of element order, capitalization, and spaces, "Exact Set Matching" is more forgiving.

Evaluation

Execution Accuracy

- Measures the percentage of correctly generated SQL queries that execute successfully on a database.
- Higher values denote better performance, with this metric focusing on syntactic and semantic correctness of the generated queries.
- As "Execution Accuracy" does not assess the relevance or comprehensiveness of the returned data, it's often combined with other metrics

Experiments & Findings

SEOSS + T5-PICARD

SEOSS evaluation with T5 PICARD

- Experiment conducted using SEOSS dataset and the PICARD model.
- Comparison made with SQLNet and RatSQL.
- T5-Base and T5-Large models used for the experiment.
- Goal: To see if PICARD can achieve similar results to SQLNet and RatSQL.

Model	Picard Mode	Beams	**Exact Matching Accuracy**	**Execution Accuracy**
T5-base	lex	4	0.3071	0.3039
T5-base	parse with guards	2	0.3297	0.3576
T5-base	parse with guards	4	0.3286	0.3512
T5-large	lex	2	0.3672	0.3629
T5-large	parse with guards	4	0.4274	0.4822

SEOSS evaluation with T5 PICARD

- Experiment conducted using SEOSS dataset and the PICARD model.
- Comparison made with SQLNet and RatSQL.
- T5-Base and T5-Large models used for the experiment.
- Goal: To see if PICARD can achieve similar results to SQLNet and RatSQL.

Exact Match Accuracy	easy 392	medium 378	hard 77	extra hard 84	all 931
SQLNet	0.023	0.000	0.000	0.000	0.010
RatSQL + Glove	0.309	0.214	0.091	0.000	0.224
RatSQL + Bert	0.161	0.201	0.065	0.012	0.156
PICARD-T5Base + 4Beam	0.446	0.254	0.182	0.012	0.307
PICARD-T5Large + 4Beam	0.571	0.410	0.182	0.060	0.427

SEOSS + ChatGPT

SEOSS evaluation with ChatGPT

- The Generative Pre-trained Transformers (GPT) architecture has significantly influenced natural language processing with self-attention mechanisms, allowing efficient training and superior NLP task performance.
- GPT models are inherently generative, designed to create text based on provided context.
- They are pre-trained on a vast volume of text data to learn structure and patterns in natural language.
- Number of parameters:
  - GPT-1 : 117 million
  - GPT-2 :1.5 billion
  - GPT-3 :175 billion
  - GPT-4 has not been officially disclosed. (rumor 170 trillion)

Prompt Engineering

- Prompt: Open-ended Instruction provided by user to LLM
- Prompt Engineering is an emerging field aimed at enhancing the performance of large language models (LLMs) across diverse applications by refining prompts.
- Mastering prompt engineering enables deeper insights into the potential of LLMs and augments their proficiency in handling various tasks.
- With OpenAI API you can use different AI model versions with custom parameters, for your specific task.

Prompt Engineering
Roles:

- **System**: Offers guidance or establishes context for the assistant in the conversation.

- **User**: Usually a question or request submitted by the user for the assistant to process.

- **Assistant**: Processes the messages from the system and user in their order and generates a response. System instructions can guide the assistant's behavior in the conversation.
    - The assistant prompt is used to provide historical context.

ChatGPT3.5-Turbo Prompt Attempt#1

Prompt:

```
Generate syntactically correct read-only SQL to answer the following
question: What was changed in version 0.12.1
```

ChatGPT:

```
Certainly! Here's the SQL statement to return what was changed in version
0.12.1:
SELECT changes FROM version_history WHERE version_number = '0.12.1'.
Note that the syntax may vary slightly depending on the specific database
management system being used and the structure of your tables,
but this should work in most cases assuming there is a table named
version_history with columns version_number and changes.
```

role(System):

```
You are a helpful text-to-sql assistant for generating syntactically
correct read-only
SQL to answer a given question.
Database: concert_singer
The following are tables you can query:
--------------
table name: stadium table columns: Stadium_ID [number (13)], Location
[text (LA)], Name [text (SLA)], Capacity [number (30000)], Highest
[number (20000)], Lowest [number (100)], Average [number (1000)]
table name: singer table columns: Singer_ID [number (1)], NName
[text (John Doe)],Country [text (USA)],Song_Name [text (Beautiful
Day)],Song_release_year [text (2020)],Age [number (30)],Is_male
[others (Yes)] table name: concert table columns: concert_ID
[number (101)],concert_Name [text (Rock Night)],Theme [text (Rock
Music)],Stadium_ID [text (13)],Year [text (2023)] table name:
singer_in_concert table columns: concert_ID [number (101)], Singer_ID
[text (1)]
--------------
Do not use IN keyword.
If it is necessary to use AS then use it like T1 T2 ..., but if the alias
name is not going to be used in query again, then do not use.
Do not filter WHERE for being NOT NULL if it is not necessary.
If in using COUNT(*) and COUNT(COLUMN) there is no difference then use
COUNT(*).
Write one valid SQL in markdown format.
```

role(User):

```
Generate syntactically correct read-only SQL to answer the following
question: How many singers do we have?
```

