李洪海, 郭利荣, 李金漳. 基于 LLM 的私有数据库的 NL2SQL 研究[J]. 智能计算机与应用,2025,15(6):171-177. DOI:10.20169/j.issn.2095-2163.250626

基于 LLM 的私有数据库的 NL2SQL 研究

李洪海, 郭利荣, 李金漳
(中数通信息有限公司, 广州 510630)

摘 要: 从 2020 年 6 月开始,GPT-3 的发布,标志着人工智能领域发展进入新阶段。在代码生成领域基于开源的通用模型通过 fine-tune、Lora 等方法微调通用模型生成了诸如:StarCode 系列,CodeFuse 系列模型。在 NL2SQL 领域,人们还在使用较为传统的编码器-上下文增强层-输出层架构,由于模型尺寸和架构,传统 NL2SQL 存在着语义表征能力弱、生成 SQL 范式受限、不具有强大的泛化能力的不足,很难在工业化应用上有所斩获,因此本文研发团队提出了面向垂直领域基于 LLM 生成的 NL2SQL 任务范式,通过通用大模型+代码大模型+筛选器模型的理念构建,在标准中文数据集 Cspider 上汇报了 500 条混合 SQL 的 0.187 7 的精准匹配率,0.354 6 结构准确率。300 条单表 SQL 报告了 0.67 的结构准确率,0.27 的精准匹配率。

关键词: LLM; NL2SQL; Cspider; 私有数据库
中图分类号: TP311 **文献标志码**: A **文章编号**: 2095-2163(2025)06-0171-07

NL2SQL research on private databases by LLMs

LI Honghai, GUO Lirong, LI Jinzhang
(China Datacom Co., Ltd., Guangzhou 510630, China)

Abstract: Since June 2020, the release of GPT-3 signifies that the development of the field of artificial intelligence has entered a new stage. In the field of code generation, based on the open source general model, the general model is fine-tuned through fine-tune, Lora and other methods, and the general model is generated, such as: StarCode series, CodeFuse series models. In the field of NL2SQL, people are still using the more traditional encoder-context enhancement layer-output layer architecture, due to the model size and architecture, the traditional NL2SQL has weak semantic representation ability, limited generation of SQL paradigm, and also does not have strong generalization ability, meanwhile is difficult to achieve technological progress in industrial applications. Therefore, the research team proposes an NL2SQL task paradigm based on LLM generation for vertical fields, through the concept of general large model + code large model + filter model. The exact match rate of 0.187 7 and the structural accuracy of 0.354 6 for 500 mixed SQL statements are reported on the standard multi-round dataset Cspider. A single table of 300 SQL statements reported a structure accuracy of 0.67 and an exact match rate of 0.27.

Key words: LLM; NL2SQL; Cspider; private databases

0 引 言

NL2SQL 也称 Text2SQL,该工作的具体释意为给定数据库结构(Context-Schema)的情况下,人为设计的模型能够将用户态得到自然语言转化为可执行的 SQL 语义分析(Semantic parsing)的子任务。在当今世界,数据飞速增加,在金融、电子商务、医疗等领域,随着数字化、信息化的发展,大量数据都被存储在关系型数据库中,因此高效利用数据分析结论则成为发挥数据价值的重要途径,而自然语言与数据库语言 SQL 之间存在着一些区别,这使得很多业务人员无法快速地使用到数据库中的数据,因此 NL2SQL 任务应运而生[1-4]。接下来将简要介绍 NL2SQL 的类型划分和相关工作的进展,并就技术问题与学术热点展开研究论述。

作者简介: 李洪海(1974—),男,硕士,高级工程师,主要研究方向:大数据,深度学习技术,大规模预训练语言模型(PLM); 李金漳(2002—),男,学士,主要研究方向:大规模预训练语言模型(PLM)。
通信作者: 郭利荣(1977—),男,硕士,助理工程师,主要研究方向:大数据,深度学习技术,大规模预训练语言模型(PLM)。Email:glr@cndatacom.com。

