doi:10.11887/j.cn.202402020 http://journal.nudt.edu.cn

辅助任务增强的中文跨域 NL2SQL 算法

胡亚红[1],刘亚冬[2],朱正东[3*],刘鹏杰[3]

(1. 浙江工业大学 计算机科学与技术学院,浙江 杭州 310023;2. 西安交通大学 软件学院,陕西 西安 710049;
3. 西安交通大学 计算机科学与技术学院,陕西 西安 710049)

摘 要:自然语言到结构化查询语言(natural language to structured query language,NL2SQL)任务旨在将自然语言询问转化为数据库可执行的结构化查询语言(structured query language,SQL)语句。本文提出了一种辅助任务增强的中文跨域 NL2SQL 算法,其核心思想是通过在解码阶段添加辅助任务以结合原始模型来进行多任务训练,提升模型的准确率。辅助任务的设计是通过将数据库模式建模成图,预测自然语言询问与数据库模式图中的节点的依赖关系,显式地建模自然语言询问和数据库模式之间的依赖关系。针对特定的自然语言询问,通过辅助任务的提升,模型能够更好地识别数据库模式中哪些表/列对预测目标 SQL 更有效。在中文 NL2SQL 数据集 DuSQL 上的实验结果表明,添加辅助任务后的算法相对于原始模型取得了更好的效果,能够更好地处理跨域 NL2SQL 任务。

关键词:人工智能;深度学习;自然语言处理;语义解析

中图分类号:TP391 **文献标志码**:A **开放科学(资源服务)标识码(OSID)**:

文章编号:1001-2486(2024)02-197-08

Chinese cross-domain NL2SQL algorithm enhanced by auxiliary task

HU Yahong[1], *LIU Yadong*[2], *ZHU Zhengdong*[3*], *LIU Pengjie*[3]

(1. College of Computer Science and Technology, Zhejiang University of Technology, Hangzhou 310023, China;
2. School of Software Engineering, Xi'an Jiaotong University, Xi'an 710049, China;
3. School of Computer Science and Technology, Xi'an Jiaotong University, Xi'an 710049, China)

Abstract: NL2SQL(natural language to structured query language) task aims to translate natural language queries into SQL(structured query language) executable by the database. A Chinese cross-domain NL2SQL algorithm enhanced by auxiliary tasks was proposed. Core idea was to perform multi-task training and improve the accuracy of the model by adding auxiliary tasks in the decoder and combining the prototype model. Auxiliary task was designed by modeling the database schema into a graph, predicting the dependency relations between the natural language queries and the nodes in the database schema graph, and explicitly modeling the dependency relations between the natural language query and the database schema. Through the improvement of auxiliary tasks, the model can better identify which tables/columns in the database schema are more effective for predicting the target SQL for specific natural language queries. Experimental results on the Chinese NL2SQL dataset DuSQL show that the algorithm after adding auxiliary tasks has achieved better results than the prototype model, and can better handle cross-domain NL2SQL task.

Keywords: artificial intelligence; deep learning; natural language processing; semantic parsing

结构化查询语言(structured query language,SQL)已成为标准的数据库查询语言,但其编写难度阻碍了非专业用户的使用。自然语言到结构化查询语言(natural language to structured query language,NL2SQL)任务将自然语言问题生成结构化查询语言,从而给非专业用户提供了一种与数据库进行交互的方式[1]。

从20世纪80年代起,NL2SQL 任务就在数据库领域以及自然语言处理领域中得到重视。随着深度学习在自然语言处理领域获得突破,将深度学习方法应用到 NL2SQL 任务的研究取得了不错的效果[2-6]。2017年,Salesforce 公布大型 NL2SQL 数据集 WikiSQL[7],此数据集包含80 654个自然语言询问和其对应的 SQL,以及

收稿日期:2022-01-18

基金项目:国家重点研发计划资助项目(2018YFB0204003,2018YFB0204004)

第一作者:胡亚红(1971—),女,陕西西安人,副教授,博士,硕士生导师,E-mail: huyahong@zjut.edu.cn

***通信作者**:朱正东(1963—),男,江苏宜兴人,研究员,博士,博士生导师,E-mail: zdzhu@xjtu.edu.cn

24 241 张数据库表。之后出现了许多研究以提高 WikiSQL 数据集的准确率。WikiSQL 中需要预测的 SQL 语句都较为简单,目前采用模式感知去噪(schema aware denoising,SeaD)的方法来训练 Seq-to-Seq 模型,已经取得了高达 93% 的测试集精度。2018 年,耶鲁大学发布名为 Spider[8] 的新 NL2SQL 数据集,Spider 相对于 WikiSQL 难度更高,它由 10 181 个问题和 5 693 个独特的复杂 SQL 组成,涉及 200 个数据库,多个表涵盖 138 个不同的领域。在 Spider 中,不同的复杂 SQL 和数据库出现在训练和测试集中。所以,如果想要更好地处理 Spider 基准,模型不仅需要能够很好地泛化到新的 SQL,还必须能很好地泛化到新的数据库模式,称之为跨域 NL2SQL 问题,这是目前大家更为关注的方向[9-14]。