role(Assistant):

```
SELECT count(*) FROM singer
```

SEOSS Accuracy Comparison

Exact Match Accuracy	easy 392	medium 378	hard 77	extra hard 84	all 931
SQLNet	0.023	0.000	0.000	0.000	0.010
RatSQL + Glove	0.309	0.214	0.091	0.000	0.224
RatSQL + Bert	0.161	0.201	0.065	0.012	0.156
PICARD-T5Base + 4Beam	0.446	0.254	0.182	0.012	0.307
PICARD-T5Large + 4Beam	0.571	0.410	0.182	0.060	0.427
GPT 3.5-turbo	0.719	0.571	0.403	0.226	**0.589**
GPT 4	0.727	0.571	0.403	0.226	**0.592**

	easy	medium	hard	extra hard	all
	for all utterances				
count	35	98	21	35	189
RatSQL + Glove	0.743	0.357	0.619	0.143	0.418
RatSQL + Bert	0.743	0.337	0.143	0.114	0.349
GPT 3.5-turbo	0.800	0.531	0.429	0.171	0.503
	for only non-specific utterances				
count	15	42	9	15	81
RatSQL + Glove	0.533	0.190	0.667	0.067	0.284
RatSQL + Bert	0.533	0.143	0.222	0.000	0.198
GPT 3.5-turbo	0.600	0.357	0.333	0.133	0.358
	for only specific utterances				
count	20	56	12	20	108
RatSQL + Glove	0.900	0.482	0.583	0.200	0.519
RatSQL + Bert	0.900	0.482	0.083	0.200	0.463
GPT 3.5-turbo	0.950	0.661	0.500	0.150	0.602

Table 13: Comparison between Exact Match Accuracy on 20% untrained queries

	for all utterances				
	easy	medium	hard	extra hard	all
count	112	108	22	24	266
RatSQL + Glove	0.866	0.806	0.591	0.333	0.771
RatSQL + Bert	0.732	0.574	0.364	0.083	0.579
GPT 3.5-turbo	0.950	0.661	0.500	0.150	0.602
	for only non-specific utterances				
count	56	54	11	12	133
RatSQL + Glove	0.839	0.704	0.636	0.250	0.714
RatSQL + Bert	0.607	0.389	0.364	0.000	0.444
GPT 3.5-turbo	0.652	0.593	0.318	0.167	0.556
	for only specific utterances				
count	56	54	11	12	133
RatSQL + Glove	0.893	0.907	0.545	0.417	0.827
RatSQL + Bert	0.857	0.759	0.364	0.167	0.714
GPT 3.5-turbo	0.750	0.685	0.545	0.333	0.669

Table 14: Comparison between Exact Match Accuracy on balanced utterances on trained RatSQL vs base

Cost/Resource

Model	Usage
GPT-4	$0.06 / 1K tokens
GPT-3.5-turbo	$0.002 / 1K tokens

As a rough rule of thumb, 1 token is approximately
4 characters or 0.75 words for English text.
(ChatGPT-3.5 has 2048 tokens limit, so its roughly 8192 char including roles and history)

Cost/Resource in SEOSS Experiment

Model	**Execution Accuracy**	**Time**	**Parameters**	**Cost**
PICARD + T5Base	0.307	400min	220M	Local Hardware
PICARD + T5Large	0.427	720min	770M	Local Hardware
GPT 3.5-turbo	0.447	37min	175B	$2/iteration
GPT 4	0.524	78min	1T	$14/iteration

SEOSS Failed Cases

[Easy] Utterance:

`Give me the count of all open issues.`

Predicted:

```
SELECT COUNT(*) FROM issue WHERE status != 'Closed'
```

Target:

```
SELECT Count(*) FROM issue WHERE status = 'Open'
```

Possible Fail reason:

`This is not consistent but the model is probably confused by the word "all" and counts "In Progress" and "Open" issues as well.`

SEOSS Failed Cases

[Medium] Utterance:
What are the ids and statuses of bugs from the Apache Pig project
Predicted:
```
SELECT issue_id, status FROM issue WHERE type = 'Bug' AND project =
'Apache Pig'
```
Target:
```
SELECT issue_id, status FROM issue WHERE type = 'Bug'
```
Possible Fail reason:
A tricky question lead to an extra condition [project = 'Apache Pig'] in the query, which is not valid since there is no project column in the issue table.

SEOSS Failed Cases

[Hard] Utterance:

How many issues of type bug and type improvement?

Predicted:

```
SELECT COUNT(*) FROM issue WHERE type IN ('Bug', 'Improvement')
```

Target:

```
SELECT type, Count(*) FROM issue GROUP BY type HAVING type = 'Bug' OR
type = 'Improvement'
```

Possible Fail reason:

This utterance could have two interpretations. Unfortunately, the model chose the wrong one, but it is syntactically and semantically correct.

SEOSS Failed Cases

[Extra Hard] Utterance:

Count the issue ids of issues that are of status Öpenğrouping them by assignee and then give me the assignee and the count of the issue ids of the assignee with most issues assigned to him/her.