收稿日期: 2023-11-23

1 相关工作

根据用户的提问和功能不同,可以从不同的维度来对这些任务进行分类:

(1)用户提问中涉及的表的数目。如果多于一张表,那么表之间的主外键链接构建。

(2)用户提问中是否包含聚合函数、关联、组合的操作。

(3)是否需要进行用户间的交互,当前用户提问是否与前一轮的对话相关。

接下来将简要介绍 NL2SQL 领域在近些年来的技术发展和数据集,并基于出现的问题进行分析,进一步给出可能的解决方法。

目前,在这些数据集和其他传统英文领域的数据集上也涌现了许多经典的工作成果,比如早期的研究很多都基于规则的方法。由于 SQL 查询语言本身是一种具有强范式的编程语言,具有严格的语法结构,同时根据 SQL 执行的复杂程度可以拆分为各个子句进行标准化处理,但是这样的工作过于机械、并且脱离了数据库结构,容易出现缺漏匹配、近似枚举的问题,因此该方法目前已然成为业界的辅助手段。而在事实上,NL2SQL 任务也是一种翻译任务,因此很多时候,业界都在推介翻译任务上的模型,并对模型架构及其缺点、即基于 RNN 或 LSTM 的 Seq2Seq 架构语义表征能力弱,无法并行训练,复杂问题解决能力弱以及基于 BERT 的混合嵌入架构语义表征能力弱,难以适应工业应用等给予了一定的分析研究,同时也对 NL2SQL 进行了改进。针对上述情况,基于序列到序列(Seq2Seq)的深度学习架构现已逐渐成为了业界关注重点。

在基于 Seq2Seq 的 NL2SQL 任务中基于循环神经网络(RNN)和长短时记忆网络(LSTM)的工作,开展得最早。其中,杨梦琴[5]在语义驱动的数据查询与智能可视化研究中基于 Word2Vec 词嵌入技术,同时结合 SQL 语法依赖关系图、Sequence-to-set 和表列名注意力机制,采用 LSTM 神经网络分别构建了 SELECT 子句与 WHERE 子句的预测模型。万文军[6]基于实体关系的思想,将嵌套查询 SQL 语法结构解析问题转为关系抽取问题,构建模型通过输入查询语句和数据表的特征表示,采用结合注意力机制的 Bi-LSTM 网络捕捉双向关键语义信息,从而实现 SQL 语句准确率的提升。孙红等学者[7]通过对传统的 SQLNet 进行改进,为了增强特征提取能力融入了预训练模型,同时对传统分类模型和条件值模型进行了优化改进,并在分类模型中增加了 LSTM 模型捕捉特征,采用正则表达式等手段在特殊条件语句上进行预处理。然而在前述研究工作中不难发现,RNN 和 LSTM 在长序列建模能力上仍有待加强,特别是 RNN,LSTM 容易出现梯度消失或者梯度爆炸的问题,表现出无法进行并行训练、训练速度慢、泛化能力差、文本表征能力差的特点。目前,随着 Transformer 架构在翻译任务中取得了显著成果,大规模 NLP 预训练模型 BERT 也在各领域获得了广泛使用。

近几年,基于 BERT(Bidirectional Encoder Representations from Transformer)及其改进的科研成果陆续推出,例如张中正等学者[8-9]基于 BERT-Base 模型并将实验模型分为 5 个组件:聚合函数预测、条件间关系预测、条件运算符预测、条件值预测和条件选择列预测,在实验中对于条件列预测,将该组件为解为 3 个部分,成功解决查询语句可能存在多个条件的情况。同样,张琰[10]基于 SQLova 模型、X-SQL 模型将 NL2SQL 任务解耦为多个多分类子任务的设计思路,基于 BERT 模型,构建了一个中文 NL2SQL 模型,将用户问题和表结构作为输入,SQL 作为模型输出,在中文 NL2SQL 数据集上取得了 87%的准确率,接近了当前 state-of-the-art 模型 X-SQL 在 WikiSQL 上的表现。在上述的模型中主要都是采用了 BERT 作为用户态输入的文本特征提取器,同时融合数据库表结构特征作为嵌入,将用户 query 嵌入和数据库结构嵌入相结合进行训练得到最后的 SQL 表达,见表 1。

