DIN-SQL：分解上下文学习文本到SQL并进行自我校正

摘要

在本次复现工作中，探讨了通过将复杂的Text-to-SQL任务分解为多个子任务，来提升大型语言模型（LLMs）在推理过程中的表现。原论文提出的DIN-SQL方法显示，通过精细化的任务分解，能够显著提高LLM在推理中的准确性，并在Spider数据集上取得了新的SOTA（State-of-the-Art）成绩。原文模型基于GPT-4，在经过精细调整后达到了85.3%的准确率。然而，在复现过程中，使用了GPT-3.5-turbo模型，并且对提示（prompt）进行了简化，结果导致准确率降至50.9%。尽管复现结果未能达到原论文所示的准确度，但仍展示了Text-to-SQL任务分解方法的巨大潜力以及实施过程中遇到的挑战。复现过程强调了模型版本、提示策略和任务分解方法对性能的关键影响。尽管GPT-3.5-turbo在准确性上未能达到预期，实验依然提供了有价值的见解，揭示了Text-to-SQL任务分解方法在不同模型和策略下的潜力及局限性。未来的研究可以进一步优化模型和提示设计，以缩小与原始方法之间的性能差距。

关键词：Text-to-SQL；Spider；prompt；大语言模型

1 引言

自然语言与数据库之间的交互日益成为数据库管理系统中的一个重要研究领域。通过数据库的自然语言接口，用户可以用自然语言查询数据库，降低了数据库的使用门槛。然而，尽管近年来自然语言处理（NLP）技术和大型语言模型（LLMs）取得了显著进展，Text-to-SQL任务仍然是一个非常具有挑战性的课题。该任务的核心是将自然语言查询自动转化为正确的SQL查询语句，而这一过程涉及到复杂的语法理解和SQL查询生成。

传统的Text-to-SQL方法 [1, 12, 13]多依赖于领域特定的规则和模板，虽然这些方法在简单的查询上取得了不错的表现，但在面对复杂的跨域查询时常常力不从心。随着深度学习技术的应用，尤其是基于LLMs的few-shot学习方法，研究者在Text-to-SQL任务中取得了突破性进展。这些方法能够在无需微调的情况下，通过精心设计的提示（prompt）和few-shot学习策略，在无监督学习模式下取得较好的性能。

尽管如此，当面对具有挑战性的基准数据集（如Spider [20]）时，这些方法的表现通常还不如经过精细调整（fine-tuning）的大型语言模型。本文提出了一种新的方法，即将Text-to-SQL任务分解为多个小的子任务，通过逐步解决这些子任务来提升模型的推理能力。该方法借鉴了Chain-of-Thought（CoT）和Least-to-Most提示技术，旨在增强模型在复杂SQL查询中的表现。

虽然在原文中报告的准确率达到85.3%，但在本次复现过程中，由于模型版本（从GPT-4切换为GPT-3.5-turbo）和提示设计的差异，性能有所下降，准确率降至50.9%。这一差异反映了模型和提示设计对Text-to-SQL任务结果的关键影响，同时也为未来在不同任务和查询类型下进一步优化该方法提供了有价值的经验。

2 相关工作

Text-to-SQL 是一项将自然语言问题自动转化为 SQL 查询的任务，对于缩小非专业用户与数据库系统之间的差距、提升数据处理效率以及推动智能数据库服务和自动化数据分析等方面具有重要作用。然而，由于自然语言问题理解和 SQL 查询生成的复杂性，Text-to-SQL 仍然是一个具有挑战性的课题。

早期的 Text-to-SQL 系统主要采用基于规则和模板的方法 [11, 17]，这些方法适用于简单的用户查询和数据库，但往往需要大量的人力工程支持和用户交互。此外，为不同场景或领域设计 SQL 模板也充满挑战。近年来，随着数据库和自然语言处理技术的进步，研究者开始探索更为复杂的 Text-to-SQL 方法，推动了基准测试和技术手段的创新。为了缩小研究与实际部署之间的差距，一些大规模的基准数据集被引入，如 WikiSQL [20]，KaggleDBQA [10]，Spider [20]，BIRD [6]。在方法论上，最新的研究将 Text-to-SQL 视为一个序列到序列的任务，并采用编码器-解码器架构来训练机器学习模型 [2]，与此同时，深度学习技术，如注意力机制 [15]，图表示法 [13, 16]，以及语法分析 [7]，在提高 Text-to-SQL 性能方面得到了广泛应用。