Predicted:

```
SELECT issue.assignee, COUNT(issue.issue_id) AS num_assigned_issues
FROM issue WHERE issue.status = ’Open’ GROUP BY issue.assignee HAVING
COUNT(issue.issue_id) = ( SELECT MAX(num_assigned_issues) FROM (
SELECT COUNT(issue.issue_id) AS num_assigned_issues FROM issue WHERE
issue.status = ’Open’ GROUP BY issue.assignee ))
```

Target:

```
SELECT assignee, Count(issue_id) FROM issue WHERE status = "Open" GROUP
BY assignee ORDER BY Count(issue_id) DESC LIMIT 1
```

Possible Fail reason:

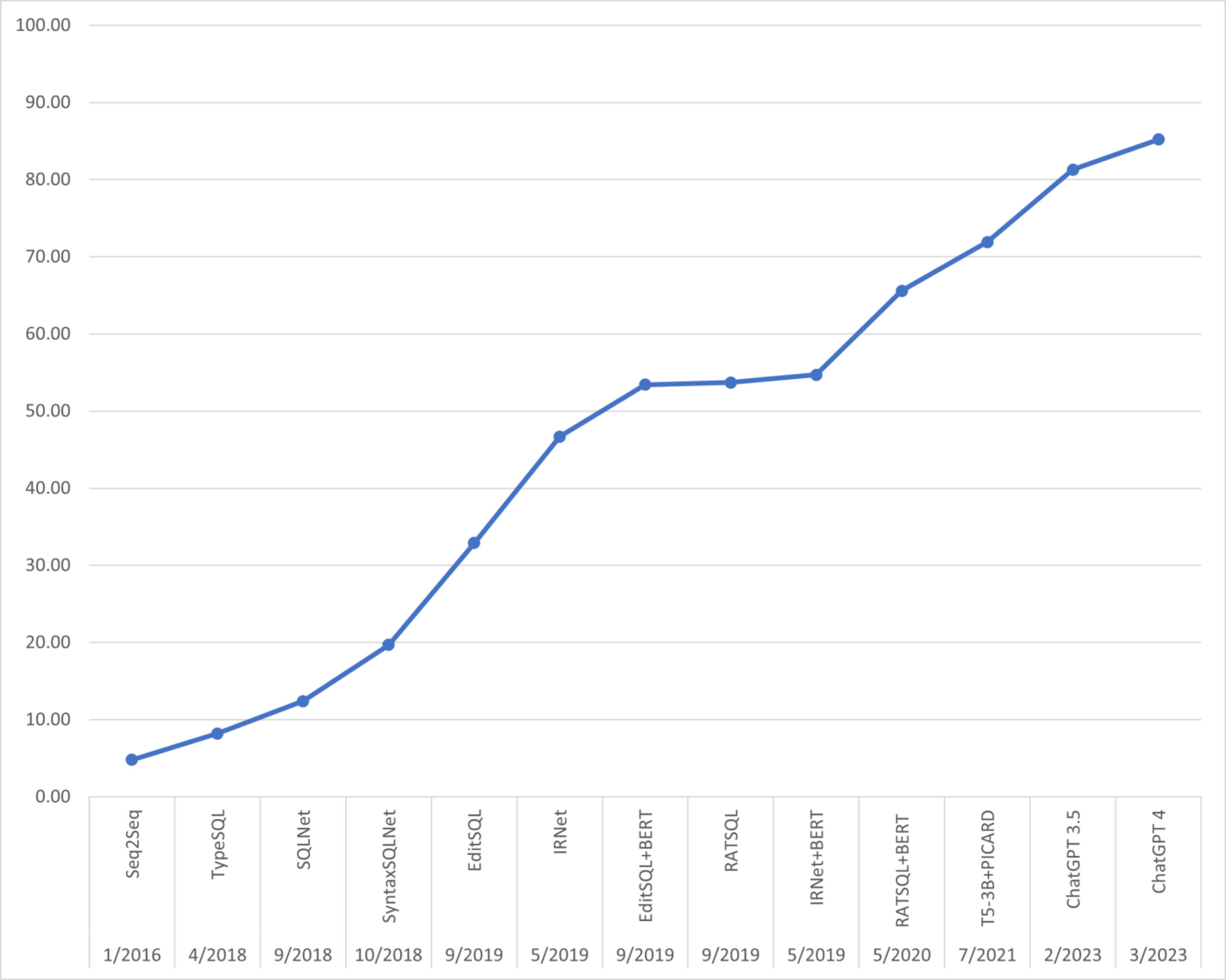
The model is having difficulties understanding the final request from the complex utterance with multiple requests. Word-to-word transformation is not working here.