表 1 架构及其缺点

Table 1 Architecture and the corresponding disadvantages

架构	缺点
基于 RNN 的 Seq2Seq	语义表征能力弱,无法并行
基于 BERT 的嵌入	语义表征能力弱,维度低

最近,鉴于 decoder-only 架构在文本生成领域和代码生成领域的出众表现,同时辅以大模型强大的表征空间,为语义理解和语义推理赋予了强大的能力,因此基于 LLM 的 NL2SQL 任务,借助语言大模型的表达能力、推理能力,通过小模型筛选用户问题相关表,进而利用大模型自动推理、代码生成模型以及结合用户提问和数据库结构信息进行推理生成,由此得到最后的 SQL 表达已然成为了一种可能。

在 NL2SQL 领域基于中文的数据集较少,同时由于中文的语义表达丰富,具有大量的近义词等模

糊表达，相较于英文与 SQL 语言之间的紧密联系，导致中文的 NL2SQL 任务面临着巨大的挑战，而在过去的数年间中也涌现了许多优秀的中文数据集，具体见表 2。

表 2　中文 NL2SQL 数据集

Table 2　Chinese NL2SQL dataset

数据集	年份	Query & SQL	数据库	领域	表格数	轮数
NL2SQL	2019	49 974	5 291	多	5 291	单
Cspider	2019	9 691	166	多	880	单
DuSQL	2020	23 797	200	多	880	单
CHASE	2021	17 940	280	多	1 280	多

2　本文工作

对于 NL2SQL 任务在 LLM 应用下的新范式，所涉及的任务规模要 NL2SQL 更为宏大，本次研究旨在实现数据库结构的自动同步，并通过用户 Query 自动识别相关表和表的结构，结合用户提问和表结构生成可用的 SQL，因此可以将 NL2SQL 拆解为 2 个子任务：

(1)如何得到用户 Query 相关的表。

(2)代码生成模型对于用户 Query 和表结构信息整合生成的 SQL 能力。

因此接下来将围绕这 2 个核心问题展开论述。

2.1　基于小模型+大模型的相关表搜索

为了能够通过用户提问实现对用户数据库的直接映射，研究提出了基于向量数据库的关系数据库结构同步方法。基于中文的 Word2Vec－base 模型[11] 与向量数据库 Chromadb，Word2Vec 和 Chromadb 的版本选择见表 3。

表 3　Word2Vec 基础模型与基础向量数据库

Table 3　Base model of Word2Vec and Chromadb versions

模型名称	基础模型	向量数据库
SBERT-Bert-Base	BERT-Base-Chinese	Chromadb 0.4.14

将关系型数据库中对表的描述注释作为索引，通过 Word2Vec 的方法将表描述向量化并将其输出，也就是 Embedding 作为向量索引存储到向量数据库的索引栏中，将表中的字段名称及其注释、数据类型作为元数据 MetaData 进行存储，过程中的操作步骤如图 1 所示。

当用户第一次使用模型时，将同步关系型数据库的表描述和表结构等信息，为用户的提问做好准备。为了能够高速地从几百条信息甚至上千张表中筛选出合适的表及表结构，选择使用 Chromadb 向量数据库来研发实现。通过将用户问题 Query 进行向量化后，快速计算用户 Query 和向量数据库中预存的表及其结构信息，可利用余弦相似度公式得到相似分数进行排序：

$$\cos(\theta)=\frac{\sum_{i=1}^{n}(Q_i\times K_i)}{\sqrt{\sum_{i=1}^{n}Q_i^2}\times\sqrt{\sum_{i=1}^{n}K_i^2}} \tag{1}$$

