Text-to-SQL Empowered by Large Language Models: A Benchmark Evaluation

Dawei Gao*
Alibaba Group
gaodawei.gdw@alibaba-inc.com

Haibin Wang*
Alibaba Group
binke.whb@alibaba-inc.com

Yaliang Li
Alibaba Group
yaliang.li@alibaba-inc.com

Xiuyu Sun
Alibaba Group
xiuyu.sxy@alibaba-inc.com

Yichen Qian
Alibaba Group
yichen.qyc@alibaba-inc.com

Bolin Ding
Alibaba Group
bolin.ding@alibaba-inc.com

Jingren Zhou
Alibaba Group
jingren.zhou@alibaba-inc.com

摘要

- **背景**：大型语言模型逐渐成为成为Text-to-SQL任务的新范式，但缺乏系统化的基准测试阻碍了有效、高效和经济的LLM-based Text-to-SQL解决方案的设计。
- **研究**：文章首先对现有的提示工程方法进行了系统和广泛的比较，包括问题表示、示例选择和组织，并基于实验结果阐述了它们的优缺点。
- **贡献**：提出了一个新的集成解决方案DAIL-SQL，刷新了Spider排行榜，执行准确率达到86.6%，设定了新的标准。
- **探索**：探索了开源LLMs在不同场景下的潜力，并通过对监督微调来提高性能，强调了开源LLMs在Text-to-SQL中的潜力以及监督微调的优缺点。

引言

- **Text-to-SQL**：作为自然语言处理和数据库领域中的一个挑战性任务，Text-to-SQL将自然语言问题映射到SQL查询。

- **先前工作**：大多数先前工作集中于通过训练编码器-解码器模型来提取问题到SQL的模式并泛化它们。

- **LLMs的出现**：LLMs的出现为Text-to-SQL任务提供了新的解决方案，使用大模型解决任务的核心问题是如何有效地提示LLM生成正确的SQL查询。

- **现状**：

    - 缺乏系统研究：利用LLM完成Text-to-SQL的进展尽管取得了显著进展，但LLM-based Text-to-SQL解决方案的提示工程仍缺乏系统研究。
    - 开源大模型能力尚未充分探索：开源LLMs在编程、数学推理和文本生成任务中显示出显著进步，但在Text-to-SQL任务中尚未得到充分研究。
    - prompt 效率问题：在LLM-based Text-to-SQL中，效率是一个关键挑战，因为调用闭源模型API昂贵、耗时且有速率限制。之前的工作提出，模型准确性与提示词长度之间存在倒U形关系，在合适的提示长度下，模型性能最佳。

- **文章贡献**：

    - 系统评估了现有的TEXT-to-SQL提示工程方法，包括零样本和少样本下的若干策略。
    - 用不同的策略对开源LLMs进行了广泛的实验，并通过监督微调提高了性能。
    - 提出了一个新的集成解决方案DAIL-SQL，刷新了Spider排行榜，执行准确率达到86.6%.

方法

几个概念：

- **零样本场景**：在没有为LLM提供示例的情况下，设计提示以引导LLM生成正确的SQL查询。其主要挑战在于如何融入相关的信息来有效的表示自然语言问题。
- **问题表示**：将表示自然语言问题和相关信息的过程称为问题表示。
- **少样本场景**：为LLM提供一小部分可用示例；因此除了问题表示之外，还需要研究如何选择最有助于问题的示例，并在提示中适当组织它们。
- **示例选择和组织**：选择并组织对LLM有益的示例的过程。

零样本

问题表示：

- Basic Prompt：最简单的表示方法，仅包含数据库模式和自然语言问题，前缀为“Q: ”，后跟“SELECT”作为响应前缀，以提示LLM生成SQL查询。

```
Table continents, columns = [ContId, Continent]
Table countries, columns = [CountryId, CountryName,
↳ Continent]
Q: How many continents are there?
A: SELECT
```

Listing 1: Example of Basic Prompt

- Text Representation Prompt ： 将数据库模式和问题都以自然语言的形式表示，并在提示的开头添加指令，以指导LLMs。例如，一个特定的任务描述，如“Write a SQL to answer the question”。

```
Given the following database schema:
continents: ContId, Continent
countries: CountryId, CountryName, Continent

Answer the following: How many continents are there?
SELECT
```