随着大型语言模型（LLM）的出现，特别是 OpenAI 的 GPT [18, 19] 和 Meta 的 Llama [8, 21]，自然语言处理和人工智能领域取得了显著进展，这些模型在理解和生成自然语言任务上表现出色。与传统机器学习模型不同，LLM 在大规模文本数据集上进行预训练，使其能够在各种语言相关任务中表现出色。Text-to-SQL 任务也不例外，LLM 在该任务中的应用有望缩小自然语言查询与结构化 SQL 查询之间的差距。当前，在 Text-to-SQL 任务中，主要有两种技术应用于 LLM，第一种是监督微调，该方法通过使用额外的 Text-to-SQL 示例对 LLM 进行优化，以提高其在特定任务上的表现。第二种技术是 in-context learning [4]，也被称为提示工程学 [16] 或 few-shot learning，它专注于在标注数据有限的情况下，使 LLM 能够适应特定的下游任务。在 Text-to-SQL 的背景下，研究人员提出了多种构建提示符的方法，具体包括问题表示方法 [3, 14]，示例选择 [5] 和示例信息组织 [6]，这些技术旨在通过精心设计和展示相关提示来提升 LLM 生成正确 SQL 查询的能力。

3 本文方法

3.1 本文方法概述

尽管在零-shot情况下有所改进，few-shot模型在更复杂的查询上仍然面临困难，尤其是那些模式链接不那么简单的查询，或者包含多个连接和嵌套结构的查询。通过将问题分解为更小的子问题，并逐一解决这些子问题，最终构建出原始问题的解决方案，来解决这些挑战。这种方法借鉴了诸如chain-of-thought提示 [22]和least-to-most提示 [23]等技术，已被证明能够提高LLM在分解成多个步骤的任务中的表现。与这些任务的特点不同，SQL查询主要是

声明性的，大多数步骤之间的界限并不明确。然而，编写SQL查询的思维过程可以被分解为以下几个步骤：（1）检测与查询相关的数据库表和列；（2）识别更复杂查询的一般结构（例如group by、嵌套、多个连接和集合操作等）；（3）根据子问题的解决方案形成任何程序子组件；（4）根据子问题的解决方案编写最终查询。

为了实现这一分解过程，提出了一个包含四个模块的框架（见图1）：（1）模式链接，（2）查询分类与分解，（3）SQL生成，（4）自我校正。这些模块通过提示技术实现，表明LLM在问题被正确分解的情况下能够成功解决所有任务。

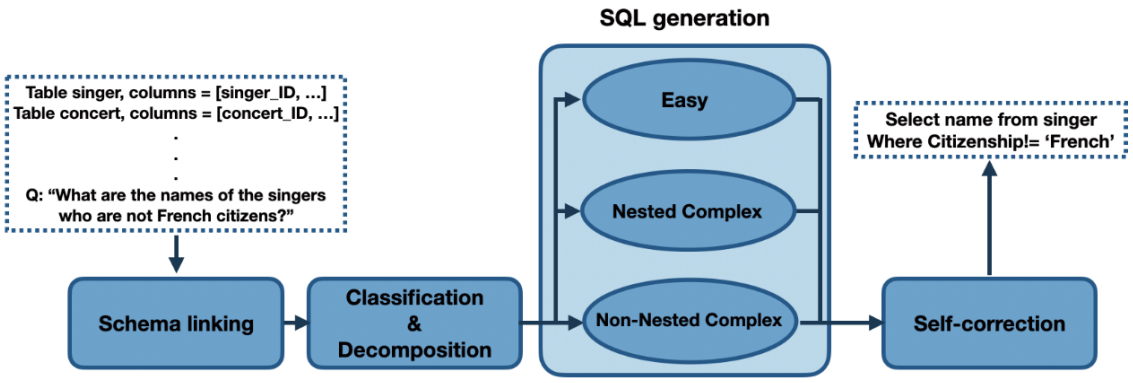

图 1. 整体方法框架

3.2 模式链接

在Text-to-SQL的转换过程中，模式链接（Schema Linking）是一个至关重要的步骤，负责将自然语言查询中的实体与数据库模式中的表、列和条件值进行对应。具体来说，模式链接模块的目标是理解并映射查询中提到的数据库表和字段，并将这些元素与数据库的实际结构相匹配。这一过程不仅有助于模型理解查询的含义，还能有效地处理跨领域的泛化问题和复杂查询的生成。因此，模式链接模块被视为几乎所有Text-to-SQL方法中的核心步骤之一。