其中，Q 表示 Query 的 Embeddeding，K 表示向量数据库中的某个索引 Embeddeding。

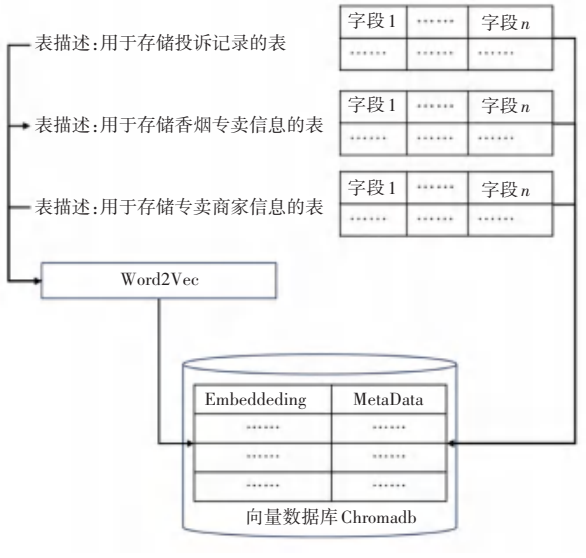


图 1　关系数据库同步到向量数据库

Fig. 1　Relational databases synchronizing to Chromadb

基于此，筛选出最有可能的 10 条并提供表的描述，将这些信息和用户 Query 通过 prompt 与 few-shot 的形式构建出提示模板传送给大模型进行推理，这对大模型的长距离建模能力和推理提出了一定的要求。因此在选择大模型时将着重考虑大模型的推理和长文件表达能力。各模型的 C－Eval、GSM8K、显存的占用情况见表 4。

表 4 各模型的 C-Eval,GSM8K,显存的占用情况

Table 4 C-Eval,GSM8K and GPU-memory of each model

模型名称	推理能力(C-Eval)	逻辑推理(GSM8K)	显存/G
ChatGLM2-6B	51.7	32.4	12.06
Qwen-14B	71.7	61.3	28.74
Llama2-13B	36.2	16.7	14.50

结合显存和推理能力,研究选择了通义千问-14 B 作为后台的文本推理模型,该过程描述如图 2 所示。

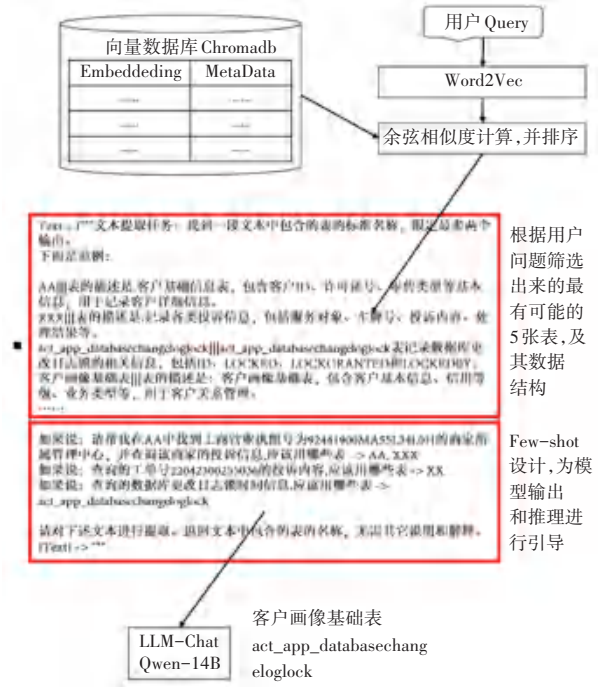


图 2 大+小模型根据用户 Query 筛选获取数据库结构信息

Fig. 2 Large + small model selecting db-schema by Query

2.2 代码生成模型的生成结果

在这个部分,使用经过预训练的 CodeFuse-34b 模型[9,12]。由于该模型是基于 CodeLlama-34b 模型经过 fine-tune 后得来,在代码生成评测中 CodeFuse-34b-4bits 模型的 *HumanEval* 、也就是 *Pass*@1 指标接近 SOTA, *Pass*@1 指标是 *Pass*@*k* 指标的特殊情况。*Pass*@*k* 是由 OpenAI 在 *HumanEval* 论文中提出,*Pass*@1 的目的是为了估计每一次代码生成时能够通过运行/编译的概率,推得的数学公式为:

