计 算 机 研 究 与 发 展
Journal of Computer Research and Development
https://crad.ict.ac.cn
63(5): 1261−1272, 2026

基于大语言模型的 2 阶段迭代模式增强 SQL 生成

张 伟[1,2] 周东傲[3] 官永顺[1] 尹义龙[1]
[1]（山东大学软件学院 济南 250101）
[2]（中国移动通信集团山东有限公司中移齐鲁创新院分公司 济南 250102）
[3]（中国人民解放军军事科学院 北京 100080）
（zhangwei4@sd.chinamobile.com）

Iterative Two-Stage Schema-Enhanced SQL Generation with Large Language Models

Zhang Wei[1,2], Zhou Dong'ao[3], Gong Yongshun[1], and Yin Yilong[1]
[1]（*School of Software*, *Shandong University*, *Jinan* 250101）
[2]（*China Mobile Qilu Innovation Research Institute*, *Jinan* 250102）
[3]（*Academy of Military Science*, *Beijing* 100080）

Abstract With the advent of the big data era, various enterprises and organizations have accumulated massive amounts of data driven by the growing demands for business expansion and information. This data are typically stored in relational databases in structured or semi-structured formats. SQL, as a structured database query language, has long been widely used for data retrieval and processing. It provides professionals with an efficient way to interact with databases, thereby facilitating quicker and more convenient data sharing and analysis. With the exponential advancement of large language models (LLMs), in-context learning (ICL) has shown significant promise for the text-to-SQL task, where LLMs utilize reference examples to accurately generate SQL queries. To fully harness the potential of LLMs through ICL for text-to-SQL, we propose a new SQL generation pipeline comprising three parts: first, we propose a schema enhancement component to strengthen LLMs' focus on question-related information. Second, we leverage the inherent structure of SQL to preselect semantically similar, aiding in the generation of initial pseudo-SQL queries. Third, a selection strategy is proposed for selecting reference examples, which considers the similarities between the question and the pseudo-SQL structure to enhance the accuracy of the generated SQL queries. We implement an iterative refinement process in both stages, employing a boosting-like mechanism that progressively improves selection until stabilization is achieved. Extensive experiments across various models and datasets demonstrate the effectiveness of our approach.

Key words large language models; text-to-SQL; in-context learning; database management; data analysis

摘 要 随着大数据时代的到来，各种企业和组织在日益增长的业务拓展以及信息需求的推动下积累了海量的数据。这些数据通常以结构化或半结构化的形式存储在对应的关系型数据库中。SQL 作为一种结构化的数据库查询语言，长期以来被广泛应用于数据的检索和处理。它为相关专业技术人员提供了一种高效的方式来与数据库进行交互，从而能够更快捷方便地实现对于数据的分析。随着大语言模型（large language models, LLMs）的快速发展，上下文学习（in-context learning, ICL）在文本生成 SQL（text-

收稿日期：2025-05-26；**修回日期**：2025-12-08
基金项目：国家自然科学基金项目（62476154）
This work was supported by the National Natural Science Foundation of China (62476154).
通信作者：周东傲（zhoudongao08@nudt.edu.cn）

to-SQL）任务中展现出巨大潜力。ICL能够使大语言模型通过利用参考示例生成准确的SQL查询。为了充分利用大语言模型在ICL机制下进行文本生成SQL,提出了一个新的SQL生成流程,该流程包括3个部分:首先,提出了迭代式的模式增强,以强化大语言模型对解决问题相关信息的关注。其次,利用SQL的骨架结构进行语义相似度筛选,从而辅助生成伪SQL（pseudo-SQL）查询。最后设计了一种参考示例选择策略,该策略结合问题与伪SQL骨架之间的相似性,以提高生成SQL查询的准确性。同时,在2个关键阶段中都引入了迭代优化过程,以逐步优化示例选择策略,直至达到稳定状态。在多个大语言模型和数据集上进行了广泛的实验,验证了所提出方法的有效性。

关键词 大语言模型;文本生成SQL;上下文学习;数据库管理;数据分析

中图法分类号 TP391

DOI: 10.7544/issn1000-1239.202550355 **CSTR:** 32373.14.issn1000-1239.202550355

关系型数据库在数据爆炸式增长的时代中对于数据的存储与处理扮演着不可或缺的角色。随着社会信息化进程的不断加快,text-to-SQL任务的重要性日益凸显,其核心目标在于将自然语言问题转换为可执行的SQL查询语句,实现对数据库的高效访问与操作,从而实现数据仓库的高效利用[1]。随着数据量的增长以及数据交互频率的不断增加,传统的数据存储与查询方式在实际应用中已经捉襟见肘。如何在海量数据中快速有效地提取有价值的信息并且保证查询的实时性和准确性已经成为在数据处理分析的时候需要解决的重要问题[2]。SQL作为一种被广泛使用的结构化查询语言,为数据分析领域的技术人员提供了高效的数据库交互方式,从而能更便捷地实现数据的共享与分析,助力各行各业的高效运行与决策支持[3]。

但是对于绝大多数非技术人员来说,编写语法正确且能够满足查询目的的SQL语句仍然存在着比较高的学习门槛。因此文本转化为SQL任务的重要性日益凸显,它能够将自然语言问题准确地转换为可执行的SQL命令[4]。这一技术使得用户能够依据自身需求来轻松访问关系型数据库中的指定数据,使用户无需精通复杂的SQL语法便可以高效地进行数据操作[5]。与此同时,大语言模型的兴起为自然语言处理领域的发展带来了革命性的突破[6]。得益于在大规模数据集上进行了广泛的预训练,大语言模型在语言理解与生成方面展现了卓越的能力[7]。这一发展为文本转化SQL任务的研究开辟了全新的方向。在仅提供数据库模式信息和自然语言问题的情况下,大语言模型生成的SQL语句结果相比传统神经网络能够实现更高的执行正确率[8]。这不仅表明大语言模型的预训练语料和数据中已包含了丰富的文本到SQL任务信息,更凸显了其在文本到SQL应用中的巨大潜力[9]。

尽管大语言模型在文本到SQL任务中展现出强大的能力,但其在实际应用中仍面临诸多挑战。传统的神经网络通常会针对数据库模式链接任务进行专门训练,但大语言模型在预训练阶段可能并未专门针对该任务进行优化[10]。此外,在实际应用中,不同数据库的模式结构信息分布往往差异比较显著,并且通常充斥着各种冗余复杂信息。过多的无关信息可能会干扰大语言模型对于输入数据库信息的理解,导致最后输出错误的SQL语句。因此,仅仅依靠大语言模型从预训练的海量自然语言数据中获得的通用知识来解析未知陌生数据库,往往容易引发理解偏差或输出错误。

大语言模型对不同的上下文示例选择以及关键的思考提示非常敏感,这两者都能显著影响其生成输出的质量[11-13]。直观来讲,一种方法是优化上下文示例的选择和组织,确保它们更具代表性且与特定的文本到SQL任务的上下文相关[14]。其中一个重要的方面是参考示例中自然语言问题的相似性,以及相应SQL骨架的指导作用[15]。此外,在实际应用中,数据库模式信息通常非常复杂且冗余,直接解释和回应自然语言问题时容易出错。因此,在使用大语言模型生成SQL语句时,强调数据库模式中与接收到的自然语言问题真正相关的特定表和列至关重要。