SPIDER + ChatGPT

SPIDER evaluation with ChatGPT

Accuracy	easy 248	medium 446	hard 174	extra hard 166	all 1034
GPT 3.5 execution	0.964	0.883	0.644	0.596	0.816
GPT 3.5 exact match	0.972	0.881	0.621	0.596	0.813
GPT 4 execution	0.980	0.930	0.678	0.651	**0.855**
GPT 4 exact match	0.980	0.933	0.667	0.639	**0.852**

Table 13: Comparison between Accuracies



TECHNISCHE UNIVERSITÄT ILMENAU

Un/Fortunately :)

Spider 1.0

Yale Semantic Parsing and Text-to-SQL Challenge

Rank	Model	Test
1 Apr 21, 2023	DIN-SQL + GPT-4 *University of Alberta* (Pourreza et al.,'2023) code	**85.3**
2 Jun 1, 2023	C3 + ChatGPT *Anonymous*	**82.3**

Update GPT-4-0613

```
completion = openai.ChatCompletion.create(
    model="gpt-4-0613",
    messages=[{"role": "user", "content": example_user_input}],
        functions=[
        {
            "name": "get_sql_query",
            "description": "Get a valid SQL query",
            "parameters": {
                "type": "object",
                "properties": {
                    "query": {
                        "type": "string",
                        "description": "Returning a valid SQL query"
                    }
                },
                "required": ["query"]
            }
        }
        ],
        function_call={"name": "get_sql_query"},
)
```

Conclusion
& Future Directions

Future Directions

- Address performance loss in other domains and also when dealing with dynamic database structure
- Incorporate DB management and modification commands.
    - Do we need to a new dataset?
- Explore conversational and multilingual Text-to-SQL applications.
- My next project ChatGPT + Text-to-SamQL-to-SQL!
- Is it possible to combine PrivateGPT or OpenLLM with PICARD?
- Discuss if Spider is still relevant?! Spider is the new WikiSQL?

Conclusion

- Rapid growth of Text-to-SQL domain, getting mature.
- Performance influence of datasets like Spider and SEOSS.
- PICARD-T5 identified as promising for fine-tuning with high-performance computing resources.
- Potential and Impact of LLMs on Text-to-SQL.
- Potential for prompt learning to enhance Text-to-SQL robustness.
- Future of this research field

Thanks,
Any Questions?

THANK-YOU.md

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

```sql
/* Thank You */
SELECT 'appreciation'
FROM 'my_heart'
WHERE 'audience' = 'you';
```