模式链接的挑战性主要体现在其需要能够在不同的数据库模式和自然语言查询之间进行有效的映射。自然语言查询中往往没有明确的结构化信息，且不同领域的数据库模式存在显著差异，这使得模式链接的任务更加复杂。例如，在一个查询中可能既包含了表格名称，也提到了一些列名或条件值，而这些元素在数据库中并不总是直接可见或具有明确的对应关系。因此，如何在不同的领域之间进行有效的模式链接，成为了一个非常具有挑战性的问题（如图2所示）。

为了解决这一挑战，设计了一个基提示（Prompt）的模块，这一模块的工作方式是通过提示（Prompting）技术，逐步引导模型理解查询的含义。特别地，在本方法中使用了来自Spider数据集的十个随机选取的查询样本，配合chain-of-thought（思维链）模板进行处理。思维链是一种通过逐步推理引导模型进行决策的方式，每一步推理都是从自然语言查询中提取出一些关键信息，并且帮助模型逐步接近最终的SQL查询。

在具体操作过程中，模型在每一步的提示中首先需要从查询中提取出列名，并将这些列名与数据库模式中的实际列进行匹配。同时，模型还需要从查询中识别出实体（如人名、地点、时间等）以及单元格值（即查询条件的具体数值）。例如，在一个查询中可能会要求列出

所有“2019年销售额超过100万”的记录，在这种情况下，模型不仅需要识别出“销售额”这一列，还需要提取出“2019年”和“100万”这些值，并将其映射到数据库中的具体数据项。

为了使得模型在处理这种复杂查询时能够更加精确，设计了一个详细的提示格式，该格式遵循“逐步思考”原则。具体而言，完整的提示格式如下：“让我们逐步思考”，这个提示引导模型按照指定的步骤进行推理，帮助其解决与数据库模式匹配的问题。每个提示步骤都会清楚地说明该如何从查询中提取信息，并且如何将这些信息与数据库的模式进行对应。例如，在某些情况下，提示可能会要求模型提取查询中的所有表名，并通过一系列步骤将其与数据库的实际表格结构匹配；在另一些情况下，模型则可能需要判断查询中的某些数值是作为条件值还是作为查询结果的一部分来使用。

通过这种逐步推理的方式，模型能够有效地理解并提取自然语言查询中的关键信息，并将其准确地映射到数据库模式中，进而生成正确的SQL查询。这种方法不仅提高了跨领域的泛化能力，也使得处理复杂查询变得更加高效。在实际应用中，这种基于提示的模型尤其适用于处理需要多步骤推理的复杂查询，对于提升Text-to-SQL转换的准确性和鲁棒性有着重要的意义。