$$Pass@1 = E_{problems}\left[1 - \frac{\binom{n-c}{1}}{\binom{n}{1}}\right] \quad (2)$$

Int4 量化后的 CodeFuse-CodeLlama-Python-34B 具有相近的代码生成能力下仍然具有接近 SOTA 的表现,具体指标见表 5。

表 5 各个模型的 *HumanEval* 与显存占用

Table 5 *HumanEval* and GPU-memory of each model

模型名称	*HumanEval*/%	显存占用/GB
CoFuse-34B(SOTA)	74.4	69.31
CodeFuse-34B-4 bits	73.8	22.19
WizardCoder-34B-V1	73.2	
GPT-4(zero-shot)	67.0	

因此在关键指标相差 0.6%的情况下,选择了显存占用仅为 1/3 的 4 bits 版本。同样地、基于样本构建器,将 2.1 节中构建的相关表搜索模型的数据结果与用户 Query 相结合推送到 LLM-Code,由此推理生成最后的查询 SQL,如图 3 所示。

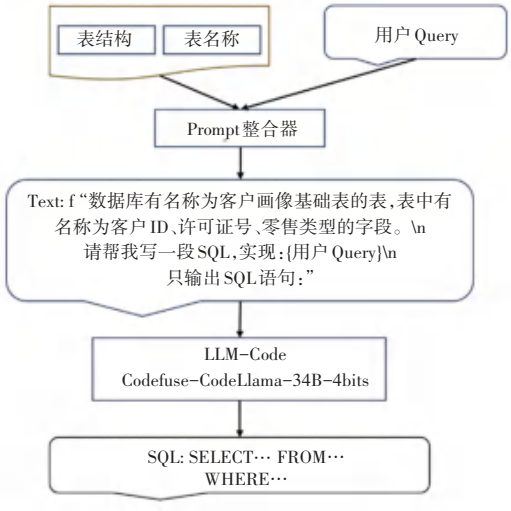


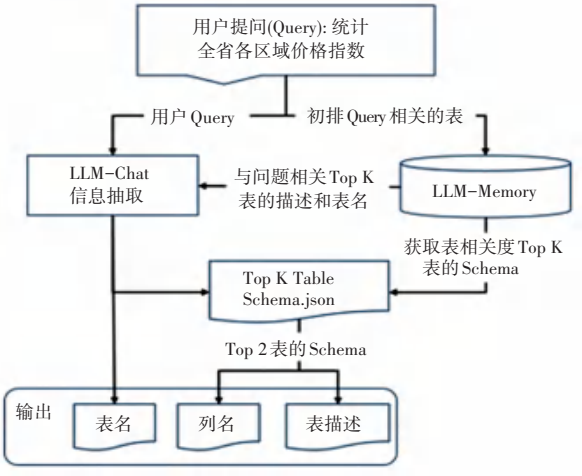
图 3 通过 Prompt 整合器传入 CodeFuse 模型生成 SQL

Fig. 3 Prompt passing in CodeFuse to generate SQL

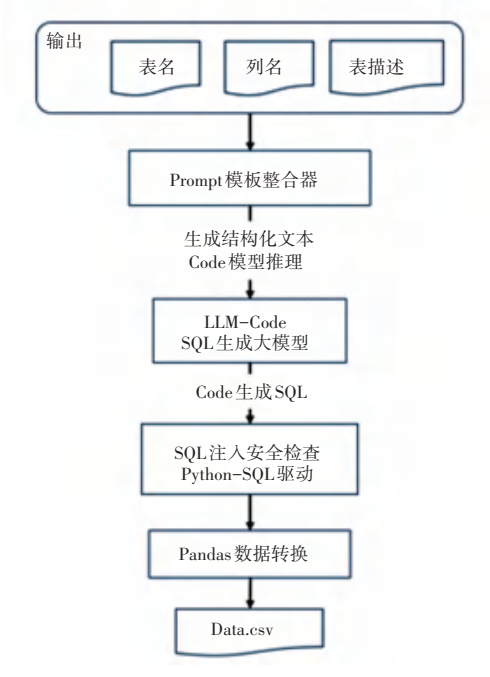
2.3 工作综述

基于 2.1 和 2.2 节的工作通过小+大模型的结合对用户的提问涉及到的表进行推理并整合用户提问传入到 CodeFuse 处生成 SQL 语句。比如当用户输入提问:东莞卷烟名录中卷烟品牌有哪些。首先模型根据用户问题在向量数据库中筛选出最相关的 10 张表,并将这 10 张表的表名及其描述和用户问题装载在预先设定的 Prompt 模板装载器包装为一个结构性文本,交给后台部署的 LLM-Chat(Qwen-14B)模型进行代理的信息抽取任务,通过 Langchain 的输出结构化工具和代理任务中的提示规范化输出,模型推理得到最有可能的 2 张表,将这 2 张表的表名和数据结构与用户提问使用 Prompt 包装器进

行结构化处理，提交到 LLM－Code（CodeFuse－CodeLlama－34B－4 bits）模型进行推理得到生成 SQL 语句。模型数据流转和模型架构如图 4 所示。