## Student:

- Shahriar Yazdipour
- shahriar.yazdipour@tu-ilmenau.de

## Project Repository:

- https://github.com/yazdipour/text-to-sql-gpt
- https://github.com/yazdipour/text-to-sql-seoss-t5
- https://github.com/yazdipour/text-to-sql-thesis
- https://github.com/yazdipour/ez-picard
````

References

- Naihao Deng, Yulong Chen, and Yue Zhang. Recent advances in text-to-sql: A survey of what we have and what we expect, 2022.
- Qin Lyu, Kaushik Chakrabarti, Shobhit Hathi, Souvik Kundu, Jianwen Zhang, and Zheng Chen. Hybrid ranking network for text-to-SQL. 2020.
- Torsten Scholak, Nathan Schucher, and Dzmitry Bahdanau. PICARD: Parsing incrementally for constrained auto-regressive decoding from language models. In Proceedings of the 2021 Conference on Empirical Methods in Natural. Language Processing, pages 9895–9901. Association for Computational. Linguistics, November 2021.
- lvis Saravia. Prompt Engineering Guide. https://github.com/dair-ai/PromptEngineering-Guide, 12 2022.
- Spider: A Large-Scale Human-Labeled Dataset for Complex and Cross-Domain Semantic Parsing and Text-to-SQL Task. 2018. Tao Yu and Rui Zhang and Kai Yang and Michihiro Yasunaga and Dongxu Wang and Zifan Li and James Ma and Irene Li and Qingning Yao and Shanelle Roman and Zilin Zhang and Dragomir Radev