NL2SQL 的研究大多聚焦在英文数据集。在国内,2019 年,追一科技举办首届中文 NL2SQL 挑战赛,并提供了中文 NL2SQL 数据集 TableQA[15],中文 NL2SQL 开始逐渐走进大家的视野。比赛获得了广泛的关注,其中来自国防科技大学的团队提出 M-SQL[16] 以 92% 的准确率夺得冠军。TableQA 数据集中的问题相对简单,问题类型为基于单条件和多条件查询匹配的答案检索。后续百度提出了难度更高、实用性更强的 DuSQL[17] 数据集。DuSQL 是第一个用于跨域 NL2SQL 解析的大规模实用中文数据集,可以更好地覆盖实际应用中常见的问题类型,在实际应用中发挥更大的价值,其基于实际应用分析构建了多领域、多表、包含复杂问题的数据集,包含了 200 个数据库、813 张数据库表以及 23 797 个问题和 SQL 对。DuSQL 数据集的提出给中文 NL2SQL 的发展做出很大贡献,很多英文数据集上的成果在中文数据集上做一定的适配也可以取得不错的效果,但是中文数据集的特点导致了在编码和解码阶段不能生搬硬套英文数据集的方法,需要针对中文数据集做一些方法上的优化。

许多 NL2SQL 算法专注于特定领域,即在相同的数据库模式下进行模型的训练和推理。但是由于 NL2SQL 任务的数据标注成本很高,数据库开发人员很难为每个特定领域从零开始构建模型。因此,如何提高模型的泛化能力,使模型能够感知其训练期间从未见过的数据库模式就成为一个跨域 NL2SQL 任务的关键问题。解决该问题的挑战在于如何更好地建立自然语言询问和数据库模式之间的映射。例如图 1 所示,自然语言询问中的“做了多少次封面人物”意在获取数据库表——“期刊封面人物”的列——“次数”的数据。模型需要准确地建立自然语言询问与数据库模式之间的映射才可以得到正确的结果。

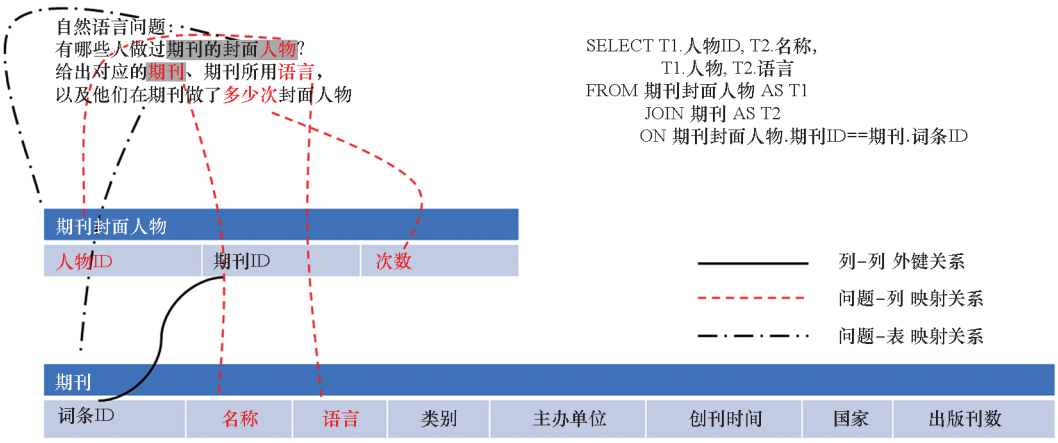

图 1 DuSQL 数据集示例

Fig. 1 Example of DuSQL dataset

现有的致力于解决自然语言询问和数据库模式之间的映射的研究主要分为两类,分别是基于匹配的方法[9] 和基于学习的方法[12,14,18]。IRNet[9] 是经典的基于匹配的方法,它使用简单有效的字符串匹配方法来连接自然语言问题和数据库模式。关系感知的文本到结构化查询语言(relation aware text-to-SQL, RAT-SQL[12])是经典的基于学习的方法,它使用关系感知 Transformer 来全局学习自然语言问题和数据库模式之间预先定义好的关系。通过分析上述方法的案例,发现这两类方法仍然存在泛化能力不足的问题。因为其都采用了隐式地学习自然语言询问中的单词与数据库模式中的表/列名称之间的映射。模型在处理跨域 NL2SQL 问题时并不能很好地捕捉自然

语言询问和数据库模式之间的映射。基于这个想法,本文引入了一个辅助任务来增强现有的模型,辅助任务通过预测自然语言询问和数据库模式图中节点之间的关系,从而提高模型的能力。

具体来说,在传统的 NL2SQL 模型的基础上设计了新的辅助任务。在此辅助任务中,显式地对自然语言询问中的单词与数据库模式中的表/列名称之间的映射进行建模。通过将主 SQL 生成任务与辅助任务相结合,以多任务学习的方式提高模型的能力。在实践中,使用在 NL2SQL 任务中表现良好的 RAT-SQL 作为原型模型,通过结合额外的辅助任务,并在中文 DuSQL 数据集上进行模型效果对比。

