融合字段类型与文本匹配的中文问句解析

纪相存[1,2] 李大林[1] 彭晓东[1*]

[1](中国科学院国家空间科学中心 北京 101499)

[2](中国科学院大学 北京 100049)

摘 要 自然语言转 SQL 语句技术可以帮助用户使用数据库,而 WikiSQL 数据集对表格内容的保护一定程度上限制了模型的使用,基于此,提出一种融合字段类型与文本匹配的中文问句解析方法。基于 SQL 结构分解问句解析任务,通过字段类型相关的分隔符将表结构信息结合到 RoBERTa 编码器输入中,并使用结合编辑距离与语义词典的文本匹配来使模型更加鲁棒。在中文数据集 TableQA 进行测试,该方法取得了最好的效果,正确率达到 93.44%。

关键词 自然语言转 SQL 语句 表结构信息 SQL 结构 文本匹配

中图分类号 TP **文献标志码** A DOI:10.3969/j.issn.1000-386x.2024.07.028

CHINESE QUESTION PARSING BASED ON FIELD TYPES AND TEXT MATCHING

Ji Xiangcun[1,2] Li Dalin[1] Peng Xiaodong[1*]

[1](*National Space Science Center, Chinese Academy of Science, Beijing 101499, China*)

[2](*University of Chinese Academy of Science, Beijing 100049, China*)

Abstract Translating natural language questions to SQL statements can help more users to obtain what they want from the database. The protection of table content by the English dataset WikiSQL limits the migration and use of the model to a certain extent. In order to solve this problem, this paper proposes a Chinese question parsing methods combining the field types with text matching. The task was decomposed based on the SQL structure. The table structure information was combined into the input of the Roberta encoder through the column separators related to the field types. The text matching method combining the edit distance and semantic dictionary was used to make the model more robust. This method was tested on the more difficult Chinese dataset TableQA. The accuracy rate was up to 93.44% and the result verified that the method was efficient.

Keywords Natural language to SQL statements Table structure SQL structure Text matching

0 引 言

当今数据库中存储的数据越来越复杂,用户依据数据库分析的需求也越来越强,但使用计算机语言查询的方式对用户的门槛较高。数据库的自然语言接口技术在近些年受到越来越多的关注,它们极大程度上帮助了没有相关计算机知识的人依靠数据库获取相关信息。

自然语言转 SQL 语句(Natural Language to SQL Statements,NL2SQL)作为数据库的自然语言接口的重要技术,传统方法大多使用高质量的语法树和词典来构造语义解析器[1],从问句和表格中匹配出 SQL 语句需要的参数,但应用范围不广,正确率也没有保障。Salesforce[2]提出的标注 NL2SQL 数据集 WikiSQL 极大地促进了神经网络方法在 NL2SQL 问题上的发展。

WikiSQL 是目前最大的 NL2SQL 数据集,处理的是单表问题。Dong 等[3]提出了一种不需要人工设计语法的端到端语义分析器,使用了两个长短时记忆网络(Long and Short-term Memory,LSTM)分别作为编码器和解码器,将自然语言转换为逻辑语句。WikiSQL 的基线模型 Seq2SQL[2]就基于这种端到端网络思想,

收稿日期:2021 - 03 - 21。**纪相存**,硕士生,主研领域:自然语言处理。**李大林**,副研究员。**彭晓东**,研究员。

将输出空间限制到 SQL 语句、问题与表格内容上。Xu 等[4]根据 SQL 语法构造出一个结构并基于其分别预测其中的参数,这种基于 SQL 结构的思想被很多当前的主流方法采用。同时得益于自然语言预训练模型的发展[5-6],Hwang 等[7]提出了使用 BERT[8]作为编码器的 SQLova 网络,He 等[9]提出了以 BERT 变体 MT-DNN[5]作编码器的 X-SQL 网络,它们都是基于 SQL 结构对多个子任务进行学习,前者在子任务中使用 LSTM 网络,后者借助文本增强使用了更简化的全连接网络结构。Lyu 等[10]使用了更多表格信息作为输入,使模型更加简化,不再需要额外的池化或者编码层。