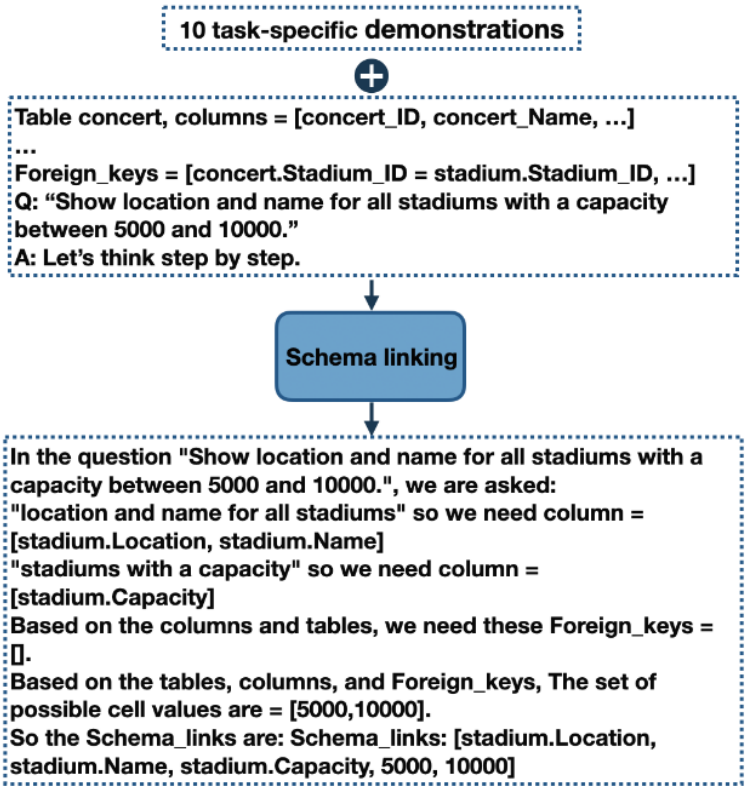

图 2. 模式链接示例

3.3 分类分解模块

对于涉及多个连接的查询，通常难以检测到正确的表或连接条件，尤其是在查询中连接数量较多时。为了解决这个问题，引入了一个分类与分解模块。该模块将查询分为三类：简单查询、非嵌套复杂查询和嵌套复杂查询。简单查询无需连接或嵌套即可完成；非嵌套复杂查询需要进行表连接但不涉及子查询；嵌套复杂查询则可能包括多个表连接、子查询和集合操作。每种类别的查询都会使用不同的提示进行生成。该模块还负责识别非嵌套和嵌套查询

中的连接表和可能的子查询。图 3展示了查询分类与分解模块的输入与输出示例。

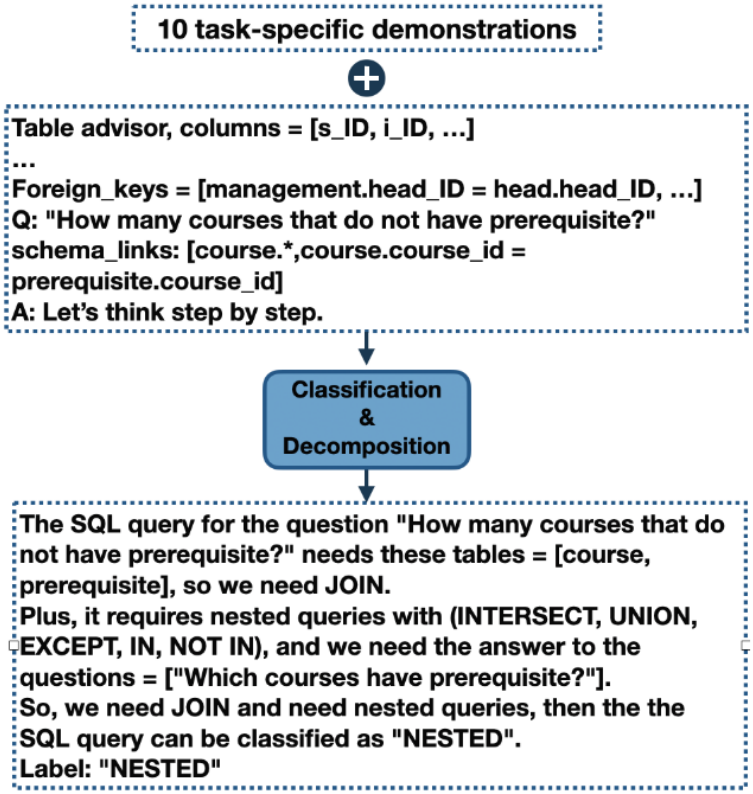

图 3. 分类分解示例

3.4 SQL生成模块

随着查询变得更加复杂，必须引入更多的中间步骤来填补自然语言与SQL之间的差距。为此，采用了三个不同类别的生成方法，来应对简单查询、非嵌套复杂查询和嵌套复杂查询。

对于简单类别的查询，少量的 few-shot 提示足以完成生成任务。这些示例遵循格式 $\langle Q_j, S_j, A_j \rangle$，其中 Q_j 为英文查询，A_j 为 SQL 查询，S_j 为模式链接。

对于非嵌套复杂类别，涉及连接的查询需要通过中间表示来弥合查询与 SQL 之间的差距。使用 NatSQL 作为中间表示，因为它已在其他模型中表现出最先进的性能。非嵌套复杂查询的示例遵循格式 $\langle Q_j, S_j, I_j, A_j \rangle$，其中 S_j 和 I_j 分别是模式链接和中间表示。