1　NL2SQL 算法模型背景介绍

1.1　关系感知 Transformer

随着大规模语言模型的成功,Transformer[19] 架构大量在自然语言处理(natural language process,NLP)任务中使用自注意力机制来编码序列。Transformer 块是由自注意力层堆叠而成,每一层通过 H 头将 $\boldsymbol{x}_i$ 转换 $\boldsymbol{y}_i$,计算公式如下:

$$\boldsymbol{e}_{ij}^{(h)}=\frac{\boldsymbol{x}_i\boldsymbol{W}_{\mathrm{Q}}^{(h)}(\boldsymbol{x}_j\boldsymbol{W}_{\mathrm{K}}^{(h)})^{\mathrm{T}}}{\sqrt{\frac{d_z}{H}}} \tag{1}$$

$$\boldsymbol{\alpha}_{ij}^{(h)}=softmax_j(\{\boldsymbol{e}_{i,j}^{(h)}\}) \tag{2}$$

$$\boldsymbol{z}_i^{(h)}=\sum_{j=1}^{n}\boldsymbol{\alpha}_{ij}^{(h)}\boldsymbol{x}_i\boldsymbol{W}_{\mathrm{V}}^{(h)} \tag{3}$$

$$\boldsymbol{z}_i=Concat(\boldsymbol{z}_i^{(1)},\cdots,\boldsymbol{z}_i^{(H)}) \tag{4}$$

$$\overline{\boldsymbol{y}_i}=LayerNorm(\boldsymbol{x}_i+\boldsymbol{z}_i) \tag{5}$$

$$\boldsymbol{y}_i=LayerNorm(\overline{\boldsymbol{y}_i}+FC\{\mathrm{ReLU}[FC(\overline{\boldsymbol{y}_i})]\}) \tag{6}$$

其中,$\boldsymbol{h}$ 表示头索引,d_z 表示 $\boldsymbol{z}_i^{(h)}$ 的隐藏层维度,$\boldsymbol{\alpha}_{ij}^{(h)}$ 表示注意力权重,*Concat* 表示拼接操作,*LayerNorm* 表示层归一化[20],*FC* 表示全连接层。Transformer 函数可以简单地表示为:

$$\boldsymbol{Y}=Transformer(\boldsymbol{X}) \tag{7}$$

式中,$\boldsymbol{Y}=\{\boldsymbol{y}_i\}_{i=1}^{|X|}$,$\boldsymbol{X}=\{\boldsymbol{x}_i\}_{i=1}^{|X|}$,$|\boldsymbol{X}|$是序列长度。

关系感知 Transformer[12] 是原始 Transformer 的一个重要拓展,它将输入序列作为有标注的有向图。关系感知 Transformer 考虑到了输入元素的关系,在式(1)和式(3)中引入了关系信息。输入元素 $\boldsymbol{x}_i$ 和 $\boldsymbol{y}_i$ 之间的关系通过向量 $\boldsymbol{r}_{ij,\mathrm{K}}$和 $\boldsymbol{r}_{ij,\mathrm{V}}$来表示。它们表示为包含在自注意力层中的偏差,计算公式如下:

$$\boldsymbol{e}_{ij}^{(h)}=\frac{\boldsymbol{x}_i\boldsymbol{W}_{\mathrm{Q}}^{(h)}(\boldsymbol{x}_j\boldsymbol{W}_{\mathrm{K}}^{(h)}+\boldsymbol{r}_{ij,\mathrm{K}})^{\mathrm{T}}}{\sqrt{\frac{d_z}{H}}} \tag{8}$$

$$\boldsymbol{z}_i^{(h)}=\sum_{j=1}^{n}\boldsymbol{\alpha}_{ij}^{(h)}(\boldsymbol{x}_i\boldsymbol{W}_{\mathrm{V}}^{(h)}+\boldsymbol{r}_{ij,\mathrm{V}}) \tag{9}$$

其中,$\boldsymbol{r}_{ij,\mathrm{K}}$和 $\boldsymbol{r}_{ij,\mathrm{V}}$在不同的头中是共享的。对于每一层关系感知的 Transformer,更新过程可以简单地表示为:

$$\boldsymbol{Y}=RAT(\boldsymbol{X},\boldsymbol{R}) \tag{10}$$

式中,$\boldsymbol{R}=\{\boldsymbol{r}\}_{i=1,j=1}^{|X|,|X|}$是序列元素之间的关系矩阵,$\boldsymbol{r}_{ij}$表示的是序列元素 i 和序列元素 j 之间的关系类型。

1.2　基于抽象语法树的 SQL 生成解码器

SQL 生成解码器是基于抽象语法树[21]。它通过深度优先遍历顺序的抽象语法树生成 SQL$\mathcal{Y}$,用长短期记忆(long short-term memory,LSTM[22])网络输出解码器动作序列。这些动作有两种方式:一种是将最后生成的节点扩展为语法规则,称为 ApplyRule;另一种遍历到叶节点时,从数据库模式中选择一个列/表名称,称为 SelectColumn/SelectTable。

具体来说,$\Pr(P|\mathcal{Y})=\prod_t\Pr(\boldsymbol{a}_t|\boldsymbol{a}_{<t},\mathcal{Y})$,其中,$\mathcal{Y}=f_{\mathrm{enc}}(\boldsymbol{G}_{\mathrm{Q}})$是自然语言询问和数据库模式的最终编码,$\boldsymbol{a}_{<t}$是所有之前的动作集合。在基于抽象语法树的 SQL 解码器中,LSTM 的状态 $\boldsymbol{m}_t$ 和 $\boldsymbol{h}_t$ 按如下方式更新:

$$\boldsymbol{m}_t,\boldsymbol{h}_t=f_{\mathrm{LSTM}}([\boldsymbol{a}_{t-1}|\boldsymbol{z}_t|\boldsymbol{h}_{pt}|\boldsymbol{a}_{pt}|\boldsymbol{n}_{ft}],\boldsymbol{m}_{t-1},\boldsymbol{h}_{t-1}) \tag{11}$$