Listing 2: Example of Text Representation Prompt

- OpenAI Demostration Prompt：这种方法最初在OpenAI的官方Text-to-SQL演示中使用，包含指令、数据库模式和问题，所有信息都以注释（例如使用“#”）的形式呈现。它的指令更具体，例如包含规则“Complete sqlite SQL query only and with no explanation”。

```
### Complete sqlite SQL query only and with no
↳ explanation
### SQLite SQL tables, with their properties:
#
# continents(ContId, Continent)
# countries(CountryId, CountryName, Continent)
#
### How many continents are there?
SELECT
```

Listing 3: Example of OpenAI Demostration Prompt

- Code Representation Prompt：将Text-to-SQL任务以SQL语法的形式呈现，直接提供“CREATE TABLE” SQL语句，并在注释中提示LLM回答目标问题。

```
/* Given the following database schema: */
CREATE TABLE continents(
    ContId int primary key,
    Continent text,
    foreign key(ContId) references countries(Continent)
);

CREATE TABLE countries(
    CountryId int primary key,
    CountryName text,
    Continent int,
    foreign key(Continent) references continents(ContId)
);

/* Answer the following: How many continents are there?
↳ */
SELECT
```

Listing 4: Example of Code Representation Prompt

- Alpaca SFT Prompt:
  这种提示是为监督式微调（Supervised Fine-Tuning, SFT）设计的，以Markdown格式提示LLM遵循指令并根据输入上下文完成任务。

```
Below is an instruction that describes a task, paired
↳ with an input that provides further context. Write a
↳ response that appropriately completes the request.

### Instruction:
Write a sql to answer the question "How many continents
↳ are there?"

### Input:
continents(ContId, Continent)
countries(CountryId, CountryName, Continent)

### Response:
SELECT
```

Listing 5: Example of Alpaca SFT Prompt

少样本

示例选择

示例选择：从候选的示例集合中选择最有助于LLM生成正确SQL查询的示例。将这些示例提供给LLM，使LLM通过上下文学习，利用给出的相似示例来推断出正确的SQL查询。

- **Random（随机选择）**：
  - 这种策略从可用的候选示例中随机抽取一定数量的示例。它通常作为其他更复杂选择策略的基线。
- **Question Similarity Selection (QTS)**:
  - QTS策略选择与目标问题最相似的示例。它通过嵌入模型将示例问题和目标问题嵌入到向量空间中，然后使用距离度量（如欧几里得距离或余弦相似性）来选择最相似的示例。
- **Masked Question Similarity Selection (MQS)**:
  - MQS策略针对跨域Text-to-SQL任务，通过将所有问题中的表名、列名和值替换为掩码标记，来消除领域特定信息的负面影响，然后计算它们与目标问题的相似性。

- **Query Similarity Selection (QRS)**:
  - QRS策略选择与目标SQL查询最相似的示例。先使用模型生成SQL查询（认为它是目标SQL的近似），根据与近似查询与所有示例的相似性和所选示例的多样性来选择k个示例。

示例组织

示例组织：将选定的示例组织到提示中，以便LLM能够有效地利用这些示例。

- **Full-Information Organization (FI)**:
  - FI策略以与目标问题相同的方式组织示例，包括指令、数据库模式、问题和正确的SQL查询。这种方法保留了完整的示例信息，有助于LLMs理解问题和查询之间的映射关系。

```
/* Given the following database schema: */
${DATABASE_SCHEMA}
/* Answer the following: How many authors are there? */
SELECT count(*) FROM authors

/* Given the following database schema: */
${DATABASE_SCHEMA}
/* Answer the following: How many farms are there? */
SELECT count(*) FROM farm

${TARGET_QUESTION}
```

Listing 6: Example of Full-Information Organization.

- **SQL-Only Organization (SO)**:
  - SO策略仅在提示中包含选定示例的SQL查询，不包括其他信息。这种方法旨在在有限的token长度内包含尽可能多的示例，但可能丢失了问题和查询之间的映射信息。

```
/* Some SQL examples are provided based on similar
↳ problems: */
SELECT count(*) FROM authors

SELECT count(*) FROM farm

${TARGET_QUESTION}
```

Listing 7: Example of SQL-Only Organization.