鉴于上述考虑,本文提出了一种新的方法,称为模式增强的迭代式2阶段SQL生成(iterative two-stage schema-enhanced SQL generation, ITS-SQL)。ITS-SQL由3个关键组件组成:模式增强、伪SQL生成和最终SQL生成。首先,我们通过大语言模型由基本的数据库信息生成初步的“粗糙SQL”语句来从数据库模式中筛选与输入自然语言查询语义相关的数据库表和列。随后,考虑到SQL本身具有的较强结构

化特性和明确语义，我们设计了一个迭代优化机制，通过逐步调整使参考示例在SQL骨架方面与模型的推理结果更加契合。这种迭代优化过程不仅使大语言模型能够从相似示例中学习有效的生成模式，还显著提升了模型的泛化能力。为进一步增强模型在跨领域场景下的适应性，我们引入了问题文本与SQL骨架关键词之间的相似度评估机制，并基于相似度评分对示例进行优化选择，为大语言模型提供更具针对性的参考示例，从而引导模型输出最终的SQL查询语句。

本文的贡献主要包括3个方面：

1）针对模式增强模块进行了改进，以突出与自然语言问题相关的数据库信息。这种方法避免了传统模式过滤可能导致关键信息丢失的风险，同时增强了大语言模型对关键细节的关注。

2）提出了一种新的样本选择方法，并结合迭代优化过程，选择出更适合作为参考的示例样本。通过这种方法，我们确保了样本选择的稳健性和有效性，从而最大化大语言模型在更复杂的text-to-SQL数据集上的执行正确率。

3）在2个大规模跨领域数据集上进行了广泛的实验。证明了该方法对模型和数据集具有普遍适用性，无论是非开源大语言模型还是开源大语言模型，该方法都能显著提升在不同数据集上的执行正确率，验证了方法的有效性。

1 相关工作

随着大语言模型的快速迭代发展，尤其是像GPT[16]，Gemini[17]，LLaMA[18]，DeepSeek[19]系列先进大语言模型的出现，大语言模型已成为自然语言处理领域的重要研究和应用方向之一。大语言模型凭借其广泛的综合知识储备已经在许多NLP任务中展现出了极其强大的语言理解能力[20]，同时也因为其强大的生成能力成为了text-to-SQL任务研究中的关键技术。Pourreza等人[21]通过任务分解（task decomposition）的少样本思维链提示引导以及自我纠正辅助大语言模型进行显式的数据库模式链接推理，展示了大语言模型在文本到SQL任务中的强大潜力。Gao等人[22]系统地对现有的基于大语言模型的文本到SQL方法进行了分类和比较，并且重点分析了提示中的问题表示方式和示例组织策略，并且通过搜索具体数据库的实体信息来针对性对自然语言问题和SQL语句进行实体掩盖，从而实现跨知识领域的相似度示例选择。Dong等人[23]通过符号化表结构描述（如分隔指令、上下文和问题）来让模型对与问题相关的多个表进行排序选择，使得模型完成对于数据库模式链接方式的理解，并且通过多路投票机制，生成多个SQL候选并基于执行结果选择最优解，显著提高了生成SQL的鲁棒性。Wang等人[24]提出了一种多代理（multi-agent）协作框架，将文本到SQL任务分解为多个子任务，并且分别由不同的代理负责处理，以提高生成复杂SQL查询的能力。模式链接方面专门微调了用于数据库表和列选择的小规模语言模型作为代理，并且代理之间通过共享信息和协作机制，确保最终生成的SQL查询具备一致性和准确性。Qu等人[10]系统性地对文本到SQL生成中的幻觉（hallucination）问题进行分类，并且通过使用数据分析中Pandas库类比的方式来辅助分析数据库模式关系，优化了大语言模型的潜在模式链接过程。这些方法通常借鉴了基于传统神经网络训练的方式，通过引入额外的中间模式链接（schema linking）阶段，使大语言模型能够将数据库结构映射到自然语言。

2 算法流程

2.1 数据库模式增强

模式链接是将自然语言问题转换为SQL过程中的关键环节，其核心目标在于建立自然语言表述与数据库模式元素（如表、列、关系等）之间的语义映射关系。以往的研究普遍采用模式过滤（schema filtering）或模式修剪（schema pruning）技术来优化模式链接过程[21]，即通过移除无关的数据库元素来简化模式信息，从而提高模型的表现效果。然而，近期研究表明，对于具备较强上下文理解能力的大语言模型，显式的模式过滤策略可能收效甚微，甚至可能对模型性能产生负面影响[25-26]。这种负面影响主要源于模型在模式过滤过程中可能误判并删除必要的列或表信息，导致生成的SQL查询因关键信息缺失而出现错误。

针对这一问题，本文提出了一种模式增强（schema enhancement）方法：首先，我们直接将完整的数据库模式信息与自然语言问题共同输入模型，使模型能够基于完整的上下文信息生成一个“粗糙SQL（imprecise SQL）”。其次，我们引入基于示例的实体解析机制。通过提供少量示例，引导大语言模型使用思维链方式从生成的“粗糙SQL”中提取出与数据库相关的实体信息，形成数据库增强提示。最

后，我们在保留完整数据库结构信息的基础上，将提取的实体信息以增强提示的形式重新注入模型，处理过程如图 1 所示。

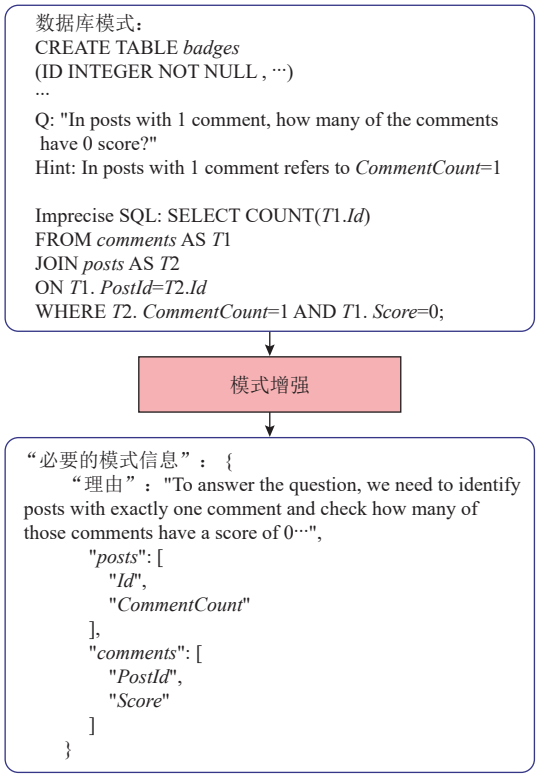


Fig. 1 Database schema enhancement

图 1 数据库模式增强

在传统的模式过滤方法中，模型可能会误删关键的数据库表列信息，从而导致后续生成错误的 SQL 语句。如图 2 所示，由于语言模型对问题语义及数据库模式的理解不准确，错误地删除了关键字段“MailStreet”，导致最终生成的 SQL 语句仅返回了“School”列的信息。相比之下，本文提出的模式增强策略能够引导模型基于初步生成的粗糙 SQL 和完整的数据库模式信息，提取与任务相关的实体。在粗糙 SQL 中潜在逻辑结构与全局模式信息的共同提示下，模型能够更准确地理解任务目标，并更有效地补全问题中涉及的实体字段，从而提升 SQL 生成结果的正确性。