式中,$\boldsymbol{m}_t$ 是 LSTM 的单元状态,$\boldsymbol{h}_t$ 是 LSTM 在 t 时刻的输出,$\boldsymbol{a}_{t-1}$是前一个动作的嵌入,pt 是当前抽象语法树节点的父节点展开对应的步骤,$\boldsymbol{n}_{ft}$是当前节点类型的嵌入。最后,$\boldsymbol{z}_t$ 是上下文的表示,它是通过 $\boldsymbol{h}_{t-1}$使用多头注意力在 $\mathcal{Y}$ 上计算得出的。

对于 ApplyRule[$\boldsymbol{R}$] 动作,计算

$$\Pr(\boldsymbol{a}_t=\mathrm{ApplyRule}[\boldsymbol{R}]|\boldsymbol{a}_{<t},\boldsymbol{y})=softmax_{\boldsymbol{R}}(g(\boldsymbol{h}_t)) \tag{12}$$

式中,$g(\cdot)$是 2 层激活函数为 tanh 的全连接层。

对于 SelectColumn 动作,计算

$$\bar{\boldsymbol{\lambda}}_i=\frac{\boldsymbol{h}_t\boldsymbol{W}_{\mathrm{Q}}^{\mathrm{sc}}(\boldsymbol{y}_i\boldsymbol{W}_{\mathrm{K}}^{\mathrm{sc}})^{\mathrm{T}}}{\sqrt{d_x}} \tag{13}$$

$$\boldsymbol{\lambda}_i=softmax_i\{\bar{\boldsymbol{\lambda}}_i\} \tag{14}$$

$$\Pr(\boldsymbol{a}_t=\mathrm{SelectColumn}[i]|\boldsymbol{a}_{<t},\boldsymbol{y})=\sum_{j=1}^{|y|}\boldsymbol{\lambda}_j\boldsymbol{L}_{j,i}^{\mathrm{col}} \tag{15}$$

SelectTable 和 SelectColumn 是类似的。

图 2 生动说明了解码器是如何应用 ApplyRule 以及 SelectColumn/SelectTable 来生成 SQL 的。其自然语言询问为“哪些城市人口比青海省全省的还多?”,对应的 SQL 为“SELECT 名称 FROM 中国城市 WHERE 人口 >(SELECT SUM(人口) FROM 中国城市 WHERE 所属省 = ‘青海’)”。

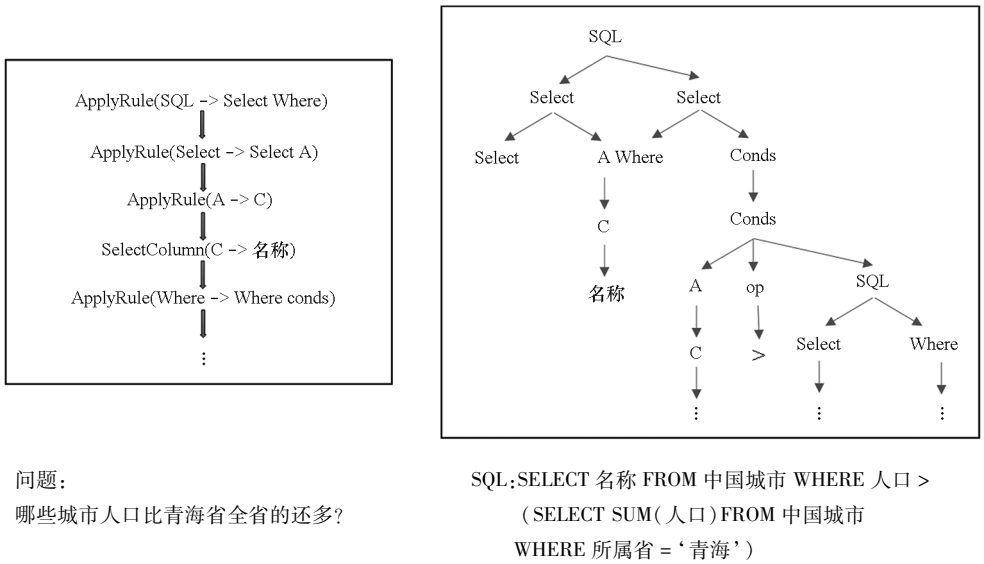

图 2 解码器操作图解

Fig. 2 Decoder operation diagram

1.3 RAT-SQL

基于关系感知 Transformer 的 SQL 生成模型 RAT-SQL 采用经典的 Seq-to-Seq 架构。在编码器中,嵌入层初始化阶段生成自然语言询问和数据库模式的初始向量,然后经过堆叠 8 层的关系感知 Transformer 来联合编码自然语言询问和数据库模式,其自然语言询问和数据库模式之间的依赖关系利用关系感知 Transformer 巧妙地融合到模型中,在解码阶段采用基于抽象语法树的 SQL 生成解码器来预测 SQL。

2 辅助任务设计

2.1 问题描述

设自然语言询问为 $Q=(q_1,q_2,\cdots,q_{|N|})$,$|N|$ 是自然语言询问的长度。Q 所对应的数据库模式为 $S=T\cup C$。其生成的目标 SQL 为 y。数据库模式 S 中包含了多张数据库表 $T=\{t_1,t_2,\cdots\}$ 以及各个数据库表中包含的数据库列 $C=\{c_1^{t_1},c_2^{t_1},\cdots,c_1^{t_2},c_1^{t_2},\cdots\}$。每一张表 t_i 是由其名字来描述,并且进一步由 (t_{i1},t_{i2}) 几个词来描述。类似地,使用 $(c_{j1}^{t_i},c_{j2}^{t_i},\cdots)$ 来表示列 $c_j^{t_i}\in t_i$。