DAIL-SQL

DAIL-Selection: 结合了MQS和QRS策略，以选择最相似的示例和SQL查询。

- 问题的相似性：首先在目标问题q和候选集Q中的示例问题qi中遮盖领域特定的词语。然后，基于掩码q和qi的嵌入之间的欧几里得距离对候选示例进行排名。
- SQL查询的相似性：计算预预测的SQL查询s'与Q中的si之间的查询相似度。
- 筛选其中SQL查询的相似性大于预定义阈值的示例，在筛选后的示例中根据问题相似性进行排序，选择最相似的k个示例。

DAIL-Organization: FI和SO之间的折衷，平衡了信息和数量。

- 保留了问题和SQL查询之间的映射信息，舍去了数据库模式的说明
- 问题表示采用Code Representation Prompt，也就是数据库模式通过SQL "CREATE TABLE"语句表示，问题和SQL查询之间的映射信息通过注释表示。其包含了数据库的全部信息，包括主键和外键，能为LLM提供更多有用的信息。

监督微调

对于Text-to-SQL数据集$\mathcal{T} = \{q_i, s_i, D_i\}$, 其中的数据为在数据库$D_i$中的问题$q_i$和对应的SQL查询$s_i$. 监督微调的数据即为$\{input = \sigma(q_i, D_i), output = s_i\}$, 其中$\sigma(q_i, D_i)$是对问题和数据库模式的表示方法，如Basic Prompt、Text Representation Prompt等。

实验

数据集

- Spider：大规模跨域的Text-to-SQL数据集，包含训练集中的8659个实例和开发集中的1034个实例，跨越200个数据库。每个实例由一个针对特定数据库的自然语言问题及其对应的SQL查询组成。文中使用开发集Spider-dev作为测试集进行评估
- Spider-Realistic：是Spider的一个更具挑战性的变种。它从Spider-dev中选择508个例子，并手动修订问题，但保留其对应的SQL查询不变。

对于少样本场景，在使用Spider-dev和Spider-Realistic进行测试时，利用Spider的训练集作为示例候选集。

指标

- EM：精确匹配，即模型生成的SQL查询与真实SQL查询完全匹配时为1，否则为0。
- EX：执行准确率，即模型生成的SQL查询在数据库上执行时返回正确结果的比例。单个查询可能有多个有效的SQL查询。

实验结果

问题表示（零样本）

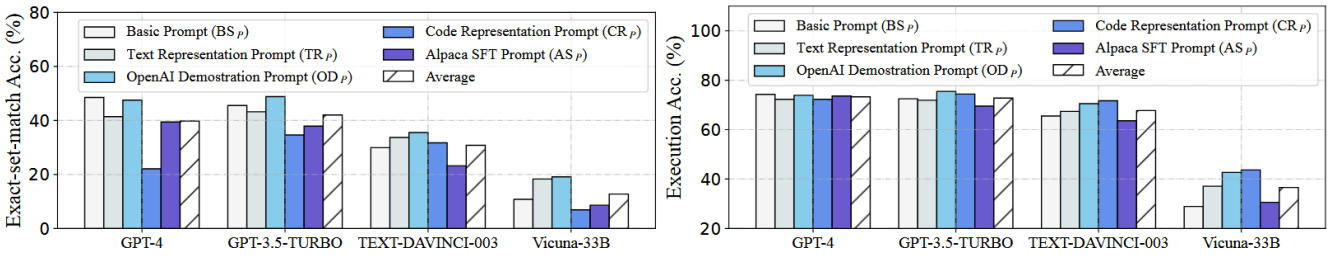


Figure 1: Results of different question representations on Spider-dev under zero-shot scenario.

实验一： 零样本下不同问题表示方法的结果对比（左：EM,右：EX）
OpenAI Demostration Prompt对各个模型的适配性都比较好，在EM和EX上都有较好的表现；而Alpaca SFT Prompt的适配较差。对于GPT-4与GPT-3.5，Basic Prompt 已经足够好，在token成本上有优势。

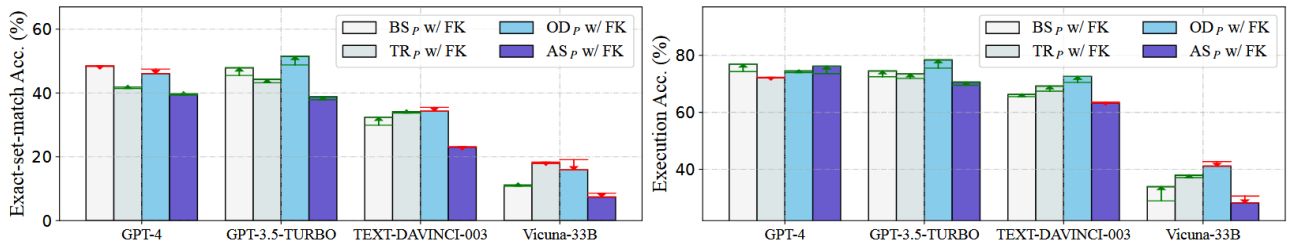


Figure 2: Ablation studies of foreign keys information on Spider-dev. The green arrow indicates an increase, and red arrow indicates a decrease.