2.2 伪 SQL 产生阶段

在 2.1 节中，我们仅利用了“粗糙 SQL”中的数据库实体信息部分作为数据库表和列的提示信息。在本节中，我们将利用“粗糙 SQL”的骨架结构来进行进一步的扩展。对于文本到 SQL 任务中的上下文示例选择，传统方法通常直接对比输入问题相似度来选择“问题-SQL 对”作为参考示例。然而，这种方法存在显著局限性：尽管某些问题在语义上可能非常相似，但由于数据库结构的差异或查询意图的表达差别，其对应的 SQL 查询语句可能截然不同，如图 3 所示。

为了解决这一问题，我们提出了一种基于“伪 SQL(pseudo SQL)”生成的示例选择方法。在模式增强阶段生成的“粗糙 SQL”虽然可能不够精确，但其骨架结构能够有效捕捉问题的核心意图和数据库操作的逻辑框架。

所以在这一步，我们利用生成的“粗糙 SQL”来指导示例选择。具体而言，我们将“粗糙 SQL”和候选集中的 SQL 语句均进行 SQL 语法关键字保留处理，将包含的实体信息均通过使用特殊标记〈unk〉来进行替换从而避免实体信息影响语义相似度计算。然后通过计算处理后的“粗糙 SQL”与候选集中 SQL 语句的语义相似度，选择相似度最高的“问题-SQL 对”作为参考示例，并且通过迭代优化生成该步的“伪 SQL”结果。为了确保“伪 SQL”生成的稳定性，我们在每次迭代中动态更新示例选择策略，确保所选示例能够逐步逼近最优解。

值得注意的是，“粗糙 SQL”主要用来提示大语

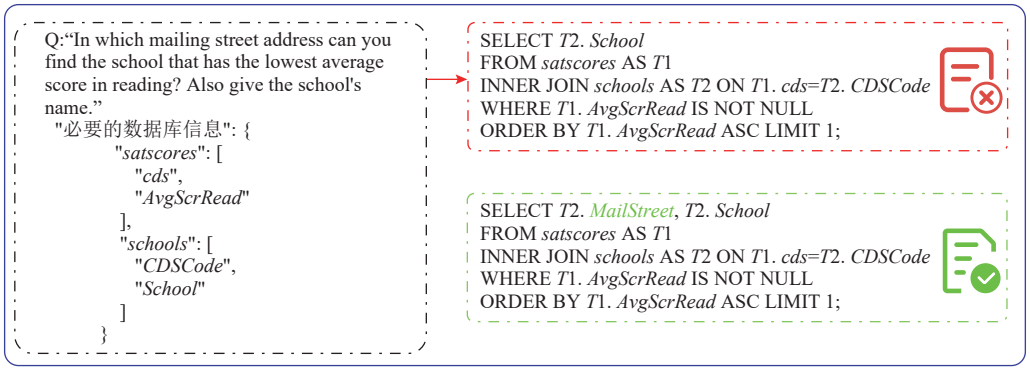


Fig. 2 Filtration mode deletes key information by mistake, resulting in wrong results

图 2 模式过滤误删关键信息导致错误结果

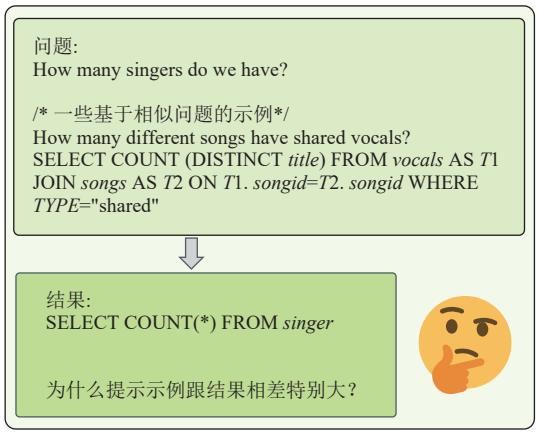


Fig. 3 Misleading example prompt

图 3 误导性的示例提示

言模型对应的数据库实体信息以及初步骨架，而“伪 SQL”则主要用于最终 SQL 产生阶段中与问题语义共同选择参考示例来使大语言模型产生最终的结果。

传统方法通常依赖于 NLTK 的实体识别[27]或从数据库中检索相关信息[22]来屏蔽无关实体数据，从而实现跨领域扩展。然而这些方法存在明显的局限性：首先，由于预训练语料库的覆盖范围有限，使用 NLTK 等方式在处理未知领域数据时，容易出现识别错误；其次对于大型数据库，检索相关实体信息的计算开销变得不可行，严重影响了方法的实用性。我们摒弃了传统的实体屏蔽策略，转而采用正则表达式匹配从“伪 SQL”中提取并保留关键词和符号。

在实现上，我们摒弃了传统的基于 n-gram[28]的 SQL 相似度计算方法，转而采用基于预训练语言模型的通用文本嵌入（general text embedding, GTE）模型[29]来处理 SQL 语句。这一选择基于 2 点考虑：首先，SQL 语法与自然语言在结构上具有高度相似性，尤其是在操作语义的表达上具有明确且可读性强的特点[30]；其次，GTE 能够更好地捕捉 SQL 语句的语义信息，而不仅仅是表面的语法结构。我们通过计算 GTE 嵌入向量的余弦相似度来量化 SQL 语句之间的相关性，从而选择最相关的示例对。

2.3 最终 SQL 产生阶段

在最终 SQL 产生阶段，为了进一步减轻问题中保留的实体对语义相似性计算的干扰，我们提出了一种基于 SQL 骨架和问题双重相似性的重叠选择策略。我们首先根据自然语言问题和在 2.2 节中产生的“伪 SQL”骨架的相似性分别对训练集数据进行排序，生成 2 个有序集合：O_{question}（基于问题相似性）和 O_{sql}（基于 SQL 骨架相似性）。然后初始化 2 个空集合 C_{question} 和 C_{sql}，并依次从 C_{question} 和 C_{sql} 中按照相似性排名将对应的“问题-SQL 对”添加到 2 个集合中，直到 C_{question} 和 C_{sql} 的交集 C_{overlap} 中的元素数量达到或超过预定义的参考示例数量 k（在少样本实验参数设置中指定）。

然而，上述示例选择策略面临一个与“伪 SQL”生成过程中类似的问题：我们是否通过选择最合适的样本充分挖掘了模型的少样本学习潜力？为了确保使用该方法能够获得稳定且最优的 SQL 结果，我们同样迭代优化示例选择过程。在每次迭代中，我们将前一次推理得到的 SQL 结果作为下一次示例选择的“伪 SQL”输入。整个方法流程如图 4 所示。