最复杂的是嵌套复杂类别，它需要多个中间步骤来生成最终答案。为此，设计的提示格式遵循 $\langle Q_j, S_j, \langle Q_{j1}, A_{j1}, ..., Q_{jk}, A_{jk} \rangle, I_j, A_j \rangle$，其中包含多个子查询 Q_{ji} 和对应的 SQL 答案 A_{ji}，子查询的生成步骤会帮助构建最终的 SQL 查询。

3.5 自我校正模块

生成的SQL查询有时可能会缺少或多余关键字，如DESC、DISTINCT和聚合函数等。为了修正这些小错误，提出了一个自我校正模块。模型在零-shot设置下被要求修正有错误的SQL查询。提供了两种不同的提示来实现这一目标：通用提示和温和提示。通用提示要求模型识别并修正“BUGGY SQL”中的错误，而温和提示则假设查询可能没有错误，而是要求模型检

查潜在问题并给出建议。评估表明，通用提示对CodeX模型更为有效，而温和提示在GPT-4模型上效果更好。在DIN-SQL框架中，默认的自我校正提示对GPT-4设置为gentle，对CodeX设置为generic。

4 复现细节

4.1 与已有开源代码对比

DIN-SQL 方法的代码已经开源，可以在以下网址找到相关资源：https://github.com/MohammadrezaPourreza/Few-shot-NL2SQL-with-prompting。在实验过程中，主要运行的是 DIN-SQL.py 文件。然而，由于模型的更换，原先使用的 GPT-4 被替换为 GPT-3.5-turbo。与 GPT-4 相比，GPT-3.5-turbo 处理的 token 数量较少，因此需要对原始的 prompt 进行删减，以适应新的模型。此外，模型更换后，源代码中解析 GPT 返回结果的部分也出现了问题。原本根据 GPT-4 的输出进行解析的方式无法直接应用于 GPT-3.5-turbo 的返回结果，因此需要对解析逻辑进行修改。为了解决这个问题，我将返回结果的解析部分修改为直接输出 “SELECT”。源代码中并未提供计算最终预测结果的 Python 文件。因此，编写了一个新的脚本，基于执行准确率和精确准确率对模型进行了测试，并计算得到了相应的正确率。

4.2 数据集介绍

评估在跨领域的挑战性Spider数据集上进行，该数据集包含10,181个问题和5,693个独特的复杂SQL查询，涵盖200个数据库和138个领域，每个领域包含多个表。该数据集的标准协议将数据集分为三个部分：8,659个训练样本，分布在146个数据库中；1,034个开发样本，分布在20个数据库中；2,147个测试样本，分布在34个数据库中。每个数据集的数据库是互不重叠的。SQL查询根据SQL关键字的数量、是否包含嵌套子查询、列选择和聚合的使用等因素被分为四个难度级别。由于WikiSQL [9]仅包含单表查询，且查询较为简单，因此未在评估中使用。

4.3 实验环境搭建

本文所使用的Python环境为Python 3.11版本，搭配的GPT-3.5-turbo模型为gpt-3.5-turbo-0613版本。

5 实验结果分析

5.1 实验设计与设置

在复现实验中，采用了原文中相同的数据集Spider，并使用了与原论文相同的评估指标，即执行准确率（EX）和精确集匹配准确率（EM）。选择了与原文相同的模型架构，但为了复现和测试不同的情况，改用了GPT-3.5-turbo而非原文中的GPT-4模型。

此外，在提示（prompt）的设计上，对原始的prompt进行了简化，删除了一些不必要的内容，以测试这种修改对性能的影响。原文中的模型使用了更多的提示信息和复杂的Schema链接模块，而在复现实验中减少了部分提示内容和Schema相关的连接部分。希望通过这种简化，评估模型对少量提示的响应能力。

5.2 实验结果描述

复现实验的主要结果是：在使用GPT-3.5-turbo模型时，执行准确率为50.9%，而原论文中使用GPT-4模型的执行准确率为85.3%。这一结果表明，GPT-3.5-turbo在处理跨领域SQL查询任务时的表现明显不如GPT-4，可能是由于模型规模和处理能力的差异。

此外，精确集匹配准确率（EM）在复现实验中也有较大下降。原文中的模型能够实现高准确率，而在删除部分提示信息后的准确率显著下降，这也表明提示的内容和结构对模型的预测效果有着重要影响。