TableQA 是首个中文 NL2SQL 数据集,同样是处理单表问题。相比 WikiSQL,TableQA 中的 SQL 语句 SELECT 的字段数量可能为 2,WHERE 条件子句以多个为主,并且增加了 OR 条件。另外,TableQA 的表格内容可见,问句中可能并不包含 WHERE 条件子句中的条件值,难度更大。与 SQLNet 的思想相似,袁志祥等[11]提出的模型使用 BERT 作为编码器,双向 LSTM 作为子任务网络,但在 TableQA 上的准确率并不是很理想。Zhang 等[12]提出的 M-SQL 使用中文 BERT[13]作为编码器,下游子任务网络上更为简化。但是这些方法都没有很充分地利用表格结构信息,仅仅使用了表格的字段名,并且需要注意力机制等额外的计算来获得列字段特征。

另外,由于表格内容可见,问句中可能并不直接包含 SQL 语句中需要的条件值,因此本文在条件值预测的子任务后加入了文本匹配方法来进一步处理。文本匹配在方法上可以分为基于编辑距离的字符串匹配和基于语义的相似度匹配。基于编辑距离的字符串匹配是最开始用于文本匹配的方法之一,Levenshtein[14]提出了莱文斯坦距离,指两个字符串之间,由一个转换为另一个所需的最少编辑次数,许可的编辑操作包括替换、插入与删除。莱文斯坦比在莱文斯坦距离基础上将替换的操作编辑设为二,并利用字符串长度之和做归一化处理。Jaro 距离[15]将换位也算作一种特殊的编辑操作,Winkler 在此基础上考虑了字符串前缀相同部分的长度,提出了 Jaro-Winkler 距离[16],目的是匹配姓名相似度。编辑距离并不能度量两个短文本的相似性,目前基于语义的文本匹配可以更好地解决这一问题,基本可以分为两类:一类是根据已有知识本体或者分类体系计算;另一类是在大规模语料库的基础上进行模型学习。基于语料库的方法[17]对语料的依赖性比较强。基于知识本体的方法[18]依据人类的世界知识,构建知识本体有一定的复杂度,并且无法处理未统计的词语。由于本文处理的文本匹配中包含了缩略词、近义词甚至无效文本,单一的文本匹配方法无法处理这一问题。

通过以上分析,本文提出的中文问句解析方法的主体是基于 SQL 结构构建的网络模型,编码器使用了 RoBERTa 语言模型,并在编码器输入中融合了字段类型信息,根据列字段类型来决定使用哪种列分隔符,以其相应的编码器输出作为列特征,不再需要额外注意力机制的计算。此外,在 SQL 语句条件值预测的子任务网络后加入了结合编辑距离和语义词典的文本匹配方法,弥补了编辑距离方法无法处理短文本近义词的问题。将本文方法与其他主流的 NL2SQL 方法在中文数据集 TableQA 上进行对比,本文方法在 SQL 语句的逻辑形式和执行结果的准确率都取得了最好的效果。

1　任务描述

自然语言转 SQL 语句处理的问题是根据自然语言问句以及对应的表格自动生成可执行的 SQL 语句。中文单表数据集 TableQA 中的问题可以不直接包括 SQL 语句所需要的条件值,如问句“南航一共引进客机多少架啊”对应的 SQL 语句应该是“SELECT 数量 FROM TABLE WHERE 公司 = ‘南方航空’”,需要的条件值为“南方航空”,而问句中仅包含“南航”。