消融实验： Code Representation Prompt中数据库模式“CREATE TABLE”语句中包含了外键的信息，尝试将外键信息加入到其他Prompt中，发现在EM和EX上都有提升。

示例选择（少样本）

Few-shot	Selection	Question Similarity	Query Similarity	GPT-4		GPT-3.5-TURBO		TEXT-DAVINCI-003		Vicuna-33B	
				EM	EX	EM	EX	EM	EX	EM	EX
0-shot	-	-	-	22.1	72.3	34.6	74.4	31.7	71.7	6.9	43.7
1-shot	Random	0.23	0.47	41.7	77.4	45.9	73.9	38.2	70.6	14.4	47.9
	Question Similarity selection	0.39	0.65	53.3	78.8	51.9	74.3	44.1	72.3	16.5	48.5
	Masked Question Similarity selection	0.57	0.80	58.2	79.1	57.4	76.0	47.9	75.0	21.4	48.7
	DAIL selection	0.56	0.95	62.1	80.2	59.5	75.5	51.9	76.9	22.8	49.2
	Upper Limit	0.56	0.98	63.7	81.0	61.4	77.2	53.1	77.5	22.7	49.4
3-shot	Random	0.23	0.48	48.9	79.4	49.0	73.6	41.7	71.6	16.8	46.9
	Question Similarity selection	0.37	0.63	56.3	79.2	53.8	74.7	52.2	74.1	21.1	47.1
	Masked Question Similarity selection	0.54	0.78	66.1	81.5	61.1	77.3	59.7	77.0	27.7	52.3
	DAIL selection	0.53	0.94	69.1	81.7	63.9	77.8	64.4	79.5	30.7	53.6
	Upper Limit	0.53	0.98	71.5	83.4	66.2	79.2	66.7	81.1	31.2	54.4
5-shot	Random	0.23	0.48	51.6	79.5	52.9	75.7	49.0	72.1	-	-
	Question Similarity selection	0.36	0.61	58.2	79.9	55.9	75.1	54.8	73.2	-	-
	Masked Question Similarity selection	0.52	0.77	66.8	82.0	62.3	77.9	64.7	78.6	-	-
	DAIL selection	0.52	0.94	71.9	82.4	66.7	78.1	67.7	80.5	-	-
	Upper Limit	0.51	0.97	74.4	84.4	68.8	79.6	70.7	82.4	-	-

Table 2: Evaluation on Spider-dev with different example selections. The organization is fixed to Full-Information Organization.