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. CoRR, abs/1810.04805, 2018.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

- You can find Full list of references in thesis document.

Encoders

Methods	Adopted by	Applied datasets	Addressed challenges
Encode token type	TypeSQL	WikiSQL	Representing question meaning
Graph-based	GNN	Spider	Representing question and DB schemas in a structured way and Schema linking
	Global-GCN	Spider	
	IGSQL	Sparc, CoSQL	
	RAT-SQL	Spider	
	LEGSQL	Spider	
	SADGA	Spider	
	ShawdowGNN	Spider	
	S2SQL	Spider	
Self-attention	X-SQL	WikiSQL	Representing question and DB schemas in a structured way and Schema linking
	SQLova	WikiSQL	
	RAT-SQL	Spider	
	DuoRAT	Spider	
	UnifiedSKG	WikiSQL, Spider	
Adapt PLM	X-SQL	WikiSQL	Leveraging external data to represent question and DB schemas
	SQLova	WikiSQL	
	Guo	WikiSQL	
	HydraNet	WikiSQL	
Pre-training	TaBERT	Spider	Leveraging external data to represent question and DB schemas
	GraPPA	Spider	
	GAP	Spider	

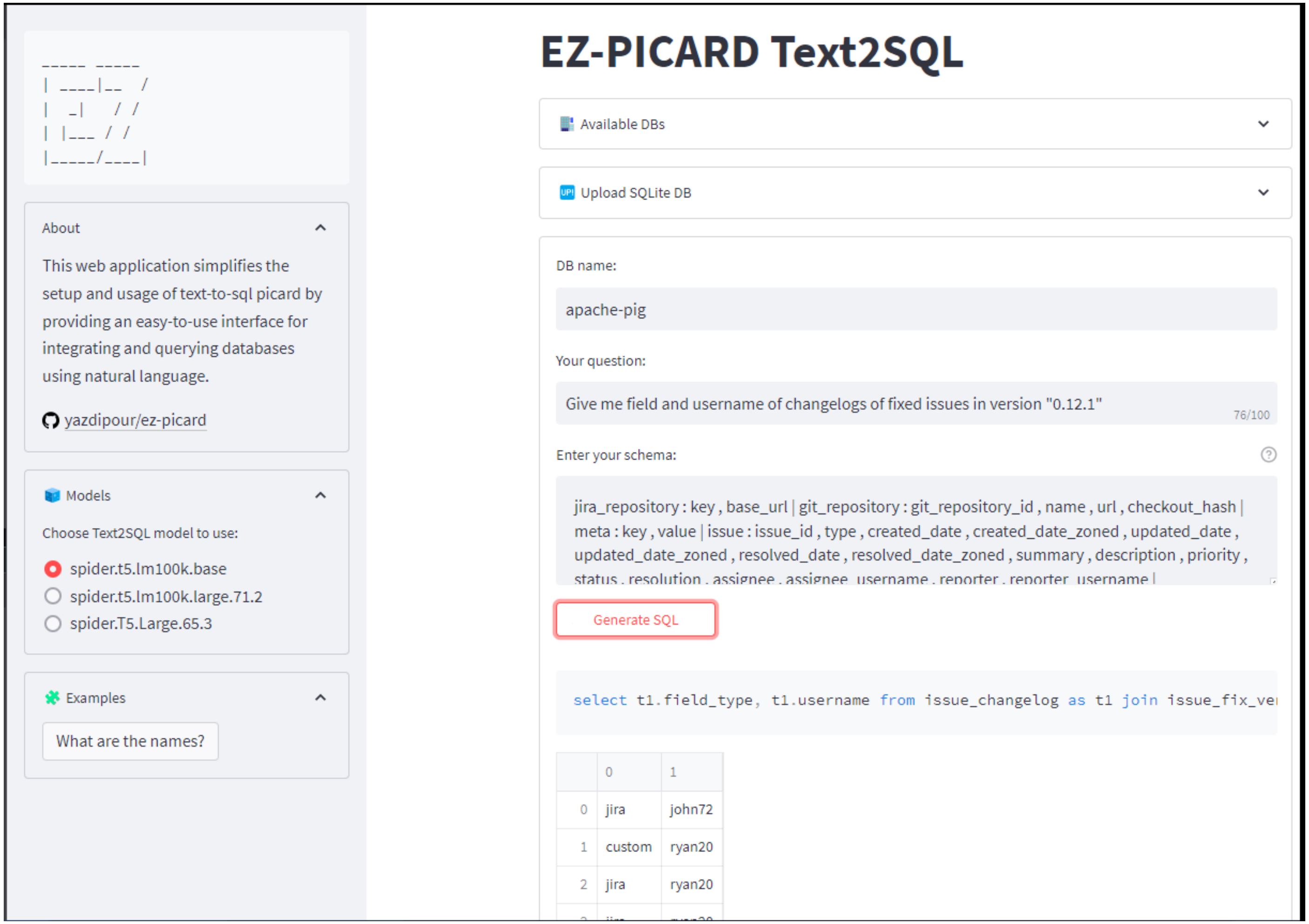
Decoders

Methods	Adopted by	Applied datasets	Addressed challenges
Tree-based	Seq2Tree Seq2AST SyntaxSQLNet	- - Spider	Hierarchical decoding
Sketch-based	SQLNet Coarse2Fine IRNet RYANSQL	WikiSQL WikiSQL Spider Spider	Hierarchical decoding