5.3 删除提示对性能的影响

进一步分析了删除提示内容对模型性能的具体影响。在原文中的模型设置中，提示包含了大量的上下文信息、SQL模式链接和示例查询，这些元素显著提升了模型在生成SQL查询时的准确性。删除这些提示后，模型获得的信息变得较为有限，可能导致模型在生成查询时无法充分理解数据结构和查询需求，进而影响预测结果的质量。

与原文中的方法相比，复现实验未能充分利用数据库模式的链接和更多的上下文信息。这种简化导致了执行准确率的显著下降，尤其是在处理复杂查询和嵌套查询时，GPT-3.5-turbo的表现远不如GPT-4。

5.4 模型规模差异对结果的影响

模型规模的差异也是导致复现实验中准确率降低的一个重要因素。原文使用的是GPT-4模型，该模型的参数规模远大于GPT-3.5-turbo，因此在处理复杂的跨领域任务时，GPT-4能够更好地捕捉到任务的上下文信息并生成更为精确的SQL查询。而GPT-3.5-turbo的能力相对较弱，特别是在缺少充分上下文信息的情况下，生成的SQL查询不如GPT-4准确。

这种差异在执行准确率（EX）上的体现尤为明显，尤其是在处理复杂的多表查询和嵌套查询时，GPT-3.5-turbo由于缺乏足够的上下文支持，无法生成与预期完全匹配的SQL查询，导致了较低的准确率。

6 总结与展望

本文通过对DIN-SQL框架的复现，研究了任务分解方法在提升大语言模型（LLM）在Text-to-SQL任务中的表现中的作用。DIN-SQL框架通过将复杂的SQL查询任务拆解为多个子任务，从而使得模型能够更清晰地理解每一部分的细节并生成更准确的SQL查询。实验结果表明，任务分解和提示设计的合理组合能够显著提升模型的推理准确性，尤其是在复杂查询的处理上。然而，在复现过程中，由于所使用的模型版本和提示策略与原始研究中的有所不同，导致性能未能达到预期的最佳效果，出现了一定程度的下降。

尽管如此，任务分解方法本身展现出了较大的潜力，特别是在处理复杂SQL查询时，它能够有效减少模型推理的难度，提升模型的可靠性和准确性。通过精细的任务分解，模型能够逐步聚焦每个子任务，从而更好地解决复杂问题。因此，任务分解依然是未来研究中的一个重要方向，值得进一步深入探索。未来的研究可以尝试通过更精确的任务分解策略来处理不同类型的SQL查询，探索更适合的提示设计方法，同时优化模型与提示之间的配合，以应对更多样化和更复杂的查询场景。这些改进有望推动Text-to-SQL领域的进一步发展，提升模型的实际应用价值。

参考文献

[1] Xiaodong Liu Oleksandr Polozov Bailin Wang, Richard Shin and Matthew Richardson. Rat-sql: Relation-aware schema encoding and linking for text-to-sql parsers. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7567–7578, 2020.

[2] Ruichu Cai, Boyan Xu, Zhenjie Zhang, Xiaoyan Yang, Zijian Li, and Zhihao Liang. An encoder-decoder framework translating natural language to database queries. *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, pages 3977–3983, 2018.

[3] Shuaichen Chang and Eric Fosler-Lussier. How to prompt llms for text-to-sql: A study in zero-shot, single-domain, and cross-domain settings. 2023.

[4] Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, Lei Li, and Zhifang Sui. A survey for in-context learning. *CoRR*, abs/2301.00234, 2023.

[5] Xuemei Dong, Chao Zhang, Yuhang Ge, Yuren Mao, Yunjun Gao, Lu Chen, Jinshu Lin, and Dongfang Lou. C3: zero-shot text-to-sql with chatgpt. *CoRR*, abs/2307.07306, 2023.

[6] Chunxi Guo, Zhiliang Tian, Jintao Tang, Pancheng Wang, Zhihua Wen, Kang Yang, and Ting Wang. A case-based reasoning framework for adaptive prompting in cross-domain text-to-sql. *CoRR*, abs/2304.13301, 2023.

[7] Jiaqi Guo, Zecheng Zhan, Yan Gao, Yan Xiao, Jian-Guang Lou, Ting Liu, and Dongmei Zhang. Towards complex text-to-sql in cross-domain database with intermediate representation. *Proceedings of the 57th Conference of the Association for Computational Linguistics*, pages 4524–4535, 2019.