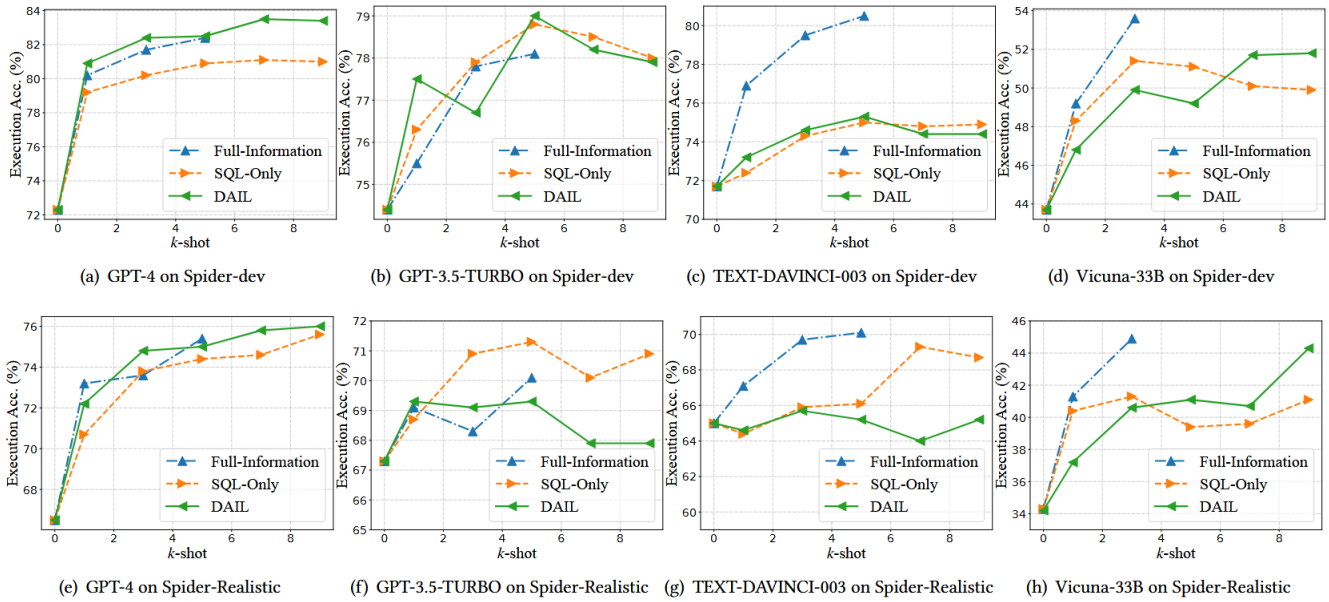
实验二： 各示例选择方法所选例子与目标示例之间的问题和查询的Jaccard相似度，在表2中“Question Similarity”和“Query Similarity”列中记录平均值。

除了几种策略，还计算了数据库Groud Truth问题和查询的Jaccard相似度，作为两个相似度的上限，记录为Upper limit。

示例组织方法固定为Full-Information Organization。

表中显示，DAIL-Selection方法优于其他。

示例组织（少样本）



(a) GPT-4 on Spider-dev (b) GPT-3.5-TURBO on Spider-dev (c) TEXT-DAVINCI-003 on Spider-dev (d) Vicuna-33B on Spider-dev

(e) GPT-4 on Spider-Realistic (f) GPT-3.5-TURBO on Spider-Realistic (g) TEXT-DAVINCI-003 on Spider-Realistic (h) Vicuna-33B on Spider-Realistic

Figure 4: Evaluation on Spider-dev with different example organizations. The selection is fixed to DAIL Selection.

实验三： 模型在数据集中不同示例组织方法在不同示例个数下的EX表现。示例选择方法固定为DAIL-Selection。

GPT-4从DAIL-SQL中获益最大，相比于其他两种组织方法获得了更高的EX。
GPT-3.5-TURBO和TEXT-DAVINCI003，增加示例可能会降低执行准确率，因为它们的上下文学习能力有限。
Vicuna-33B在Full-Information Organization下表现最好，DAIL次之。

具有更强上下文学习能力的LLM，如GPT-4，从DAIL Organization中受益最大，而较弱的LLM则需要更多信息来从示例中学习。

开源模型的监督微调

LLM	Org.	0-shot		1-shot		3-shot		5-shot	
		EM	EX	EM	EX	EM	EX	EM	EX
LLaMA -7B	FI $_O$	3.1	13.0	23.4	30.1	23.7	30.3	24.7	30.9
	SO $_O$	3.1	13.0	13.3	21.4	15.2	24.1	15.3	25.0
	DAIL $_O$	3.1	13.0	18.5	25.4	22.1	28.1	22.6	29.3
+ SFT	FI $_O$	63.9	66.7	59.6	61.4	58.7	61.4	59.4	61.5
	SO $_O$	63.9	66.7	59.8	62.3	58.8	61.1	59.5	62.2
	DAIL $_O$	63.9	66.7	58.5	61.9	59.8	61.7	58.9	60.9
LLaMA -13B	FI $_O$	2.4	20.3	21.6	33.8	27.3	38.1	28.5	38.8
	SO $_O$	2.4	20.3	20.7	33.6	23.2	35.9	27.4	36.9
	DAIL $_O$	2.4	20.3	13.2	30.0	15.5	32.3	16.2	32.4
+ SFT	FI $_O$	62.7	67.0	61.9	67.1	60.5	65.0	60.9	65.0
	SO $_O$	62.7	67.0	61.9	66.2	60.1	64.6	60.2	65.2
	DAIL $_O$	62.7	67.0	62.5	66.5	60.6	66.0	61.3	66.4