Bottom-up	SmBop	Spider	Hierarchical decoding
Self-Attention	Seq2Tree Seq2SQL	- WikiSQL	Synthesizing information
Bi-attention	BiSQL	Spider	Synthesizing information
Relation-aware Self-attention	DuoRAT	Spider	Synthesizing information
Copy Mechanism	Seq2AST Seq2SQL SeqGenSQL	- WikiSQL WikiSQL	Synthesizing information
Intermediate Representation	IncSQL IRNet ValueNet	WikiSQL WikiSQL Spider	Bridging the gap between natural language and SQL query
Constrained decoding	PICARD	Spider	Fine-grained decoding

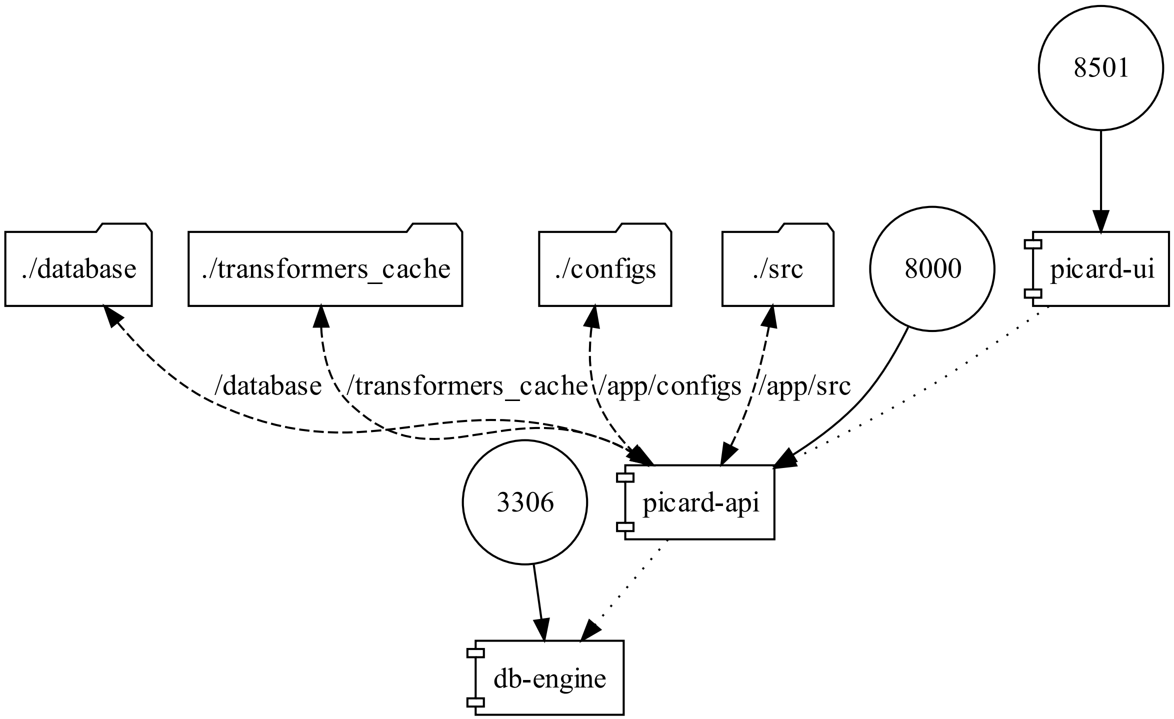
Prompt Engineering
Rules:

1. Give ChatGPT an identity and intended audience "You are a text-to-sql assistant, do...”
2. Offer and give specific context
3. Highlight information to include or exclude
4. Choose a relevant tone of voice and writing style
5. Give examples to base the response on
6. Specify response length
7. Clarity and specificity

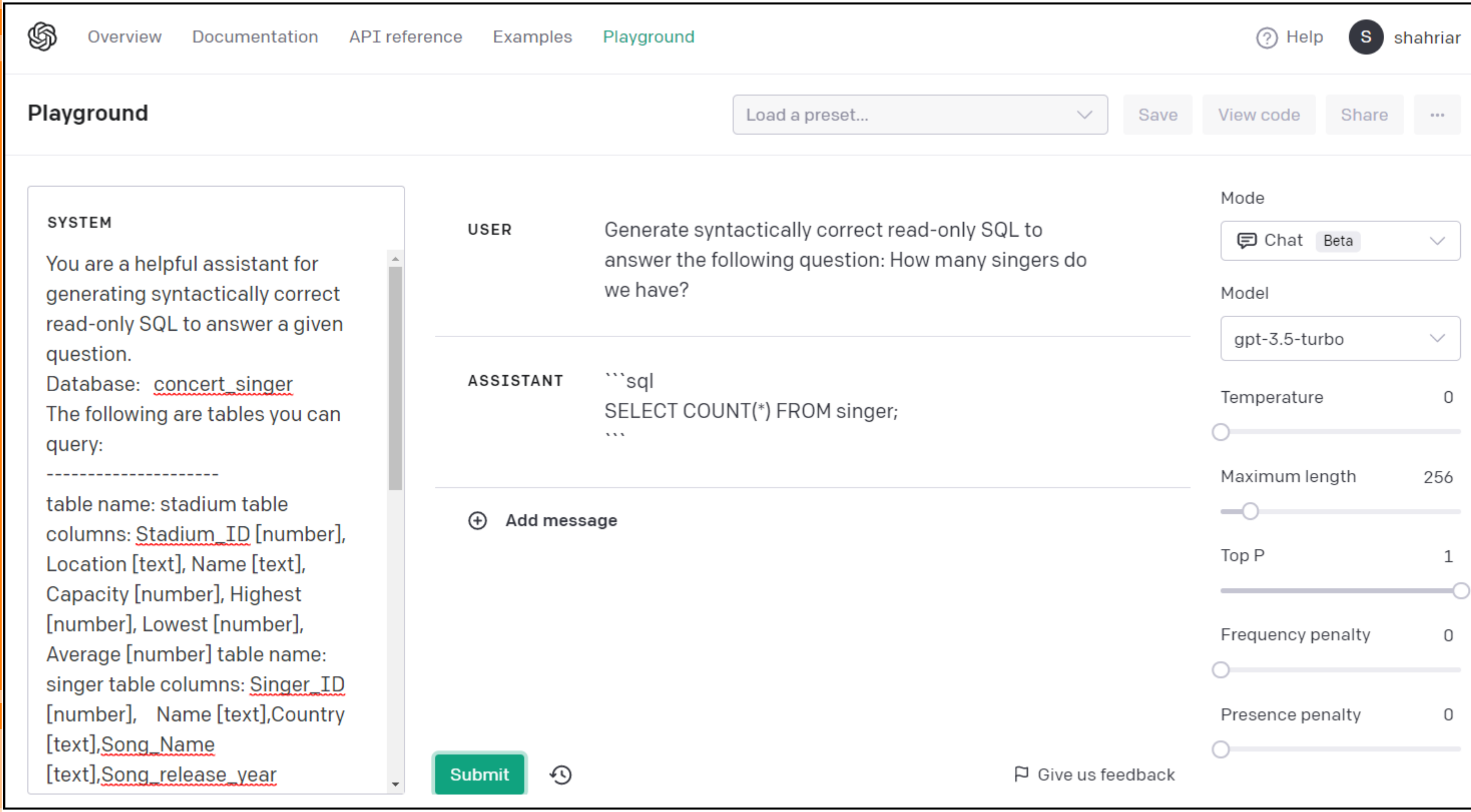
EZ-PICARD Text2SQL Client



EZ-PICARD Text2SQL Client

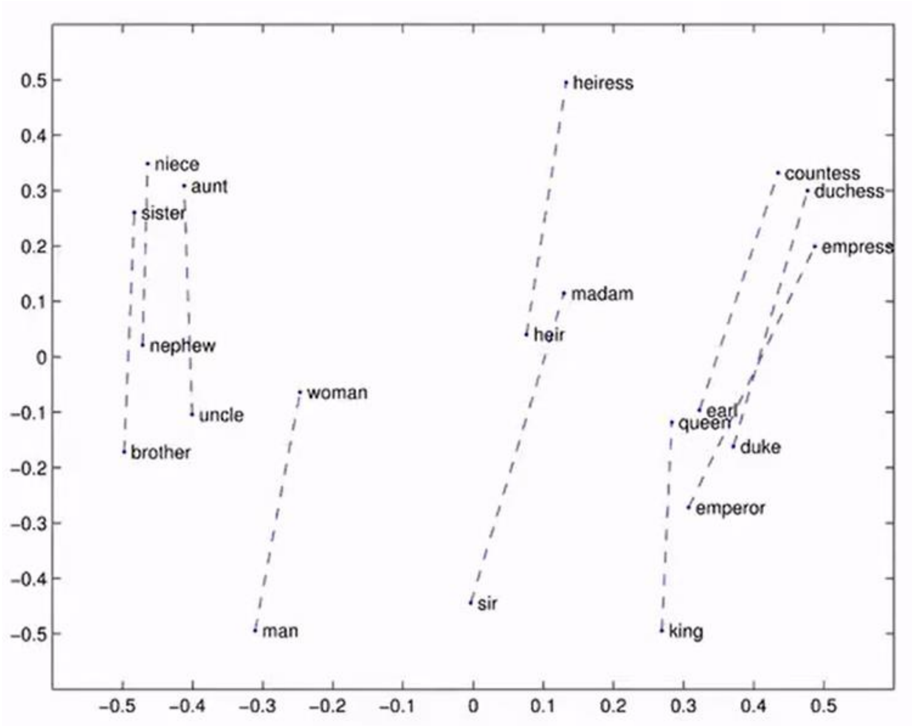


OpenAI Playground



GloVe - Embedding

- Create **meaningful vector representations**
- **Unsupervised** learning based on word co-occurrence in the training corpus
- Useful **linear substructures** for word relations