当前使用基于 SQL 结构的神经网络方法在处理单表 NL2SQL 问题上效果较好。相比端到端直接生成整个 SQL 语句的方法,利用 SQL 语法构造的 SQL 结构,可以更好地利用单表 SQL 语句中的关键字,如 SELECT、WHERE 和 AND/OR 等,将复杂的任务分解为多个子任务,解释性更强,子任务的结果再结合模板就可以生成可执行的 SQL 语句。本文也是基于这种思想,针对 SQL 结构划分出的所需参数来学习分析问题,使用文本匹配来提高网络对于不同表格的适应能力。针对 TableQA,本文使用的 SQL 结构如图 1 所示,需要对其中带有前缀的参数进行预测,* 标注部分表示可以存在多个也可以不存在。本文对所需的 SQL 语句有以下约束:

(1) 所有自然语言问题都可以转换为图 1 的 SQL 结构相应的 SQL 语句。

```
SELECT $S-AGG ( $S-COL )
        {, $S-AGG ( $S-COL )}*
FROM TABLE
WHERE $W-COL $W-OP $W-VAL
{$W-CON $W-COL $W-OP $W-VAL}*
```

图 1　本文使用的 SQL 结构

（2）$S-COL、$S-AGG 对应 SELECT 下选取列名以及相应的聚合函数，并且可以存在多对，其中 $S-AGG 可以为“”“AVG”“MAX”“MIN”“COUNT”和“SUM”。

（3）WHERE 下条件子句可以为多条，$W-CON 为条件子句连接符，可以为“”“AND”和“OR”。

（4）TABLE 表示自然语言问句相应的表格。

（5）$W-COL 表示 WHERE 子句中的列名。

（6）$W-OP 表示 WHERE 条件子句中的运算符，可以为“>”“<”“=”“!=”。

（7）$W-VAL 表示 WHERE 条件子句中的条件值，对应到问句中的相关文本用 $W-VAL-S 表示。

2 模型设计

2.1 整体方法

与 WikiSQL 数据集相比，TableQA 数据集的主要的难点在于问句中并不包含 SQL 语句所需要的真正的条件值。本文将 WHERE 条件子句中的条件值对应在问句中的文本称为问句条件值，并将其在 TableQA 数据集上做了进一步标注。这样就将问题划分成两个部分，一部分是与 WikiSQL 相似的问句结构解析任务，一部分是从问句条件值到真实条件值的文本匹配任务。本文模型的整体框架如图 2 所示，其中：梯形部分表示方法，长方形部分表示数据，主要可以分成编码器、子任务网络和文本匹配三个部分。本文的编码器采用在 BERT 模型之上改进了的 RoBERTa 语言模型[6]。基于图 1 的 SQL 结构，结合对 TableQA 数据集的分析，本文将问句结构解析任务分解成了以下八个子任务。

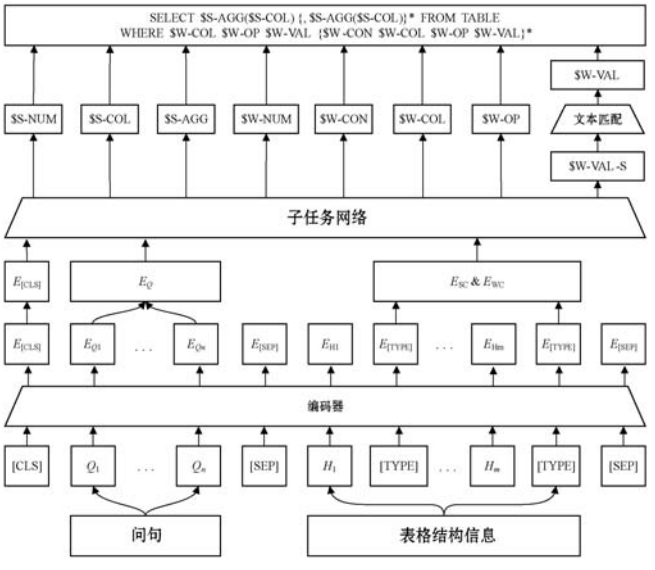

图 2 本文方法整体框架

（1）S-NUM，预测 SELECT 下选中的列的数量，这是一个分类问题，类别为 1 到表格列总数的所有整数。

（2）S-COL，预测表格中各列为 SELECT 下选中字段的概率，需要结合预测的 SELECT 下选中的列的数量来决定最终 SELECT 下的选中字段。

（3）S-AGG，预测 SELECT 下各个字段所需的聚合函数，这是一个分类问题，类别分别为“”“AVG”“MAX”“MIN”“COUNT”和“SUM”。对于每个字段都会预测相应聚合函数的概率。