2.2 数据库模式图构建

用有向图 $G=\langle V,E\rangle$ 表示数据库模式图。其图中节点 $V=C\cup T$ 是数据库模式中包含的数据库列名和数据库表名。图中的边 E 是数据库中预先存在的关系。在数据库模式中使用了一些典型的特定于数据库的关系,例如列与列之间的主键或外键关系、列属于表的关系。图 3 展示了数据库模式图的示例。

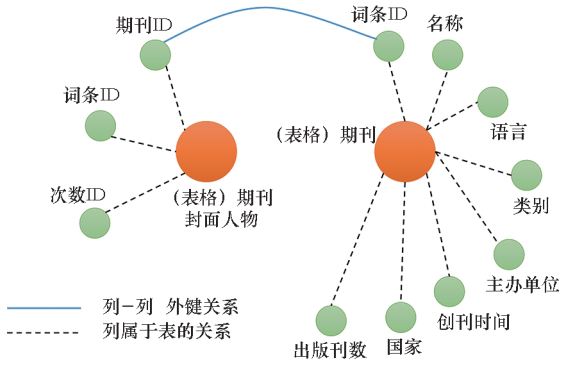

图 3 数据模式图示例

Fig. 3 Example of database schema graph

2.3 自然语言询问与数据库模式依赖关系建立

利用二部图来建模自然语言询问与数据库模式之间的依赖关系。具体来说 $G=\langle V,E\rangle$,其中 $V=Q\cup S$,Q 表示自然语言询问,S 表示数据库模式,E 表示自然语言询问和数据库模式之间的依赖关系。如表 1 所示,根据 SQL 中常用的关键字、聚合函数和操作预定义了 17 种依赖类型。具体来说,根据 N-gram 规则和利用 SQL 语句的特性而设计的触发函数来构建其间的依赖。例如,针对图 3 的数据库模式,自然语言询问为“中国

出版刊数最多的期刊出版了多少次?”,对应的 SQL 为“SELECT MAX(出版刊数) FROM 期刊 WHERE 国家 = ‘中国’”。根据 SQL 可以获取到提到的列(出版刊数)且可以找出为什么提到它们(SELECT 和 MAX),并得到自然语言询问中与该被提到的列逻辑相关的词汇(出版刊数最多的)。凭借这些细化的依赖关系类型,可以看出哪一列在自然语言询问中被提到以及为什么提到。

表 1　自然语言询问与数据库模式之间的依赖关系以及其解释

Tab. 1　Dependency relationships and their explanations between natural language queries and database schemas

依赖类型	描　述
None	没有依赖
SELECT	自然语言询问中提到带有关键字 SELECT 的数据库列,并且没有聚合函数
SELECT-AGG	自然语言询问中提到带有关键字 SELECT 的数据库列,并且有聚合函数
JOIN	自然语言询问中提到带有关键字 JOIN 的数据库列
WHERE	自然语言询问中提到带有关键字 WHERE 的数据库列
WHERE-OP	自然语言询问中通过 WHERE 子句中的操作修改数据库列
WHERE-VALUE	自然语言询问中存在数据库列下的值,并在 WHERE 子句中提到
GROUP-BY	自然语言询问中提到带有关键字 GROUP BY 的数据库列,并且没有聚合函数
GROUP-BY-AGG	自然语言询问中提到带有关键字 GROUP BY 的数据库列,并且有聚合函数
HAVING	自然语言询问中提到带有关键字 HAVING 的数据库列,并且没有聚合函数
HAVING-AGG	自然语言询问中提到带有关键字 HAVING 的数据库列,并且有聚合函数
HAVING-OP	自然语言询问中通过 HAVING 子句中的操作修改数据库列
HAVING-VALUE	自然语言询问中存在数据库列下的值,并在 HAVING 子句中提到
ORDER-BY	自然语言询问中提到带有关键字 ORDER BY 的数据库列,并且没有聚合函数
ORDER-BY-AGG	自然语言询问中提到带有关键字 ORDER BY 的数据库列,并且有聚合函数
ORDER-BY-ORDER	自然语言询问中通过 ORDER BY 升序或降序操作修改数据库列
LIMIT-VALUE	自然语言询问中存在 LIMIT 之后的值

2.4　依赖关系预测

本文设计的辅助任务通过依赖关系预测来显式地建模自然语言询问和数据库模式之间的依赖关系。如图 4 所示,在依赖关系建立阶段已经生成了有向边和标签,其中预定义的依赖边以及标签从数据库模式的列指向自然语言询问中的单词。

在解码阶段,首先使用多头注意力机制[19]以及单层前馈神经网络[19](feed-forward network, FFN),融合编码阶段生成的自然语言询问的单词嵌入 $\boldsymbol{X}_q$ 和数据库模式图中的节点 $\boldsymbol{x}_{s_i}$ 计算得出上下文向量 $\bar{\boldsymbol{x}}_{s_i}$,具体如式(16)~(18)所示。

$$\boldsymbol{\gamma}_{ji}^{h} = softmax_j \frac{(\boldsymbol{x}_{s_i}\boldsymbol{W}_{\mathrm{sq}}^{h})(\boldsymbol{x}_{q_j}\boldsymbol{W}_{\mathrm{sk}}^{h})^{\mathrm{T}}}{\sqrt{d/H}} \tag{16}$$

$$\bar{\boldsymbol{x}}_{s_i} = \Big(\Big\|_{h=1}^{H} \sum_j \boldsymbol{\gamma}_{ji}^{h}\boldsymbol{x}_{q_j}\boldsymbol{W}_{\mathrm{sv}}^{h}\Big)\boldsymbol{W}_{\mathrm{so}} \tag{17}$$