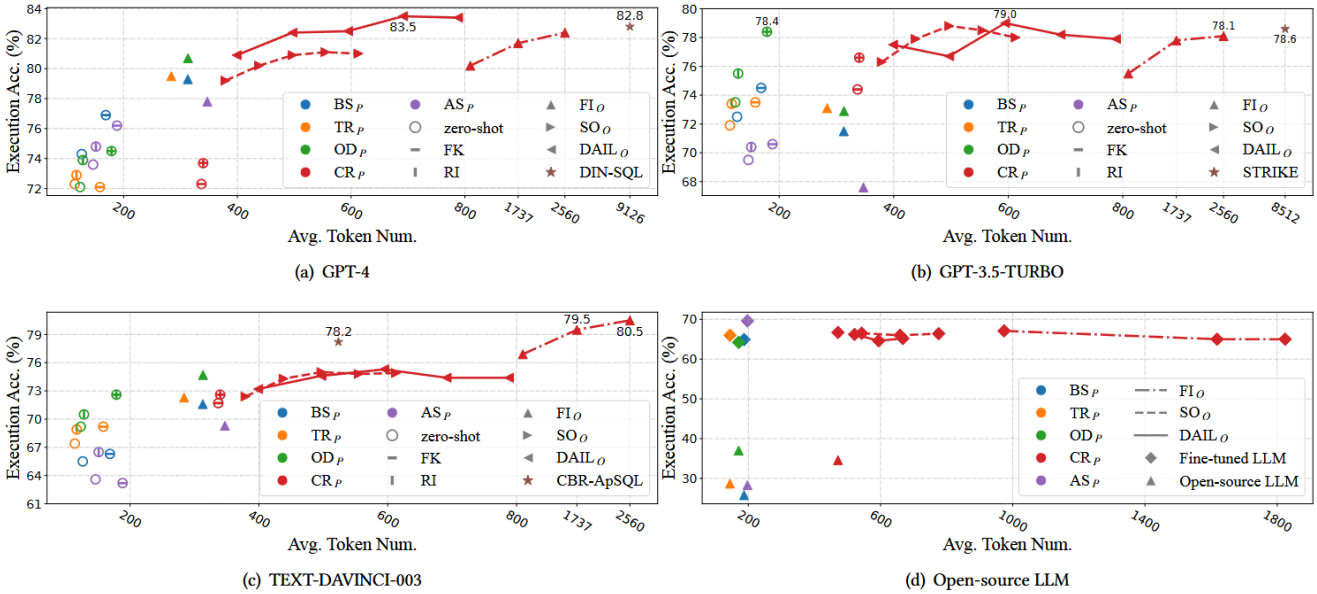
Table 4: Few-shot evaluation results of supervised fine-tuned LLMs on Spider-dev.

将原始LLaMA-7B和13B 与 微调后的模型在零样本及少样本情形下评估

虽然两个模型在微调之后，EM和EX都得到了显著提升；但是在微调之后的模型中，少样本情形相较于零样本的性能反而有所下降。

一个可能的原因是LLM对零样本提示过度拟合，使得示例变得无用。

token 效率的研究



(a) GPT-4 (b) GPT-3.5-TURBO

(c) TEXT-DAVINCI-003 (d) Open-source LLM

实验四： 不同模型、不同问题表示和示例组织方法下的token数和EX的关系。
不同的颜色表示了不同的问题表示方式，红色表示Code Representation Prompt，红色在四个模型中都取得了几乎最好的结果。

不同的形状表示了不同的示例组织方法，圆圈表示零样本，向上三角为Full-Information Organization，向左三角为DAIL Organization。
在GPT-4中，少样本的DAIL Organization相较于零样本，token数增加且EX有所提升。
而GPT-3.5和其他模型中，可能由于模型理解能力不够，少样本增加了token数，但是EX并没有明显有效的提升。

结论

- 本文系统评估了现有的TEXT-to-SQL提示工程方法，包括零样本和少样本下的问题表示、示例选择和组织等策略
- 提出了一个新的集成解决方案DAIL-SQL，执行准确率达到86.6%
- 探索了开源LLMs在不同场景下的潜力，并通过对监督微调来提高性能，强调了开源LLMs在Text-to-SQL中的潜力以及监督微调的优缺点。