3 实验及分析

3.1 数据集

为了更好地评估本文所提出的 ITS-SQL 具体应用的有效性，我们在当前比较流行的 2 个大型数据集 Spider[7]和 BIRD[31]上进行了实验对比与分析。Spider 数据集包含 10 181 个问题和 5 693 个独特的复杂 SQL 查询，分布在 200 个数据库中。其中 140 个数据库被划归为训练集，20 个数据库作为验证集，40 个数据库则构成测试集。BIRD 数据集包含超过 12 751 对独特的“问题-SQL 对”和 95 个大型数据库，而且其引入了诸如外部知识、脏数据等新挑战。在数据集划分方面，训练集包含 69 个数据库，验证集包含 11 个数据库，测试集包含 15 个数据库。由于地区的限制，我们仅在 BIRD 的验证集上进行实验，因为测试集需要提供 OpenAI API 密钥给 BIRD 官方数据集才可以测试。

对于开源模型，本文使用 vLLM 框架加速推理。在文本嵌入模型方面，本文选择了 GTE-Large 来将自然语言问题和 SQL 嵌入到 1 024 维向量中。GTE 使用了多阶段对比学习方法，首先在大量无监督文本对上进行预训练，然后在多个任务的标注文本三元组上进行微调，从而模型具有较高的泛化能力。

在具体实验中，本文使用了 CREATE TABLE 语句来表示数据库的模式信息，包括表名、列名、主键、外键等。为了实现不同大语言模型的公平比较，对比实验基准数据均来自于数据集官方网站公示结果以及对应发表论文。

3.2 实验设置

本文统一将所有模型的温度参数值设置为 0（即关闭采样随机性，使模型生成结果更具确定性），以消除随机性对实验结果的干扰；同时，将最大生成令

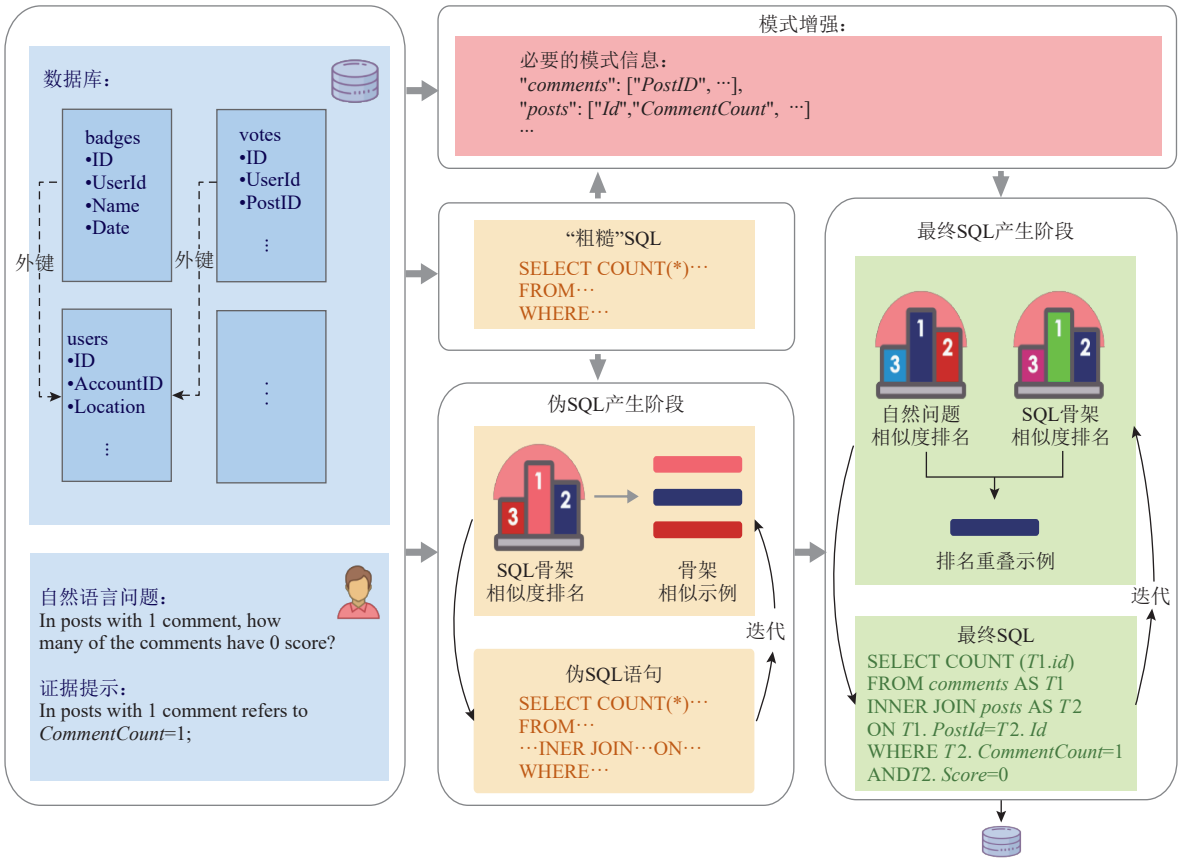


Fig. 4 Flowchart of ITS-SQL

图 4 ITS-SQL 流程图

牌数设定为 512。需要说明的是，若未明确标注示例数量的实验部分，则默认采用 5-shot 设置。

3.3 Spider 实验结果

在表 1 中，我们展示了模型在 Spider 的验证数据集上，采用不同数量的示例选择方法作为提示示例时的执行正确率。结果证明，我们的示例选择方法在帮助模型生成正确的 SQL 查询方面，比其他的传统示例选择策略更为有效。几乎所有模型在使用更多示例参考的时候性能都会普遍提高，这显示了在适当范围内提供更多示例能有效提升模型在该任务上的适应能力。其中 ITS-SQL 的表现始终高于其他策略，值得注意的是，Qwen2.5-Coder-32B-Instruct 在 ITS-SQL 策略下取得了高达 86.8% 的执行正确率。

值得指出的是，在众多其他自然语言相关任务表现优异的 DeepSeek-V3 却并没有在本任务中取得理想的效果。本文推测这可能跟 text-to-SQL 任务本身的特点以及 DeepSeek-V3 的 MoE(mixture of experts)架构有关。MoE 架构通过集成多个“专家”网络来提高模型的表达能力和处理复杂任务的能力。每个“专家”网络通常是一个小型神经网络，专门处理输入数据的一个子集或特定类型的任务。数据库和自然语言问题通常具有很强的跨域性，同时生成的 SQL 语言又具有很强的结构性，所以在面对此类任务时，MoE 架构可能因专家路由机制未能分配到最合适的子网络，导致其 SQL 生成能力相较于编程语言模型存在劣势。尽管 DeepSeek-V3 的总参数量高达 671B，但在“专家”网络分配后，其实际激活参数量仅为 37B，从而影响模型表现。

除此之外，本文也使用了 Qwen2.5-Coder-32B-Instruct 和 GPT-4o 来与之前基于上下文学习的前沿方法进行了对比。如表 2 所示，本文提出的方法取得了显著的改进。