- Easy to find semantical near neighbors
- Pre-trained vectors created from large corpuses are available for download

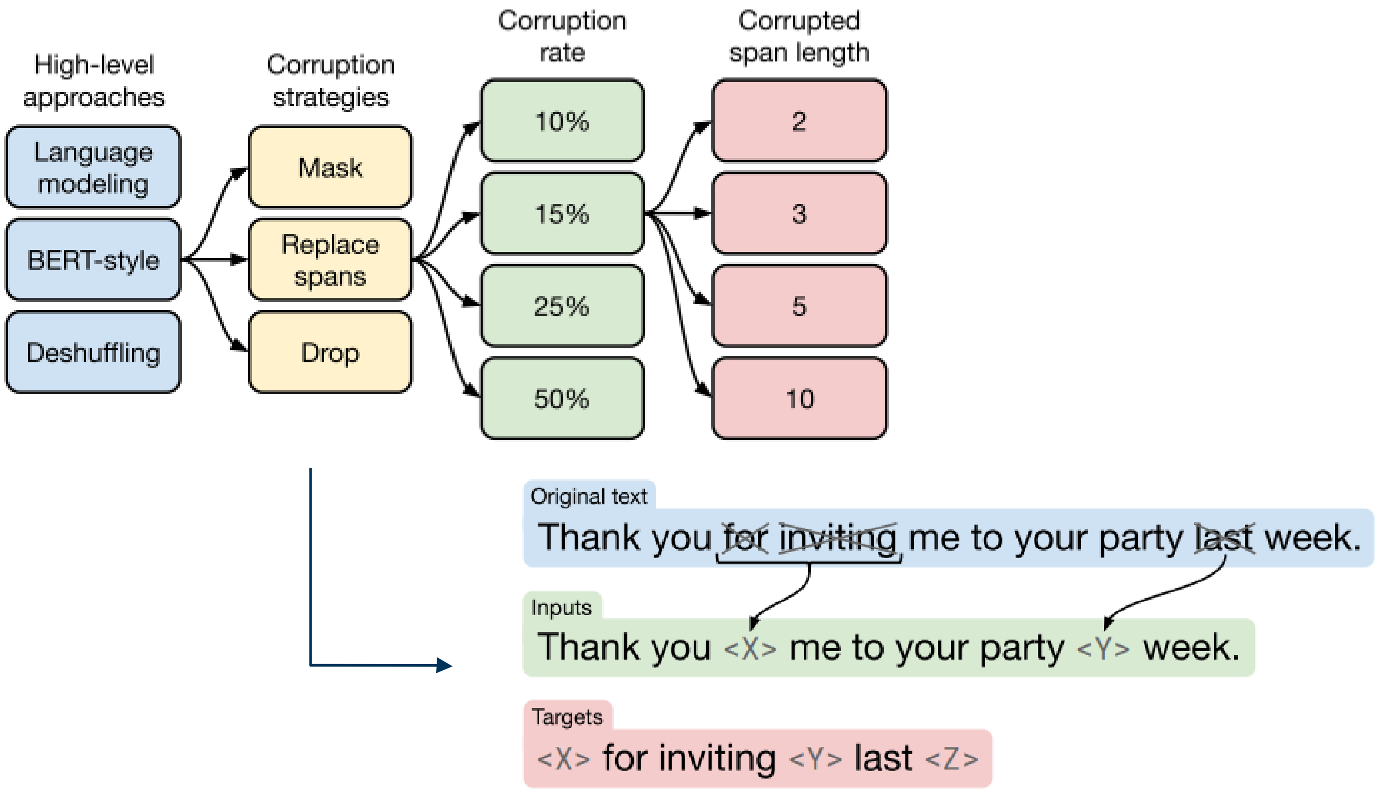


NearestNeighbours(**frog**) = [frogs, toad, litoria, leptodactylidae, rana, lizard, eleutherodactylus]

Colossal Clean Crawled Corpus (C4 Dataset)

- Unlabeled dataset gathered and filtered from CCD
- Common Crawl Dataset: a non-commercial crawler that saves snapshots of the web every month. And web content is dumped out on the order of **20 terabytes.**
- The corpus is 'clean' because it undergoes a comprehensive cleaning process to remove inappropriate content, non-textual elements, and duplicates.

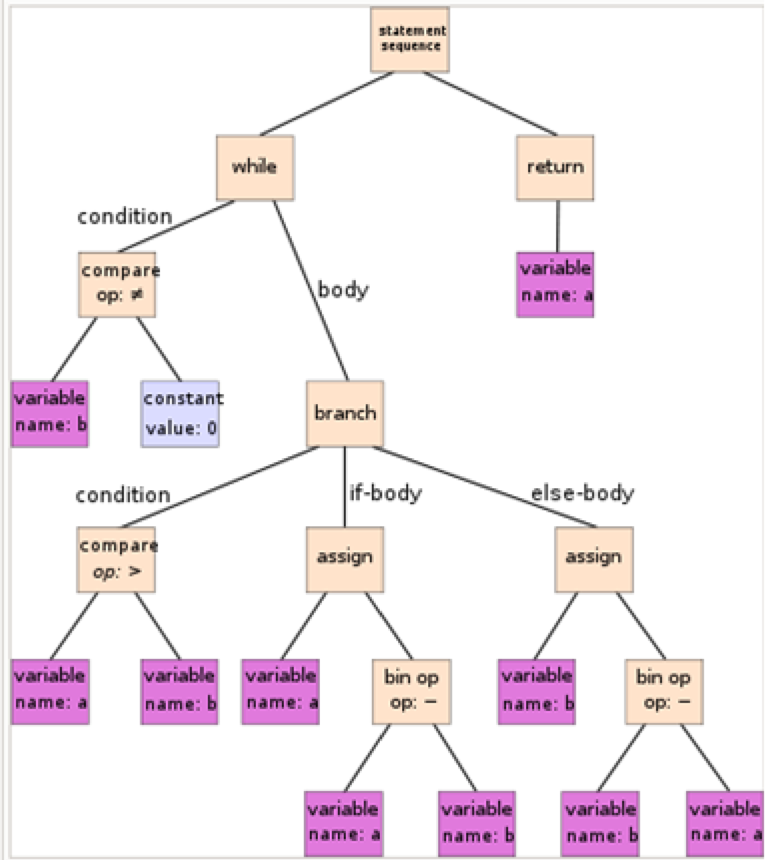
T5



High-level approaches
Language modeling
BERT-style
Deshuffling
Corruption strategies
Mask
Replace spans
Drop
Corruption rate
10%
15%
25%
50%
Corrupted span length
2
3
5
10
Original text
Thank you ~~for inviting~~ me to your party ~~last~~ week.
Inputs
Thank you <X> me to your party <Y> week.
Targets
<X> for inviting <Y> last <Z>

AST

Abstract Syntax Tree



An abstract syntax tree for the following code for the Euclidean algorithm:

```
while b ≠ 0:
    if a > b:
        a := a - b
    else:
        b := b - a
return a
```