$$\bar{\boldsymbol{x}}_{s_i} = FFN(\bar{\boldsymbol{x}}_{s_i}) \tag{18}$$

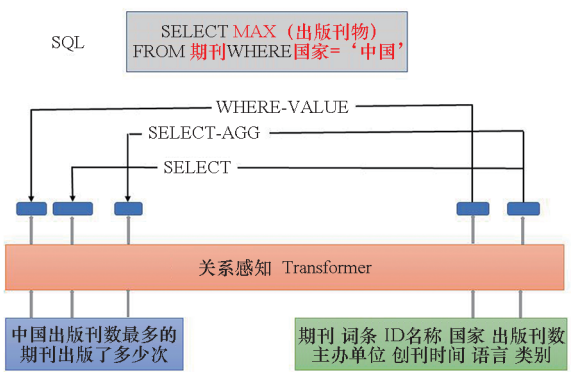

图 4　辅助任务结构图

Fig. 4　Structural graph of auxiliary task

其中,$\boldsymbol{W}_{\mathrm{sq}}^{h}$、$\boldsymbol{W}_{\mathrm{sk}}^{h}$、$\boldsymbol{W}_{\mathrm{sv}}^{h} \in \mathbb{R}^{d\times d/H}$ 和 $\boldsymbol{W}_{\mathrm{so}} \in \mathbb{R}^{d\times d}$ 是网络中可训练参数,“‖”表示向量的拼接。式(16)计算一个数据库模式项和自然语言询问所有单词之间的注意力权重 $\boldsymbol{\gamma}_{ji}^{h}$,然后通过多头注意力和单层的前馈神经网络产生一个上下文向量 $\bar{\boldsymbol{x}}_{s_i}$。之后

使用双仿射注意力机制[23]来捕捉上下文向量 $\bar{\boldsymbol{x}}_{s_i}$ 和数据库模式图中节点 $\boldsymbol{x}_{s_i}$ 之间的复杂依赖。

$$\boldsymbol{y}'_{x_{s_i}}, \bar{\boldsymbol{x}}_{s_i} = softmax(Biaffine(\boldsymbol{x}_{s_i}, \bar{\boldsymbol{x}}_{s_i})) \quad (19)$$

式中，$\boldsymbol{y}'_{x_{s_i}}, \bar{\boldsymbol{x}}_{s_i}$ 是预测上下文向量和数据库模式之间的依赖关系。Biaffine 双仿射注意力机制是双线性映射的推广，其中还包括了上下文向量和数据库模式节点之间的乘法交互，详细如下：

$$Biaffine(\boldsymbol{x}_1, \boldsymbol{x}_2) = \boldsymbol{x}_1 \boldsymbol{U} \boldsymbol{x}_2^{\mathrm{T}} + [\boldsymbol{x}_1; \boldsymbol{x}_2] \boldsymbol{W} + \boldsymbol{b} \quad (20)$$

式中，$\boldsymbol{U}$、$\boldsymbol{W}$ 和 $\boldsymbol{b}$ 是网络可训练参数。辅助任务预测结果和标签之间的损失通过交叉熵损失函数来计算。此辅助任务以多任务训练的方式和主生成 SQL 任务结合训练。

3 实验结果与分析

3.1 DuSQL 数据集

为验证所提出模型的有效性，在 DuSQL 中文数据集上进行评估实验。DuSQL 中的自然语言询问分为 5 类，分别是匹配、排序、聚合、计算和其他。百度利用其公司的搜索引擎日志，从中随机挑选一些问题，然后手动将问题分类。其自然语言询问和数据库模型跨域 160 多个领域，覆盖生活的方方面面，例如大学、城市、歌手、电影、航空公司、动物等。

将数据集拆分为训练、开发、测试，以便数据库模式在三个子集之间不重叠，即针对同一个数据库的所有问题/SQL 对都在同一个子集中，开发、测试中的数据库模式不会出现在训练中，所以是跨域 NL2SQL 问题。最后将 200 个数据库拆分为 160/17/23，将 23 797 个问题/SQL 对拆分为 18 602/2 039/3 156。

3.2 对比模型

为客观定量评价模型的有效性，将模型与下述表现较好的模型进行比较。IRNet[9]设计了一种称为 SemQL 的中间表示，用于编码比 SQL 更高级别的抽象结构，然后使用基于语法的解码器来合成 SemQL 查询。IRNet-Ext[17]是 IRNet 的拓展版本，通过简单地扩展 IRNet 来解析计算问题和预测值，以适应中文数据集 DuSQL 的特点。RAT-SQL[12]使用关系感知 Transformer 来联合编码自然语言询问以及数据库模式，能够很好地处理跨域 NL2SQL 任务，取得了很好的效果。

3.3 评价指标

采用 NL2SQL 任务中最常见的精确匹配率作为评价指标。精确匹配率指，预测得到的 SQL 语句与标准 SQL 语句精确匹配成功的问题占比。为了处理由成分顺序带来的匹配错误，当前精确匹配评估将预测的 SQL 语句和标准 SQL 语句按照 SQL 关键词分成多个子句，每个子句中的成分表示为集合，当两个子句对应的集合相同，则两个子句相同；当两个 SQL 所有子句都相同，则两个 SQL 精确匹配成功。DuSQL 数据集在模型评估时可以考虑“值”的准确性，其中“值”包含在 WHERE 从句、HAVING 从句和 LIMIT 从句中。