3.4 BIRD 实验结果

我们进行了不同开源模型及 GPT-4o 执行 ITS-SQL 方法时针对不同难度查询的实验效果对比。结果表明，采用 ITS-SQL 方法显著提高了难度级别查询的执行正确性。具体而言，在 CodeLlama 的结果中，整体执行正确率提升了 12.91 个百分点，其中简单查询的提升最为显著，提升了 16.54 个百分点，而困难难度的查询则提升了 14.58 个百分点。虽然 DeepSeek-Coder 在基础模型测试中的表现尚佳，但其整体执行正确率仅提升了 2.68 个百分点。其中中等难度查询

Table 1 Execution Accuracy of Large Language Models on Spider-dev Dataset Under Different Few-Shot Example Selections

表 1 各大语言模型在不同少样本示例选择下 Spider-dev 数据集上的执行正确率 %

样本数量	样本选择策略	CodeLlama	DeepSeek	DeepSeek-V2	Qwen2.5	Llama3	Qwen2	DeepSeek-V3
0-shot		71.0	72.6	75.0	83.0	75.5	79.2	78.6
1-shot	随机	69.5	74.2	75.0	85.0	78.0	81.8	80.4
	问题语义相似度	69.4	75.6	76.2	84.8	78.2	79.6	81.1
	掩码问题相似度	69.6	76.5	75.8	86.0	78.2	79.8	81.2
	SQL 语义相似度	69.9	74.1	75.3	84.8	79.0	80.7	81.4
	SQL 骨架相似度	71.6	75.1	75.6	84.2	80.0	80.8	81.2
	ITS-SQL（本文）	71.9	78.7	76.8	85.8	82.4	83.4	82.3
3-shot	随机	70.6	75.4	74.0	85.4	79.4	80.9	80.5
	问题语义相似度	70.9	76.1	76.5	84.3	78.7	80.9	82.3
	掩码问题相似度	69.4	75.6	76.9	85.4	78.2	79.6	82.4
	SQL 语义相似度	72.0	76.2	75.3	84.4	78.8	82.5	80.2
	SQL 骨架相似度	72.3	75.1	75.1	85.1	78.8	81.8	80.9
	ITS-SQL（本文）	73.4	80.8	77.2	86.3	82.9	84.4	82.5
5-shot	随机	72.9	76.2	75.1	85.5	78.9	80.8	80.3
	问题语义相似度	71.5	76.7	76.3	84.6	78.7	81.0	82.4
	掩码问题相似度	69.4	75.6	77.1	84.8	78.2	79.6	81.8
	SQL 语义相似度	72.2	77.1	74.4	85.1	78.9	82.0	81.8
	SQL 骨架相似度	72.0	75.3	74.7	85.3	78.7	81.1	82.3
	ITS-SQL（本文）	73.9	80.7	77.7	86.8	82.9	84.5	83.1

Table 2 Execution Accuracy on Spider

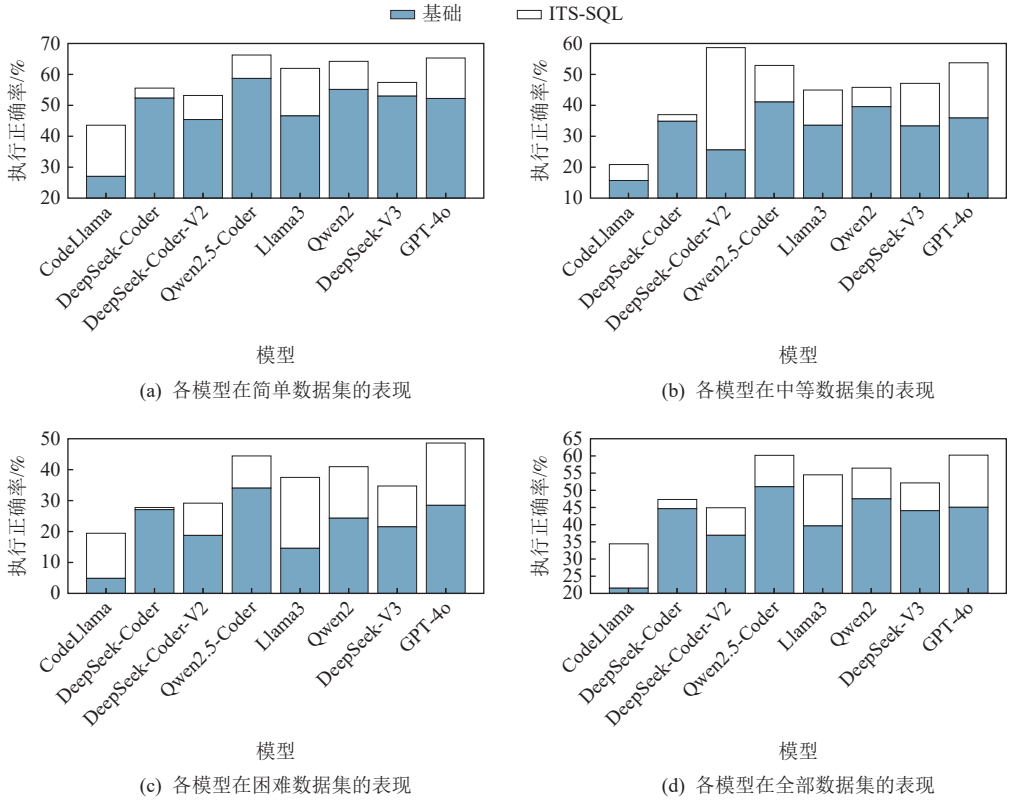
表 2 在 Spider 上的执行正确率 %

方法	验证集	测试集
T5-3B	71.5	74.4
RAT-SQL	75.0	73.3
RATSAT+PICARD	80.5	75.5
Graphix-T5	81.0	77.6
RESDSQL-3B	84.1	79.9
Codex	71.7	72.1
PaLM-2	69.9	70.5
ChatGPT	72.8	74.3
COT+ChatGPT	75.5	76.7
GPT-4	76.6	83.9
COT+GPT-4	78.2	83.9
C3+ChatGPT	81.8	82.3
DAIL-SQL+GPT-4	82.4	86.6
DINSQL+GPT-4	82.8	85.3
GPT-4o	77.8	82.8
ITS-SQL+GPT-4o（本文）	85.8	86.9
ITS-SQL+Qwen2.5-Coder（本文）	86.8	87.5

的执行正确率相对有较大幅度提高。

可能由于专注于编程方面且发布版本较早，Code-Llama 和 DeepSeek-Coder 未能在模式增强步骤中准确提取并输出表和列信息，这种不准确性可能导致后续 SQL 生成过程中的偏差。Qwen2.5-Coder 尽管初始执行正确率已经高达 51.04%，但仍在中等和困难查询上取得了显著进步，执行正确率分别增加了 11.82% 和 10.41%，最终实现了 60.17% 的整体执行正确率。Llama3 的执行正确率整体提升了 14.87 个百分点，其中简单难度查询的执行正确率增长了 15.36 个百分点，困难查询的则增长了 22.92 个百分点，均显示出明显的改进。Qwen2 的总平均提升为 8.93 个百分点，尤其在处理困难难度查询时，执行正确率显著提升了 16.66 个百分点。最后，GPT-4o 在所有查询难度上都取得了出色表现，整体执行正确率提升了 15.18 个百分点，最终达到了 60.23 个百分点的执行正确率。具体改进结果如图 5 所示。