（4）W-NUM，预测 WHERE 条件子句的数量，这是一个分类问题，类别分别为 1、2 和 3。

（5）W-CON，预测 WHERE 条件子句间的连接符，这是一个分类问题，类别分别为“”“AND”以及“OR”。

（6）W-COL，预测 WHERE 条件子句中的字段名。

（7）W-OP，预测 WHERE 条件子句中的运算符，这是一个分类问题，类别分别为“>”“<”“=”“!=”。

（8）W-VAL-S，预测 WHERE 条件子句中参数对应的问句条件值，其预测结果可能并不能直接在 SQL 语句中使用，需要文本匹配方法做进一步处理。

文本匹配负责将问句条件值转换成在 SQL 语句中使用的真实条件值，与一般的语义匹配不同，预测的问句条件值可能在前后包含问句中的无关信息，仅仅使用基于语义的文本匹配方法无法排除无关信息的干扰，因此本文选用了结合编辑距离与语义信息的文本匹配方法。

2.2 编码器与特征提取

本文的编码器采用中文 RoBERTa 预训练模型 Chinese-RoBERTa-wwm-ext-large。本方法提出字段类型分隔符，为了更加充分地利用列信息，使用[$TYPE_1$]、[$TYPE_2$]、[$TYPE_3$]代替[SEP]对列字段进行分隔，TableQA 数据集中字段数据类型包括“REAL”“TEXT”，针对不同的数据类型使用不同的分隔符[$TYPE_1$]、[$TYPE_2$]、[$TYPE_3$]，使编码器能更充分来融合列字段的类型特征。使用问题以及表格结构作为编码器的输入，表格结构包括列字段以及其类型，编码器的输入如下：

$$[\mathrm{CLS}],Q_1,Q_2,\cdots,Q_n,[\mathrm{SEP}],H_{1,1},H_{1,2},\cdots,H_{1,m1},[\mathrm{TYPE}_1],[\mathrm{TYPE}_2],[\mathrm{TYPE}_3],\cdots,H_{n,1},\cdots,H_{n,mn},[\mathrm{TYPE}_1],[\mathrm{TYPE}_2],[\mathrm{TYPE}_3],[\mathrm{SEP}] \quad (1)$$

式中：$Q_1,Q_2,\cdots,Q_n$ 表示自然语言问题预处理后再进行分词处理的结果；$H_{i,1}-H_{i,m1}$ 表示表格第 i 列字段的分词结果，问题与表格信息之间使用[SEP]标签分隔。

编码器输出表示为：

$$E_{[CLS]},E_{Q1},E_{Q2},\cdots,E_{Qn},E_{[SEP]},E_{H1,1},E_{H1,2},\cdots,E_{H1,m1},E_{[TYPE1]},E_{[TYPE2]},E_{[TYPE3]},\cdots,E_{Hn,1},\cdots,E_{Hn,mn},E_{[TYPE1]},E_{[TYPE2]},E_{[TYPE3]},E_{[SEP]} \quad (2)$$

式中:使用 E_Q 表示问题特征,本文使用 $E_{[TYPE1]}$、$E_{[TYPE2]}$ 和 $E_{[TYPE3]}$ 直接表示对应的列特征,不需要额外的注意力机制来对列特征进行提取,将前两个分隔特征在 WHERE 条件子句下的子任务中使用,第三个分隔特征在 SELECT 下的子任务中使用,分别用 E_{WC} 和 E_{SC} 表示。