（a）数据流 1



（b）数据流 2

图 4　实际使用时模型的数据流转

Fig. 4　Data flow of the model during using

3　实验工作

3.1　实验分析

基于第 2 节对本文工作的基本分析可知，本文的研究工作是一种两阶段任务，因此最终实验误差也主要来源于 2 个部分：

（1）是 LLM-Chat 模型在执行信息抽取的代理任务时，是否能够正确地返回涉及到的表与表的结构。

（2）是 LLM_Code 模型在进行长文本建模时是否够准确地进行建模生成 SQL。

但是由于数据集的原因尚未出现带有数据库表注释的公开 NL2SQL 数据集，但是在本次研究生产环境中的数据集具有敏感性，因此本次的实验只基于 Cspider 进行第 2 个部分的实验展示，在实际使用发现部分 1 也会带来大约 10%～15%的错误比例，因此在后续工作中这样的两阶段任务必须做进一步的优化。首先将简要介绍本文实验中涉及的数据集（见表 6）与所使用的机器环境：操作系统为 Ubuntu 20.04.6 LTS（GNU/Linux 5.15.0-82- generic x86_64），Python 解释器 Python 3.9.6，核心库版本为 Torch 2.0.1，Langchain 0.0.316，Chromadb 0.4.14，Pandas 2.0.3，GPU 为 NVIDIA A800 80 GB PCIe，CPU 与内存为 Intel（R）Xeon（R）Gold 6330 CPU/512 G。

表 6　实验的数据集

Table 6　Dataset of the experiment

数据集	Query&SQL	语言	数据库	表格数	领域
Cspider	9 691	中文	166	880	多

3.2　实验结果

为了执行效率，本文在 Cspider 数据集上选择了 500 条 Query-SQL 对作为测试数据计算得分，同时观察 100 条、200 条、300 条、400 条、500 条时的数据表现保障测试具有稳定性和代表性。接下来将介绍测试中的评价指标。与传统的标准不同基于本文的方法和任务，改造了常用的评价指标，将逻辑准确率（运行通过率）转化为结构准确率，同时保留准确匹配率作为匹配指标。由于在实验中发现模型有时出现自动提供过滤条件的可能，但是出现过滤条件的语句是能够通过运行的，因此本文提出了结构准确率 ACC_{struct} 被定义为：