如表 3 所示，本文将所提出的方法与其他先前基于上下文学习的方法进行了系统性实验对比。我们基于 BIRD 的验证集部分开展具体执行正确率的对比分析，为了保证对比的全面性，我们选取了表现较好且具有代表性的 Qwen2.5-Coder-32B-Instruct 和 GPT-4o 作为基线模型进行实验对比。实验结果表明，当本文所提出的 ITS-SQL 方法与 GPT-4o 结合使用时，



(a) 各模型在简单数据集的表现 (b) 各模型在中等数据集的表现

(c) 各模型在困难数据集的表现 (d) 各模型在全部数据集的表现

Fig. 5 Improving performance on different difficulty levels of BIRD dataset

图 5 在 BIRD 数据集不同难度部分的性能改进

在 BIRD 的验证集上取得了 60.23% 的执行正确率，显著优于所对比的其他基于上下文学习的方法。

Table 3 Execution Accuracy on BIRD Validation Set

表 3 在 BIRD 验证集上的执行正确率 %

方法	执行正确率
RESDSQL-3B	43.9
T5-3B	23.34
Palm-2	27.38
Codex	34.35
ChatGPT	37.22
ChatGPT+COT	36.64
Claude-2	42.70
GPT-4	46.35
C3+ChatGPT	50.2
DIN-SQL+GPT-4	50.72
DAIL-SQL+GPT-4	54.76
TA-SQL+GPT-4	56.19
ITS-SQL+Qwen2.5-Coder（本文）	60.17
ITS-SQL+GPT-4o（本文）	60.23

值得注意的是，本文方法不仅在非开源语言模型（如 GPT-4o）上表现优异，在性能相对较弱的开源模型中也展现出显著优势，充分体现了其模型无关的特性。特别值得关注的是，本文方法在困难难度 SQL 查询任务上的改进幅度最为显著，显示出解决复杂问题的强大能力。综合实验结果可以得出，ITS-SQL 展现出卓越的稳健性、有效性和适应能力，这种优势在不同基准性能水平的模型上均得到稳定体现，充分验证了方法的泛化性。

3.5 消融实验结果

为系统验证本文方法的有效性，我们针对各部分设计了对应的消融实验，围绕 3 个核心模块展开评估：模式增强、伪 SQL 生成和最终 SQL 生成。如表 4 和表 5 所示，我们选用了 Spider 和 BIRD 的验证集，并且在所选择的开源大语言模型上完成消融实验。

需说明的是，表中“上限”指在最终 SQL 生成阶段采用对应真实值来当作伪 SQL 提取 SQL 骨架的理想情况，以此来方便展示 ITS-SQL 应用的上限，但是我们并没有将真实值直接输入模型，而是仅作用于实验阶段的示例筛选过程，因此不存在数据泄露的问题。

3.6 详细案例研究

为了更具体地展示 ITS-SQL 的有效性，本文选取了 3 个案例以说明 SQL 结果的改进，如图 6 所示，

Table 4 Execution Accuracy of Module Ablation study of Different Models on Spider Dataset

表 4 各模型在Spider数据集上的模块消融实验执行正确率 %

模型	w/o 伪 SQL	w/o 最终 SQL	w/o 模式增强	ITS-SQL（本文）	上限
CodeLlama	73.8	73.0	73.9	74.5	76.7
DeepSeek-Coder	80.4	78.3	78.7	80.7	82.7
DeepSeek-Coder-V2	77.1	74.4	75.7	77.7	79.5
Qwen2.5-Coder	85.4	85.1	85.3	86.8	88.1
Llama3	81.2	80.6	80.7	82.9	85.0
Qwen2	83.2	82.1	84.0	84.5	85.8
DeepSeek-V3	82.3	82.4	82.1	83.1	84.5

Table 5 Execution Accuracy of Module Ablation study of Different Models on BIRD Dataset

表 5 各模型在BIRD数据集上的模块消融实验执行正确率 %

模型	w/o 伪 SQL	w/o 最终 SQL	w/o 模式增强	ITS-SQL（本文）	上限
CodeLlama	34.03	33.77	34.32	34.42	36.44
DeepSeek-Coder	46.28	45.76	47.20	47.33	49.67
DeepSeek-Coder-V2	43.02	39.77	44.33	44.92	46.22
Qwen2.5-Coder	59.45	57.37	57.91	60.17	60.82
Llama3	50.98	50.85	51.24	54.50	56.39
Qwen2	54.30	53.98	54.95	56.45	58.28
DeepSeek-V3	51.04	48.89	51.50	52.15	53.72

案例1 california_schools

问题 What is the free rate for students between the ages of 5 and 17 at the school run by Kacey Gibson?

真实值 SELECT CAST(*T2*.`*FreeMealCount* (*Ages* 5-17)`AS *REAL*)/*T2*.`*Enrollment* (*Ages* 5-17)`FROM *schools* AS *T*1 INNER JOIN *frpm* AS *T*2 ON *T*1. *CDSCode*=*T2*. *CDSCode*WHERE *T*1. *AdmFName1*='*Kacey*'AND *T*1. *AdmLName1*='*Gibson*'",

GPT-4o SELECT *s*. *School*, *f*. `*FreeMealCount* (*Ages* 5-17)`, *f*. `*Enrollment* (*Ages* 5-17)`, (*f*. `*FreeMealCount* (*Ages* 5-17)`/ *f*. `*Enrollment* (*Ages* 5-17)`)AS *FreeRate* FROM *schools s* JOIN *frpm f*ON *s*. *CDSCode*=*f*. *CDSCode* WHERE *s*. *AdmFName*1='*Kacey*' AND *s*. *AdmLName*1='*Gibson*'

ITS-SQL（本文） SELECT `*FreeMealCount* (*Ages* 5-17)`/`*Enrollment* (*Ages* 5-17)`AS *free_rate* FROM *frpm* JOIN *schools* ON *frpm*. *CDSCode*=*schools*. *CDSCode* WHERE *schools*. *AdmFName*1='*Kacey*'AND *schools*. *AdmLName*1='*Gibson*'

案例2 toxicology

问题 Among the molecules between TR004 to TR010, how many of them has single bonds?

真实值 SELECT COUNT (DISTINCT *T*. *molecule_id*) FROM *bond* AS *T* WHERE *T*. *molecule_id* BETWEEN '*TR*004'AND '*TR*010' AND *T*. *bond_type*='-'

GPT-4o SELECT COUNT (DISTINCT *m*. *molecule_id*) AS *molecule_count* FROM *molecule m* JOIN *bond b* ON *m*. *molecule_id*= *b*. *molecule_id* WHERE *m*. *molecule_id* BETWEEN '*TR*004' AND' *TR*010' AND *b*. *bond_type*= '-'

ITS-SQL（本文） SELECT COUNT (DISTINCT *b*. *molecule_id*) FROM *bond* AS *b* WHERE *b*. *bond_type*= '-' AND *b*. *molecule_id* BETWEEN '*TR*004' AND '*TR*010'

案例3 codebase_community

问题 What is the detailed content of the comment of the post which was created on 7/19/2010 7:37:33 PM?