2.3　子任务网络

本文将 NL2SQL 任务分解成了八个子任务,下面将对每个子任务的网络结构进行介绍。

第一个子任务,S-NUM,预测 SELECT 下选择字段的数量 N_{UMsel},使用整体特征 $E_{[CLS]}$ 进行预测,模型如式(3)所示。

$$P^{SN}(N_{UM_{sel}}) = \text{softmax}(W_{S\text{-}NUM}E_{[CLS]}) \quad (3)$$

式中:W 指网络模型参数。

第二个子任务 S-COL 和第三个子任务 S-AGG,分别预测 SELECT 下选择的字段 C_{OLsel} 和字段使用的聚合函数 A_{GGsel},使用列特征 E_{SC} 作为输入。模型分别如式(4)和式(5)所示。

$$P^{SC}(C_{OL_{sel}}) = \text{softmax}(W_{S\text{-}COL}E_{SC}) \quad (4)$$

$$P^{SA}(A_{GG_{sel}} \mid C_{OL_{sel}}) = \text{softmax}(W_{S\text{-}AGG}E_{SC}) \quad (5)$$

第四个子任务 W-NUM 和第五个子任务 W-CON,分别预测 WHERE 下条件子句的数量 $N_{UMwhere}$ 和连接符 $C_{ONwhere}$,使用整体特征 $E_{[CLS]}$ 进行预测。模型分别如式(6)和式(7)所示。

$$P^{WN}(N_{UM_{where}}) = \text{softmax}(W_{W\text{-}NUM}E_{[CLS]}) \quad (6)$$

$$P^{WCON}(C_{ON_{where}}) = \text{softmax}(W_{W\text{-}CON}E_{[CLS]}) \quad (7)$$

第六个子任务 W-COL 和第七个子任务 W-OP,分别预测 WHERE 下条件子句中的字段 $C_{OLwhere}$ 和运算符 O_{Pwhere},使用列特征 E_{WC} 作为输入。模型分别如式(8)和式(9)所示。

$$P^{WCOL}(C_{OL_{where}}) = \text{softmax}(W_{W\text{-}COL}E_{WC}) \quad (8)$$

$$P^{WO}(O_{P_{where}} \mid C_{OL_{where}}) = \text{softmax}(W_{W\text{-}OP}E_{WC}) \quad (9)$$

第八个子任务,W-VAL-S,预测 WHERE 下条件子句中的真实条件值对应于问题中的问句条件值。问句条件值可以看作是问句 Q 的一部分,因此再分解成预测问句条件值的起始位置 W-VAL-S-ST 与结束位置 W-VAL-S-ED 两个部分,使用问题特征 E_Q 与表格列特征 E_{WC} 作为输入,两个部分的模型分别如式(10)和式(11)所示。

$$P^{VS}(V_{ST_{where}} \mid C_{OL_{where}}) = \text{softmax}(W_{WS}\text{ReLU}(W_{QS}E_Q + W_{COL\text{-}S}E_{WC})) \quad (10)$$

$$P^{VE}(V_{ED_{where}} \mid C_{OL_{where}}) = \text{softmax}(W_{WE}\text{ReLU}(W_{QE}E_Q + W_{COL\text{-}E}E_{WC})) \quad (11)$$

2.4　文本匹配

英文数据集 WikiSQL 上各子任务中难度最大的是预测 SELECT 子句下的聚合函数,而目前处理 TableQA 方法的子任务中正确率较低的是 WHERE 下的条件值预测,因此本文在数据集中进一步标注了问题条件值,并结合文本匹配方法来进一步优化 WHERE 下的条件值预测结果。相比于直接使用 SQL 语句中的真实条件值,这种方式也更有利于子任务网络学习问题结构,以文本匹配方法来适应更多的数据。在本文方法中,文本匹配的任务是将问句条件值转换成 SQL 中使用的真实条件值,现有的大部分文本匹配方法都是针对长文本的语义匹配或者针对标准短词汇的近义匹配,但本任务中匹配的并不是这类文本,问题条件值往往是由多个标准词组成并且前后可能存在无效的词,并且包含一些缩略词。针对这些特点,本文设计了结合编辑距离与语义词典的文本匹配方法,如算法 1 所示。