$$ACC_{\text{struct}} = \frac{k}{N_{\text{all}}} \tag{3}$$

其中，k 表示生成的 SQL 与标准 SQL 只存在字段、表名差异的数目，N_{all} 表示所有进行测试的 SQL 数目。

精准匹配率的定义为模型生成的 SQL 与真实的 SQL 仅允许存在空格或者转义字符上的差别，如果存在差别视为失败，对此可表示为：

$$ACC_{\text{qm}} = \frac{k}{N_{\text{all}}} \tag{4}$$

其中，k 表示生成的 SQL 与标准 SQL 精准匹配的数目，N_{all} 表示所有进行测试的 SQL 数目模型。在试验中，在 Cspider 数据集上的得分见表 7。

表 7 在 Cspider 上随机 500 条的各类得分

Table 7 ACC_{struct} and ACC_{qm} in Cspider

指标	500 条混合 SQL	300 条单表 SQL	100 条多表 SQL
结构准确率	0.354 6	0.67	0.15
精确匹配率	0.187 7	0.27	0.02

以下将对不同 SQL 条数下各指标的稳定性进行分析并绘制出图表,如图 5、图 6 所示。可以发现本文的实验结构具有稳定性。

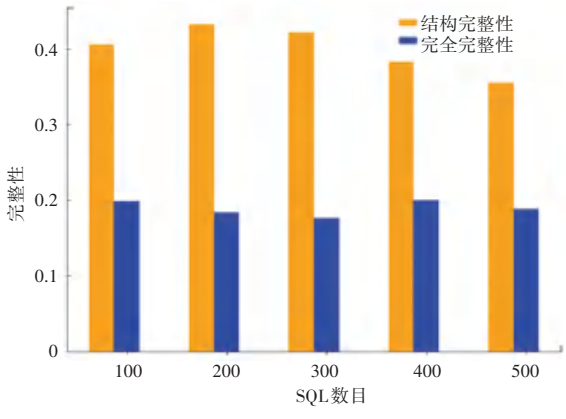


图 5 500 条混合 SQL 在不同 SQL 条数下的稳定性分析图表

Fig. 5 500 mixed stability analysis charts

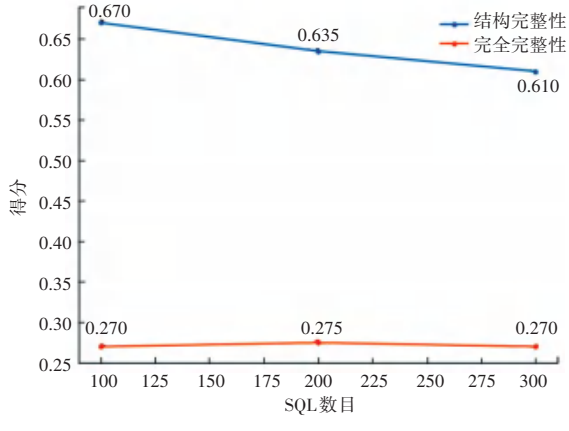


图 6 每百条单表 SQL 的精确匹配率/结构准确率变化

Fig. 6 ACC_{struct} and ACC_{qm} of SQL

4 结束语

在第 3 节中,本文的模型取得了较好的结构准确率,但是在精确匹配率上存在着一些问题。在此,对实验进行消融分析至关重要。在此对于第 3 节实验中表现欠佳的原因进行分析。可知:Cspider 上的数据库缺乏必要的字段注释,然而在实际业务数据库中,对数据库中表的初始化,并且保留表的注释等工作是 DBA 进行数据库建设时不可避免的重要工作,因此模型在业务数据集上得到了 77%的单表查询准确率,在 Cspider 上仅有一半。