真实值 SELECT *T*1. *Text* FROM *comments* AS *T*1 INNER JOIN *posts* AS *T*2 ON *T*1. *PostId*=*T*2. *Id* WHERE *T*1. *CreationDate*= '2010-07-19 19:37:33.0'

GPT-4o SELECT *c*. *Text* AS *CommentText* FROM *posts p* JOIN *comments c* ON *p*. *Id*=*c*. *PostId* WHERE *p*. *CreationDate*='2010-07-19 19:37:33.0'

ITS-SQL（本文） SELECT *T*. *Text* FROM *comments* AS *T* INNER JOIN *posts* AS *P* ON *P*. *Id*=*T*. *PostId* WHERE *T*. *CreationDate*= '2010-07-19 19:37:33.0';

Fig. 6 Detailed case study

图 6 详细案例研究

这 3 个示例代表了 SQL 任务中的常见错误：属性理解错误、选择错误的数据表以及数据表连接条件错误。在案例 1 中，GPT-4o 选择了“*School*”“*FreeMeal Count*（*Ages* 5-17）”“*Enrollment*（*Ages* 5-17）”“（*Free MealCount*（*Ages* 5-17）/ *Enrollment*（*Ages* 5-17））”作为最终的 SQL 输出属性结果，出现了与自然语言问题所要查询的属性不一致的情况。这表明在复杂的上下文中，大型语言模型可能会误解用户的查询问题需要并产生错误的检索。然而通过模式增强，ITS-SQL 可以引导模型更准确地对正确查询属性赋予更多的注意，使得问题属性能够更加精准地映射到具体的列名上。

在案例 2 中，GPT-4o 在选择查询目标数据表时发生了错误，真实值显示仅从“*bond*”表中检索便可以获得所需结果，但是 GPT-4o 错误地选择了多个表并且进行连接操作，导致了查询语句执行速度变慢并且最终返回了错误的查询结果。这个问题揭示了大语言模型在处理涉及一些特定名词数量计算的自

然语言查询时,容易对特定量产生误解,从而可能错误地判断了处理计算所需的数据表操作。相比之下,ITS-SQL 利用具有相似自然语义的提示,增强了模型对于数值关系的理解,从而实现了更准确的查询选择。

案例 3 揭示了大型语言模型在 SQL 生成中的另一个典型问题:数据表连接条件错误。具体而言,在由 GPT-4o 生成的 SQL 查询中,存在对“*comments*”表和“*posts*”表的错误关联操作——错误地将“*comments.CreationDate*”字段与“*posts.CreationDate*”字段进行关联匹配。此类连接错误可能具有多种表现形式,突显了当前大语言模型在全面理解复杂数据库关系模式方面的显著不足。针对这一问题,ITS-SQL 使用了 2 阶段优化策略:首先,通过优化“伪 SQL”的框架结构提升问题语义表征的准确性;其次借助自身迭代机制有效纠正此类错误。

3.7 困难数据库分析

为了能在难度较大的 BIRD 数据集上进行更细致的分析,本文对 ITS-SQL 方法下的 GPT-4o 在具体不同数据库上的执行正确率变化进行了统计。如表 6 所示,ITS-SQL 在 thrombosis_prediction, student_club, european_football_2 数据库上相对提升效果最高,执行正确率分别实现了 25.54 个百分点、22.78 个百分点、17.83 个百分点的提升。同时本文将每个数据库中的中等和困难难度的查询所占的百分比表示为“困难比”(用于衡量该数据库整体大致复杂度和难度),值得注意的是,查询数量较多(163)且具有最高的“困难比”(69.94%)的 thrombosis_prediction 数据库取得了最为显著的相对改进(25.54 个百分点)。这表明,对于中等到高难度的 SQL 查询,ITS-SQL 能够提供更为显著的效果改进。

Table 6 Execution Accuracy on Different Databases Within BIRD-dev Dataset

表 6 BIRD-dev 数据集的不同数据库上的执行正确率

数据库	数量	困难比/%	不同方法的执行正确率/%		变化
			GPT-4o	ITS-SQL(本文)	
california_schools	89	39.33	25.84	40.45	+14.61 个百分点
card_games	191	34.55	39.27	52.36	+13.09 个百分点
codebase_community	186	18.82	56.99	64.52	+7.53 个百分点
debit_card_specializing	64	32.81	39.06	56.25	+17.19 个百分点
european_football_2	129	49.61	51.94	69.77	+17.83 个百分点
financial	106	40.57	36.79	47.17	+10.38 个百分点
formula_1	174	32.76	35.06	47.13	+12.07 个百分点
student_club	158	36.08	53.80	76.58	+22.78 个百分点
superhero	129	37.21	71.32	84.50	+13.18 个百分点
thrombosis_prediction	163	69.94	28.83	53.37	+25.54 个百分点
toxicology	145	47.59	48.97	64.14	+15.17 个百分点

4 结 论

在本研究中,我们提出了 ITS-SQL,通过模式增强和 2 阶段生成方法,使大语言模型在 SQL 组织、表达以及自我迭代修正能力方面表现出更强的能力。在 Spider 和 BIRD 这 2 个大规模跨领域数据集上的实验结果和分析验证了我们方法的有效性和适用性。未来,我们将进一步探索上下文学习在大语言模型中的应用,以低成本构建更具实用价值的 SQL 生成策略。

作者贡献声明:张伟提出算法思路,完成实验并撰写论文;周东傲、宫永顺和尹义龙提出指导意见并修改论文。

参 考 文 献

[1] Labrinidis A, Jagadish H V. Challenges and opportunities with big data[J]. Proceedings of the VLDB Endowment, 2012, 5(12): 2032−2033

[2] Pan Qingfeng, Xu Chen. Advances in SQL execution techniques based on query compilation[J]. Journal of Computer Research and Development, 2024, 61(7): 1754−1770 (in Chinese)
(潘青峰, 徐辰. 基于查询编译的 SQL 执行技术研究进展[J]. 计算机研究与发展, 2024, 61(7): 1754−1770)

[3] Agarwal S, Mozafari B, Panda A, et al. BlinkDB: Queries with bounded errors and bounded response times on very large

data[C]//Proc of the 8th ACM European Conf on Computer Systems. New York: ACM, 2013: 29−42

[4] Deng Naihao, Chen Yulong, Zhang Yue. Recent advances in text-to-SQL: A survey of what we have and what we expect[C]//Proc of the 29th Int Conf on Computational Linguistics. Stroudsburg, PA: ACL, 2022: 2166−2187

[5] Katsogiannis-Meimarakis G, Koutrika G. A survey on deep learning approaches for text-to-SQL[J]. The VLDB Journal, 2023, 32(4): 905−936

[6] Khot T, Trivedi H, Finlayson M, et al. Decomposed prompting: A modular approach for solving complex tasks[J]. arXiv preprint, arXiv: 2210.02406, 2022

[7] Chowdhery A, Narang S, Devlin J, et al. Palm: Scaling language modeling with pathways[J]. Journal of Machine Learning Research, 2023, 24(240): 1−113

[8] Yu Tao, Zhang Rui, Yang Kai, et al. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-SQL task[C]// Proc of the 27th Conf on Empirical Methods in Natural Language Processing. Stroudsburg, PA: ACL, 2018: 3911−3921