[8] Kevin Stone Peter Albert Amjad Almahairi Yasmine Babaei Nikolay Bashlykov Soumya Batra Prajjwal Bhargava Shruti Bhosale Dan Bikel Lukas Blecher Cristian Canton Ferrer Moya Chen Guillem Cucurull David Esiobu Jude Fernandes Jeremy Fu Wenyin Fu Brian Fuller Cynthia Gao Vedanuj Goswami Naman Goyal Anthony Hartshorn Saghar Hosseini Rui Hou Hakan Inan Marcin Kardas Viktor Kerkez Madian Khabsa Isabel Kloumann Artem Korenev Singh Koura Marie-Anne Lachaux Thibaut Lavril Jenya Lee Diana Liskovich Yinghai Lu Yuning Mao Xavier Martinet Todor Mihaylov Pushkar Mishra Igor Molybog Yixin Nie Andrew Poulton Jeremy

Reizenstein Rashi Rungta Kalyan Saladi Alan Schelten Ruan Silva Eric Michael Smith Ranjan Subramanian Xiaoqing Ellen Tan Binh Tang Ross Taylor Adina Williams Jian Xiang Kuan Puxin Xu Zheng Yan Iliyan Zarov Yuchen Zhang Angela Fan Melanie Kambadur Sharan Narang Aurelien Rodriguez Robert Stojnic Sergey Edunov Hugo Touvron, Louis Martin and Thomas Scialom. Llama2: Open foundation and fine-tuned chat models. *CoRR*, abs/2302.13971, 2023.

[9] Wonseok Hwang, Jinyeong Yim, Seunghyun Park, and Minjoon Seo. A comprehensive exploration on wikisql with table-aware word contextualization. *arXiv preprint arXiv:1902.01069*, 2019.

[10] Chia-Hsuan Lee, Oleksandr Polozov, and Matthew Richardson. Kaggledbqa: Realistic evaluation of text-to-sql parsers. *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pages 2261–2273, 2021.

[11] Fei Li and H. V. Jagadish. Constructing an interactive natural language interface for relational databases. *Proceedings of the VLDB Endowment*, pages 73–84, Sep 2014.

[12] Haoyang Li, Jing Zhang, Cuiping Li, and Hong Chen. Resdsql: Decoupling schema linking and skeleton parsing for text-to-sql. *AAAI-23*, pages 13067–13075, 2023.

[13] Jinyang Li, Binyuan Hui, Reynold Cheng, Bowen Qin, Chenhao Ma, Nan Huo, Fei Huang, Wenyu Du, Luo Si, and Yongbin Li. Graphix-t5: Mixing pre-trained transformers with graph-aware layers for text-to-sql parsing. *37th AAAI Conference on Artificial Intelligence*, pages 13076–13084, 2023.

[14] Aiwei Liu, Xuming Hu, Lijie Wen, and Philip S. Yu. A comprehensive evaluation of chatgpt's zero-shot text-to-sql capability. *CoRR*, abs/2303.13547, 2023.

[15] Hu Liu, Yuliang Shi, Jianlin Zhang, Xinjun Wang, Hui Li, and Fanyu Kong. Multi-hop relational graph attention network for text-to-sql parsing. *International Joint Conference on Neural Networks*, pages 1–8, 2023.

[16] Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. What makes good in-context examples for gpt-3? *Proceedings of Deep Learning Inside Out: The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 100–114, 2022.

[17] Tanzim Mahmud, K. M. Azharul Hasan, Mahtab Ahmed, and Thwoi Hla Ching Chak. A rule based approach for nlp based query processing. *2015 2nd International Conference on Electrical Information and Communication Technologies (EICT)*, 2015.

[18] OpenAI. Gpt-4 technical report. *CoRR*, abs/2303.08774, 2023.

[19] OpenAI. Introducing chatgpt. 2023. Available at: https://openai.com/blog/chatgpt, Last accessed on 2023-07-24.

[20] Kai Yang Michihiro Yasunaga Dongxu Wang Zifan Li James Ma Irene Li Qingning Yao Shanelle Roman Zilin Zhang Tao Yu, Rui Zhang and Dragomir R. Radev. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3911–3921, 2018.

[21] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971, 2023.

[22] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

[23] Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, et al. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*, 2022.