算法 1　结合编辑距离与语义词典的文本匹配方法

输入:表格内容,语义词典,列序号,问题条件值 s_{rc}。

输出:真实条件值。

1. 根据列序号和表格内容判断列类型,如果为“real”,转 step 2,否则返回问题条件值 s_{rc}。
2. 根据列序号和表格内容提取出该列不重复元素集合 S。
3. 将 s_{rc} 与集合 S 中每个元素作编辑距离计算,得到最优得分 s_{core} 和相应的元素 v_{alue}。
4. 遍历语义词典的键值对(k, v),若 s_{rc} 中存在 k,转 step 5,否则转 step 7。
5. 则将 s_{rc} 中的 k 替换为对应的 v,然后与集合 S 中每个元素作编辑距离计算,得到最优得分 s 和相应元素 v。
6. 比较 s_{core} 与 s,若 s 优于 s_{core},则将 v 赋值给 v_{alue},将 s 赋值给 s_{core}。
7. 若遍历完成,转 step 6,否则转 step 4。
8. 返回元素 v_{alue}。

本文使用莱文斯坦比[13]来计算编辑距离,公式如式(2)所示。

$$r_{atio} = (s_{um} - l_{dist})/s_{um} \quad (12)$$

式中:r_{atio} 表示两个字符串的莱文斯坦比;s_{um} 表示两个字符串长度之和;l_{dist} 表示类编辑距离,类编辑距离的计算中,替换一个字符时距离加二,插入一个字符或删除一个字符时距离加一。

本文基于 TableQA 数据集构建了语义词典,将一些近义的标准词汇做了归纳总结,在算法 1 中通过在问题条件值搜索替换的方式将近义文本替换再作基于编辑距离的文本匹配。

2.5 模型训练与预测

在训练阶段,为了让子任务网络更好地学习问题结构,不使用文本匹配方法。将标注数据中的问题经过预处理后结合表格结构转换成编码器输入,经过编码器后以及各个子任务网络,对 S-COL 和 W-COL 两个子任务网络输出与标注值做相对熵计算,对其他子任务输出与标注值做交叉熵计算,将各子任务的损失之和作为总的损失。相对熵和交叉熵的计算方法分别如式(13)和式(14)所示。

$$Loss_{\mathrm{KL}}(p,q) = \sum_{i=1}^{n} p_i \log \frac{p_i}{q_i} \tag{13}$$

$$Loss_{\mathrm{CE}}(p,q) = -\sum_{i=1}^{n} (p_i \log(q_i) + (1-p_i)\log(1-q_i)) \tag{14}$$

式中:p 和 q 分别表示标注信息和网络预测输出;n 表示所有的可能类别数。训练时采用的学习率为 6E-6,批次大小设置为 16,数据集的循环训练次数为 60。

在预测阶段,在得到各子任务网络输出后,按照以下步骤生成最终的 SQL 语句:

(1) 根据 S-NUM、W-NUM 和 W-CON 子任务网络输出得到 SELECT 下字段数量 $S-NUM、WHERE 下的子句数量 W-NUM、子句连接符 $W-CON。

(2) 根据 S-COL 子网络输出的各列被选择概率,选取前 S-NUM 个字段作为所选字段,再根据 S-AGG 子网络输出得出每个字段对应的聚合函数。

(3) 根据 W-COL 和 W-OP 子任务网络输出作乘积,基于此排序然后选取前 $W-NUM 对作为子句下的($W-COL, $W-OP),依据 W-VAL-S 子任务网络输出结果得到问题条件值在问句中的起止位置,即可组成 WHERE 下条件子句对($W-COL, $W-OP, $W-VAL-S)。