[9] Josh A, Steven A, Sandhini A, et al. GPT-4 technical report[J]. arXiv preprint, arXiv: 2303.08774, 2024

[10] Qu Ge, Li Jinyang, Li Bowen, et al. Before generation, align it! A novel and effective strategy for mitigating hallucinations in text-to-SQL generation[J]. arXiv preprint, arXiv: 2405.15307, 2024

[11] Parciak M, Vandevoort B, Neven F, et al. Schema matching with large language models: An experimental study[J]. arXiv preprint, arXiv: 2307.07306, 2023

[12] Dong Qingxiu, Li Lei, Dai Damai, et al. A survey on in-context learning[C]//Proc of the 29th Conf on Empirical Methods in Natural Language Processing. Stroudsburg, PA : ACL, 2024: 1107−1128

[13] Liu Jiachang, Shen Dinghan, Zhang Yizhe, et al. What makes good in-context examples for GPT-3? [J]. arXiv preprint, arXiv: 2101.06804, 2021

[14] Wang Liang, Yang Nan, Wei Furu. Learning to retrieve in-context examples for large language models[C]//Proc of the 18th Conf of the European Chapter of the Association for Computational Linguistics. Stroudsburg, PA : ACL, 2024: 1752−1767

[15] Nguyen T, Wong E. In-context example selection with influences[J]. arXiv preprint, arXiv: 2302.11042, 2023

[16] Radford A, Narasimhan K, Salimans T, et al. Improving language understanding by generative pre-training[EB/OL]. 2018[2025-03-02]. https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf

[17] Anil R, Borgeaud S, Alayrac J, et al. Gemini: A family of highly capable multimodal models[J]. arXiv preprint, arXiv: 2312.11805, 2025

[18] Touvron H, Lavril T, Izacard G, et al. LLaMA: Open and efficient foundation language models[J]. arXiv preprint, arXiv: 2302.13971, 2023

[19] Bi Xiao, Chen Deli, Chen Guanting, et al. DeepSeek LLM: Scaling open-source language models with long termism[J]. arXiv preprint, arXiv: 2401.02954, 2024

[20] Radford A, Wu J, Child R, et al. Language models are unsupervised multitask learners[EB/OL]. 2019[2024-04-23]. https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf

[21] Pourreza M, Rafiei D. DIN-SQL: Decomposed in-context learning of text-to-SQL with self-correction[C]// Proc of the 37th Int Conf on Neural Information Processing Systems. New York: ACM, 2023: 36339−36348

[22] Gao Dawei, Wang Haibin, Li Yaliang, et al. Text-to-SQL empowered by large language models: A benchmark evaluation[J]. Proceedings of the VLDB Endowment, 2024, 17(5): 1132−1145

[23] Dong Xuemei, Zhang Chao, Ge Yuhang, et al. C3: Zero-shot text-to-SQL with ChatGPT[J]. arXiv preprint, arXiv: 2307.07306, 2023

[24] Wang Bing, Ren Changyu, Yang Jian, et al. MAC-SQL: A multi-agent collaborative framework for text-to-SQL[C]//Proc of the 32nd Int Conf on Computational Linguistics. Stroudsburg, PA : ACL, 2025: 540−557

[25] Maamari K, Abubaker F, Jaroslawicz D, et al. The death of schema linking? Text-to-SQL in the age of well-reasoned language models[C]//Proc of the 38th Advances in Neural Information Processing Systems. New York: ACM, 2024: 1−10

[26] Hasan A C, Özgür U. E-SQL: Direct schema linking via question enrichment in text-to-SQL[J]. arXiv preprint, arXiv: 2409.16751, 2025

[27] Li Haoyang, Zhang Jing, Liu Hanbing, et al. Codes: Towards building open-source language models for text-to-sSQL[J]. Proceedings of the ACM on Management of Data, 2024, 2(3): 1−28

[28] Peter F B, Vincent P J D, Peter V D, et al. Class-based n-gram models of natural language[J]. Computational Linguistics, 1992, 18(4): 467−480

[29] Li Zehan, Zhang Xin, Zhang Yanzhao, et al. Towards general text embeddings with multi-stage contrastive learning[J]. arXiv preprint, arXiv: 2308.03281, 2023

[30] Alomari Z, Halimi O E, Sivaprasad K, et al. Comparative studies of six programming languages[J]. arXiv preprint, arXiv: 1504.00693, 2015

[31] Li Jinyang, Hui Binyuan, Qu Ge, et al. Can LLM already serve as a database interface? A big bench for large-scale database grounded text-to-SQLs[C]//Proc of the 37th Int Conf on Neural Information Processing Systems. New York: ACM, 2024: 42330−42357



Zhang Wei, born in 1999. Master. His main research interests include natural language processing and data mining.

张　伟, 1999 年生。硕士。主要研究方向为自然语言处理、数据挖掘。



Zhou Dong'ao, born in 1990. PhD, assistant researcher. His main research interests include artificial intelligence and signal detection.

周东傲, 1990 年生。博士，助理研究员。主要研究方向为人工智能、信号检测。



Gong Yongshun, born in 1990. PhD, professor. His main research interests include data mining and machine learning.

宫永顺, 1990 年生。博士, 教授。主要研究方向为数据挖掘、机器学习。



Yin Yilong, born in 1972. PhD, professor. His main research interests include data mining and machine learning.

尹义龙, 1972 年生。博士, 教授。主要研究方向为数据挖掘、机器学习。

《计算机研究与发展》投稿指南

征稿范围

计算机体系结构; 计算机网络与通信; 网络与信息安全; 人工智能; 软件技术; 并行与分布式计算; 图形图像、信息检索等应用技术; 其他计算机相关领域。

征稿类型

1)学术论文: 有创新学术见解的研究成果的完整论述, 对该学术领域的发展有积极意义, 或者支持我国学术生态和产业生态的重要成果。

2)综述: 对新兴的、活跃的学术研究领域或技术开发领域的评述。

3)快报短文: 对目前国内外计算机领域新思想、新观点、新技术的解读文章。

征稿要求

1)本刊只接收中文稿, 不受理英文稿。学术论文建议不超过 15 页, 综述不超过 20 页, 短文不超过 4 页。

2)要求来稿未在正式出版物上发表过, 不存在一稿多投问题。

3)作者在投稿时, 需同时向编辑部提交所有作者手写签字确认的“投稿声明”扫描版, 如不提供不予受理.对于违反投稿声明相关条款的来稿, 本刊将视情节做严肃处理, 后果自负。

4)本刊双盲评审, 作者投稿时上传系统的电子版中需要隐去作者、单位、基金、作者简介等信息, 初审通过后给编辑部邮寄的纸质版本需要包含前述署名信息。改后复审的稿件在提交修改版本以及修改说明时, 同样需符合双盲评审要求。

5)稿件评审通过后, 作者需提交单位审核盖章的“不涉密证明”和所有作者签字确认的“著作权转让声明”。

6)投稿时间在 6 个月以上的, 若没有收到编辑部发出的审理结果通知, 在书面通知编辑部并收到编辑部确认回复之后, 作者方可自行处理此稿件。

7)无论何种原因撤回投稿, 需以书面形式通知编辑部并确保所有作者知情。