基于 LLM 的 NL2SQL 方法,虽然目前架构尚未成熟,与 Cspider 上的得分为 60.6 的 SOTA 模型还存在巨大的差距,但是相比较传统的 NL2SQL 方法,本文摒弃了 Seq2Seq 的框架,借助大模型的语义理解能力和 Decoder only 架构在生成式模型方面的天然优势,将 NL2SQL 任务设计为一系列的大模型代理任务,因此是具有一定的可用性和潜力发展空间的。总结本文的研究工作和展望 NL2SQL 领域的未来发展,引入 LLM 能让以下问题得到缓解或解决:

(1)本文的模型能够通过自动同步关系型数据库,将分类模型转换为文本推理问题具有可迁移、无需重复训练的特点。

(2)本文的模型基于 LLM,具有语义上的天然丰态,能够较好地对用户进行意图识别和推理,能在一定程度上避免通过率虚高但是答非所问的情况。

但是本文提出模型也存在一些问题和不足:

(1)双 LLM 模型。对系统资源要求较大,并且由于任务设计时的流无法支持并行。

(2)过度依赖业务数据库上的表信息标注。当出现无人工标注的数据库时(指的是表的注释、字段的注释),模型的理解会大幅减弱,例如 Cspider 上的数据,相比业务数据集 80%的通过率出现较大问题。

接下来,将更新架构基于 LLM 的推理能力实现语义的丰态理解,得到潜在的用户 query 与数据库中 Schema 的关系,并给出更为准确的用户 query 和相关表的推导。

更新 SQL 生成架构,不适用庞大的代码生成大模型作为 SQL 端的生成,考虑使用 7B 级别的模型使用大量的 SQL 数据作为微调,同时避免 overfit,打造一个专属于 SQL 的 decoder only 架构的 SQL 语句生成器。

参考文献

[1] 刘译璟,徐林杰,代其锋. 基于自然语言处理和深度学习的 NL2SQL 技术及其在 BI 增强分析中的应用[J]. 中国信息化,2019(11):62-67.

[2] WU Qiankun, PENG Dunlu, MTL-BERT: A multi-task learning modelutilizing bert for Chinese text [J]. Journal of Chinese Computer Systems, 2021,42(2):291-296.

[3] BAIK C, JAGADISH H V, LI Y. Bridging thesemanticgapwith SQL query logsinnatural languageinter faces to databases[C]//

Proceedings of the IEEE 35th Internatinal Conference on Data Engineering(ICDE). Piscataway, NJ:IEEE,2019:374-3895.

[4] WANG Bailin, SHIN R, LIU Xiaodong, et al. RAT-SQL: Relation-aware schemaen coding and linking for text-to-SQL parsers[C]//Proceedings of thes 50th Annual Meeting of the Association for Computational Linguistics. ACL, 2020: 7567-7579.

[5] 杨梦琴. 语义驱动的数据查询与智能可视化研究[D]. 重庆:重庆大学,2018.

[6] 万文军. 基于实体关系的 NL2SQL 语法结构构建[D]. 烟台:山东工商学院,2020.

[7] 孙红,黄陔严. 融合 LSTM 的自然语言转结构化查询语句算法的研究与设计[J]. 小型微型计算机系统,2023,44(1):63-67.

[8] 张中正,王蓓,赵建保,等. 基于 NL2SQL 实现电力数据智能交互[J]. 电网技术,2022,46(7): 2564-2571.

[9] Modelscope. CodeFuse-CodeLlama-34B[EB/OL]. (2024-04-18). https://modelscope.cn/models/codefuse-ai/CodeFuse-CodeLlama-34B/summary.

[10] 张琰. 一种基于 BERT 的中文 NL2SQL 模型[D]. 济南:山东大学,2020.

[11] XU M. Text2vec: Text to vector toolkit (Version 1.1.2)[Computersoftware][EB/OL]. (2023-09-01). https://github.com/shibing624/text2ve.

[12] Modelscope. CodeFuse-StarCode-13B[EB/OL]. (2023-10-25). https://modelscope.cn/models/codefuse-ai/CodeFuse-13B/summary.