(4) 遍历每个条件子句对,由 $W-COL 从表中提取出对应列的内容集合,然后与 $W-VAL-S 作文本匹配,得到真实条件值 $W-VAL,更新条件子句对为($W-COL, $W-OP, $W-VAL)。

(5) 最终,将以上所得的 SELECT 下的聚合函数-字段对($S-AGG, $S-COL)、条件子句对($W-COL, $W-OP, $W-VAL)、条件连接符 $W-CON 填入模板即可得到最终的可执行 SQL 语句。

3 实验与结果分析

3.1 实验设置

TableQA 数据集是第一个中文 NL2SQL 数据集。同样作为一个单表数据集,WikiSQL 为了保护数据,限制方法不能使用表格具体内容,因此问句中必须直接包含 WHERE 下条件子句的条件值,而 TableQA 对表格内容使用并没有限制,问句中也并不直接包含 WHERE 下条件子句的条件值。TableQA 中包括了训练集 41 522 条数据,验证集 4 396 条数据、未标注的测试集 4 086 条数据。本文在测试集数据上进行了标注,并将所有数据集的标注信息进行了拓展,即添加了问题条件值。最终使用的数据集包括 39 594 条训练集数据、4 119 条验证集数据、4 026 条测试集数据。

在编码器上使用的是参数量约为 3.3 亿个的中文 RoBERTa 预训练模型,因此在模型训练时需要很大的显存。本文使用了带有单个 24 GB 显存的 RTX3090 显卡、至强 4112 处理器、32 GB 内存的台式机进行模型训练及测试,在此设备的基础上,训练的批次大小已经不支持 32 了,最终选用了 16 作为批次大小。

本文使用逻辑形式准确率(Logical Form,LF)和执行结果准确率(Execution,Exec)作为模型性能的评估标准。逻辑形式准确率指的是生成的 SQL 语句与标注的 SQL 语句比对的准确率,WHERE 下子句的顺序并不会影响 SQL 语句的逻辑形式比较结果。执行结果准确率指的是生成 SQL 语句与标注 SQL 语句在对应表上的执行结果比对的准确率。

3.2 总体模型与子任务实验

本文方法以及 SQLNet[4]、Coarse2Fine[19]、SQLova[7]、X-SQL[9]和 M-SQL[12]在 TableQA 上的实验结果如表 1 所示,表 1 中加粗数字表示最优值。本文方法的逻辑形式准确率无论在验证集还是测试集上都取得了最好的成绩。与 M-SQL[12]相比,本文方法在测试集上的逻辑形式准确率和执行结果准确率分别得到了 1.7 百分点和 1.8 百分点的提升,并且本文方法的子任务网络更加简单。本文认为这些提升主要来源于以下几点:(1) 使用了 RoBERTa 编码器并且在输入中融合了字段类型信息,直接通过列类型分隔符来提取字段特征,更高效地利用了语言模型的学习能力;(2) 添加了问句条件值标注,分离了问句结构的学习和文本匹配,使子任务网络的学习更加高效;(3) 在文本匹配中结合了编辑距离与语义词典的方法,对 TableQA 中较难的 W-VAL 任务提供了很好的帮助。

表 1 TableQA 数据集上的测试结果(%)

模型	Dev LF	Dev Exec	Test LF	Test Exec
SQLNet	63.2	69.7	61.4	67.2
Coarse2Fine	73.0	76.9	72.6	76.7
SQLova	80.6	86.7	81.7	85.9

续表 1

模型	Dev LF	Dev Exec	Test LF	Test Exec
X-SQL	82.9	87.0	83.3	87.6
M-SQL	86.9	**91.8**	88.8	91.6
本文方法	**88.4**	91.3	**90.5**	**93.4**

本文方法和其他模型在 TableQA 测试集上各子任务的准确率如表 2 所示,加粗数字表示最优结果。本文方法在大多数子任务上的性能都比其他模型好,从整体的逻辑形式准确率上在最好的模型的基础上提升了 1.7 百分点。

