**Advanced techniques for evaluating Text to SQL generation**

**Introduction**

Traditionally, two evaluation metrics have been primarily used for assessing the accuracy of SQLs. They are **Exact Matching**(EM) and **Execution Accuracy(EX)**. EM measures whether the predicted query as a whole is equivalent to the gold query. The challenge however is it is possible to encounter false negative evaluations since a question might be solvable by multiple syntactically different but semantically identical SQL statements.  Execution accuracy is a more widely used metric, which instead of directly comparing the SQL queries, executes the queries against a database and compares the resulting outputs. This also however has its own set of challenges

systems take natural language questions and generate SQL queries as output. The evaluation of these systems needs to account for the fact that SQL queries can often be written in different ways that are logically equivalent but not syntactically identical.

We also acknowledge that many times, there are multiple correct ways to answer a given question, which we would like to accept / mark as correct. Here are 4 acceptable examples for the question: What are the top 3 restaurants in New York?

SELECT name

FROM restaurants

GROUP BY name

ORDER BY AVG(rating) DESC LIMIT 3

SELECT id, name

FROM restaurants

GROUP BY name

ORDER BY AVG(rating) DESC LIMIT 3

To account for these different approaches, this paper discusses different approaches and metrics to solve this issue. Moreover, the

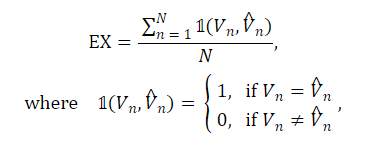
1. Using better techniques
2. Using better metrics
3. Using LLM as an evaluator (human aligned evaluation)

**Traditional Approach**

**Exact matching**

**Challenges**

**Execution Accuracy**

****

**Challenges**

**Alternative techniques**

**Multiple Gold query creation**

**Subset Evaluation based Execution Accuracy**

1. For each column in df1, we check to see if the same column of values exists in df2. We ignore data types, column names (since these could be [aliased](https://www.w3schools.com/sql/sql_alias.asp)) and row order while doing so.

2. After picking out the relevant columns from df2, renaming them with the names from df1, we check that the overall dataframe matches df1. This is to ensure that we don’t accidentally match on shuffled data columns (which is quite common for column data types with low cardinality like booleans or enums) with the same unordered list of values but from the wrong column.

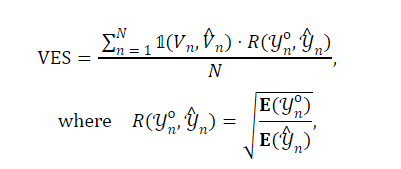
**Alternative metrics**

**Soft F1 Score**

 This metric is specifically designed to assess the performance of text-to-SQL models by measuring the similarity between the tables produced by predicted SQL queries and those from the ground truth. In a nutshell, the soft F1-score is a more lenient metric that reduces the impact of column order and missing values in the tables produced by predicted SQL queries.

<https://github.com/bird-bench/mini_dev>

**Valid Efficiency Score**

****

<https://github.com/bird-bench/mini_dev>

**Reward based Valid Efficiency Score**

**LLM as an evaluator**

**Accuracy score**

[**https://ekzhu.medium.com/human-aligned-text-to-sql-evaluation-399123fa0a64**](https://ekzhu.medium.com/human-aligned-text-to-sql-evaluation-399123fa0a64)

**Application**

**Dataset (academic dataset)**

**Academic dataset (Sqlite from defogAi/defogdata)**

**Take 3 sql queries and match them using accuracy, ves, rves, soft f1 score,**

**References**

1. [**https://arxiv.org/html/2403.02951v1#A3.SS1**](https://arxiv.org/html/2403.02951v1#A3.SS1)
2. [**https://defog.ai/blog/open-sourcing-sqleval**](https://defog.ai/blog/open-sourcing-sqleval)
3. [**https://ekzhu.medium.com/human-aligned-text-to-sql-evaluation-399123fa0a64**](https://ekzhu.medium.com/human-aligned-text-to-sql-evaluation-399123fa0a64)
4. [**https://github.com/jkkummerfeld/text2sql-data**](https://github.com/jkkummerfeld/text2sql-data)