3.4 嵌入层初始化

采用预训练方法初始化自然语言问题词和数据库模式中的表/列的输入嵌入。就预训练向量而言，首先采用 Glove[24]来获得自然语言询问中的单词和数据库模式图中的表/列的初始化嵌入，随后自然语言询问通过双向的 LSTM 来获得其进一步的嵌入。预训练语言模型对于学习通用语言的初始化嵌入是非常有效的。为了进一步研究方法的有效性，也采用 ERNIE[25]预训练语言模型对自然语言询问以及数据库模式进行了初始化嵌入。

3.5 实验设置

本文模型使用 PyTorch[26]框架实现。采用 Glove 初始化嵌入层向量，单词的嵌入维度为 300，训练时最频繁的 50 个单词固定，剩下的单词微调。双向 LSTM 的隐藏层大小为 128，并且使用循环神经网络 dropout 方法[27]，其 dropout 率为 0.2。编码器以及辅助任务中的多头注意力机制的注意力头数为 8，dropout 率为 0.2。使用 AdamW[28]预热学习率梯度优化算法，训练时的预热率为 0.1。当嵌入层的初始化为 Glove 时，学习率为 5×10^{-4}，权重衰减系数为 1×10^{-4}，训练周期数为 100。当使用 ERNIE 预训练语言模型微调时，学习率为 1×10^{-5}，权重衰减系数为 0.1，训练周期数为 200。

3.6 实验结果

在 DuSQL 数据集上的主要实验结果见表 2，表中“考虑值”即为在评价生成的 SQL 时，会对出现在 WHERE、HAVING 和 LIMIT 从句中的数值也进行评估。

从实验结果来看，辅助任务增强的中文跨域 NL2SQL 算法得益于辅助任务在解码端显式地建模自然语言询问和数据库模式之间的依赖关系，增强了自然语言询问和数据库模式的表示。同时辅助任务的加入避免了算法出现过拟合，模型不再简单地记忆数据库模式中的模式项，而是更加

表 2　各模型在 DuSQL 数据集上的实验结果

Tab. 2　Experimental results of each model on the DuSQL dataset

%

模型	不考虑值		考虑值	
	开发集	测试集	开发集	测试集
IRNet	38.4	34.2	18.4	15.4
IRNet-Ext	59.8	54.3	56.2	50.1
IRNet-Ext + ERNIE	64.9	59.1	61.3	53.9
RAT-SQL	70.2	61.5	60.6	50.3
RAT-SQL + ERNIE	76.1	65.8	64.1	53.2
本文模型	72.7	63.8	63.5	52.0
本文模型 + ERNIE	78.8	67.2	67.4	55.8

理解自然语言询问想要获得的结果。因此,辅助任务增强的中文跨域 NL2SQL 算法在 DuSQL 数据集上取得了更优异的成绩,相对于在 NL2SQL 任务中表现较好的原始模型 RAT-SQL 在不考虑值评估的情况下,在测试集合上取得 2.3% 的精确匹配率提升;在考虑值评估的情况下在测试集合上取得 1.7% 精确匹配率的提升。此外,因为预训练语言模型可以大幅提高算法处理自然语言任务的效果,所以也采用了预训练语言模型 ERNIE 来初始化各个算法的嵌入。由实验结果可以得到,采用 ERNIE 初始化 RAT-SQL 的嵌入相对于 Glove 的初始化嵌入,在考虑值评估的情况下在测试集合上取得了 2.9% 的精确匹配率提升。同时,提出的辅助任务增强的中文跨域 NL2SQL 算法在 ERNIE 增强下相对于 RAT-SQL + ERNIE 也取得了更好的效果。在不考虑值评估的情况下,在测试集上达到了 67.2% 的精确匹配率,相对于 RAT-SQL + ERNIE 取得了 1.4% 的精确匹配率提升;在考虑值评估的情况下,在测试集上达到了 55.8% 的精确匹配率,相对于 RAT-SQL + ERNIE 取得了 2.6% 精确匹配率的提升。

3.7　案例分析

如图 5 所示,案例的生成是在 ERNIE 预训练语言模型下微调的。由案例可以看出,模型在添加辅助任务后,在遇到跨越多表 JOIN 的情况时表现更好,并且模型对自然语言询问中想要得到的信息有更清晰的认知,模型能够更好地区分数据库模式中哪些表中的元素对是最终生成 SQL 所需要的。

图 5　辅助任务添加前后的案例分析

Fig. 5　Case study before and after adding auxiliary task

4　结论

本文对跨域 NL2SQL 任务当前的模型做分析后,发现当前的模型大多都隐式地建模自然语言询问和数据库模式之间的依赖关系,并不能很好地捕捉两者之间的依赖关系。基于此想法,提出了一个辅助任务,以多任务训练的方式结合原始模型。在中文数据集 DuSQL 上的实验证明了添加辅助任务后的模型相对于原始模型取得了更好的效果。

1)通过辅助任务来显式建模自然语言询问和数据库模式之间的依赖关系,从而让模型处理跨域 NL2SQL 问题时能够更好地捕捉其间的依赖关系,提高模型的泛化能力。

2)辅助任务是通过多任务训练的方式和生成 SQL 的主任务结合训练,所以提出的辅助任务可以迅速地迁移到别的 NL2SQL 模型,提高原始模型的性能。

参考文献(References)

[1]　孟小峰, 王珊. 数据库自然语言查询系统 Nchiql 中语义依存树向 SQL 的转换[J]. 中文信息学报, 2001,15(5): 40 – 45.
MENG X F, WANG S. Method of transforming semantic dependent tree to SQL in Nchiql[J]. Journal of Chinese Information Processing, 2001, 15(5): 40 – 45. (in Chinese)

[2]　CAI R C, XU B Y, ZHANG Z J, et al. An encoder-decoder framework translating natural language to database

queries[C]// Proceedings of the 27th International Joint Conference on Artificial Intelligence, 2018: 3977 – 3983.

[3] HWANG W, YIM J, PARK S, et al. A comprehensive exploration on WikiSQL with table-aware word contextualization[C]//Proceedings of 33rd Conference on Neural Information Processing Systems, 2019.

[4] YU T, LI Z F, ZHANG Z L, et al. TypeSQL: knowledge-based type-aware neural text-to-SQL generation[C]// Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2018: 588 – 594.

[5] XU X J, LIU C, SONG D. SQLNet: generating structured queries from natural language without reinforcement learning[EB/OL]. [2021 – 03 – 22]. https://arxiv.org/pdf/1711.04436.pdf.

[6] 曹金超, 黄滔, 陈刚, 等. 自然语言生成多表SQL查询语句技术研究[J]. 计算机科学与探索, 2020, 14(7): 1133 – 1141.
CAO J C, HUANG T, CHEN G, et al. Research on technology of generating multi-table SQL query statement by natural language[J]. Journal of Frontiers of Computer Science and Technology, 2020, 14(7): 1133 – 1141. (in Chinese)

[7] ZHONG V, XIONG C M, SOCHER R. Seq2SQL: generating structured queries from natural language using reinforcement learning[EB/OL]. [2021 – 05 – 01]. https://arxiv.org/pdf/1709.00103.pdf.

[8] YU T, ZHANG R, YANG K, et al. Spider: a large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-SQL task[C]//Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, 2018: 3911 – 3921.

[9] GUO J Q, ZHAN Z C, GAO Y, et al. Towards complex text-to-SQL in cross-domain database with intermediate representation[C]//Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, 2019: 4524 – 4535.

[10] BOGIN B, BERANT J, GARDNER M. Representing schema structure with graph neural networks for text-to-SQL parsing[C]//Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, 2019: 4560 – 4565.

[11] BOGIN B, GARDNER M, BERANT J. Global reasoning over database structures for text-to-SQL parsing[C]//Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, 2019: 3659 – 3664.

[12] WANG B L, SHIN R, LIU X D, et al. RAT-SQL: relation-aware schema encoding and linking for text-to-SQL parsers[C]//Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, 2020: 7567 – 7578.

[13] RUBIN O, BERANT J. SmBoP: semi-autoregressive bottom-up semantic parsing[C]//Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2021: 311 – 324.

[14] LIN X V, SOCHER R, XIONG C M. Bridging textual and tabular data for cross-domain text-to-SQL semantic parsing[C]//Proceedings of the Findings of the Association for Computational Linguistics, 2020: 4870 – 4888.

[15] SUN N Y, YANG X F, LIU Y F. TableQA: a large-scale Chinese text-to-SQL dataset for table-aware SQL generation[EB/OL]. [2021 – 06 – 02]. https://arxiv.org/pdf/2006.06434.pdf.

[16] ZHANG X Y, YIN F J, MA G J, et al. M-SQL: multi-task representation learning for single-table Text2SQL generation[J]. IEEE Access, 2020, 8: 43156 – 43167.

[17] WANG L J, ZHANG A, WU K, et al. DuSQL: a large-scale and pragmatic Chinese text-to-SQL dataset[C]//Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, 2020: 6923 – 6935.

[18] LEI W Q, WANG W X, MA Z X, et al. Re-examining the role of schema linking in text-to-SQL[C]//Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, 2020: 6943 – 6954.

[19] VASWANI A, SHAZEER N, PARMAR N, et al. Attention is all you need[EB/OL]. [2021 – 08 – 21]. https://arxiv.org/pdf/1706.03762.pdf.

[20] BA J L, KIROS J R, HINTON G E. Layer normalization[EB/OL]. [2021 – 09 – 15]. https://arxiv.org/pdf/1607.06450.pdf.

[21] YIN P C, NEUBIG G. A syntactic neural model for general-purpose code generation[C]//Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, 2017: 440 – 450.

[22] HOCHREITER S, SCHMIDHUBER J. Long short-term memory[J]. Neural Computation, 1997, 9(8): 1735 – 1780.

[23] DOZAT T, MANNING C D. Deep biaffine attention for neural dependency parsing[EB/OL]. [2021 – 10 – 10]. https://arxiv.org/pdf/1611.01734.pdf.

[24] PENNINGTON J, SOCHER R, MANNING C. Glove: global vectors for word representation[C]//Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, 2014: 1532 – 1543.

[25] ZHANG Z Y, HAN X, LIU Z Y, et al. ERNIE: enhanced language representation with informative entities[C]// Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, 2019: 1441 – 1451.

[26] PASZKE A, GROSS S, MASSA F, et al. PyTorch: an imperative style, high-performance deep learning library[C]// Proceedings of the 33rd International Conference on Neural Information Processing Systems, 2019.

[27] GAL Y, GHAHRAMANI Z. A theoretically grounded application of dropout in recurrent neural networks[C]// Proceedings of the 30th International Conference on Neural Information Processing Systems, 2016.

[28] LOSHCHILOV I, HUTTER F. Decoupled weight decay regularization[C]//Proceedings of the International Conference on Learning Representations, 2019.

(编辑: 熊立桃, 杨琴)