表 2　TableQA 测试集上各个子任务测试结果(%)

模型	S-NUM	S-COL	S-AGG	W-NUM	W-CON
文献[11]	99.4	91.5	97.9	93.4	97.1
M-SQL	99.5	97.5	98.9	97.5	—
本文方法	**99.8**	**98.5**	**99.3**	**97.8**	**98.2**
模型	W-COL	W-OP	W-VAL	LF	
文献[11]	89.3	97.1	84.1	87.6	
M-SQL	**98.2**	99.1	**97.0**	88.8	
本文方法	97.8	**99.3**	96.0	**90.5**	

为了更为直观地证明本文方法在中文 NL2SQL 任务上的有效性,使用本文方法对多种结构的问句进行解析,问句以及解析生成的 SQL 语句展示在表 3 中,生成的 SQL 语句对表名作了截断处理。从解析结果中可以看出,本文方法多种针对单表的自然语言问句可以正确地解析,在问句中不直接包含真实条件值时,可以通过文本匹配方法得到真实条件值,以保证生成的 SQL 语句能够正确执行。

表 3　对多种结构问句的解析结果

自然语言问句	解析生成的 SQL 语句
哪些城市的本月周均成交面积与去年同期均成交面积都是小于 10 万平的?	SELECT (城市) FROM Table_c98a1b4 WHERE 本月周均成交面积 < 10 AND 去年同期周均成交面积 < 10
我想你帮我查一下大黄蜂还有密室逃脱这两部电影票房的占比加起来会是多少来着?	SELECT SUM(票房占比) FROM Table_4d39d05 WHERE 影片名称 = '大黄蜂' OR 影片名称 = '密室逃脱'
国航和南航 2017 年一共引进客机多少架呀?	SELECT SUM(2017 年) FROM Table_01556f3 WHERE 公司简称 = '中国国航' OR 公司简称 = '南方航空'
有几家公司的股票收盘价是破二十元的啊?	SELECT COUNT(名称) FROM Table_69cdfd8 WHERE 收盘价 > 20

续表 3

自然语言问句	解析生成的 SQL 语句
万科的湖心岛和华润的橡树湾这两个楼盘的最高卖价是多少钱一平呀?	SELECT MAX(售价) FROM Table_5a4f2fa WHERE 项目 = '万科湖心岛' OR 项目 = '华润橡树湾'
哪些股的滚动市盈率是高于 10.65 的,再告诉我他们分别对应的代码。	SELECT(股票代码),(股票简称) FROM Table_43afe8f WHERE PE-TTM > 10.65

针对几种 WikiSQL 数据集的方法,本文对其子任务 WHERE 子句条件值预测结果使用文本匹配方法做进一步处理来得到最终的条件值。原方法与加入文本匹配的方法在 WikiSQL 数据集上的测试结果如表 2 所示,TM 表示文本匹配(Text Matching),括号内表示相比原模型的提升。可以发现文本匹配方法对 SQLNet[4]、SQLova[7]、HydraNet[10] 的性能均有一定的改进。

表 4　添加文本匹配方法前后在 WikiSQL 上的测试结果(%)

模型	Dev LF	Dev Exec	Test LF	Test Exec
SQLNet	63.16	69.65	61.35	67.87
SQLNet + TM	66.45 (+3.29)	74.60 (+4.95)	64.30 (+2.95)	72.65 (+4.78)
SQLova	80.59	86.68	79.67	85.86
SQLova + TM	81.64 (+1.05)	88.47 (+1.79)	80.53 (+0.86)	87.70 (+1.84)
HydraNet	83.65	89.13	83.81	89.17
HydraNet + TM	84.67 (+1.02)	90.04 (+0.91)	84.55 (+0.75)	90.19 (+1.02)

本文也对几种常见的字符串匹配方法进行了测试,测试的数据就是标注的问题条件值、真实条件值、对应的列值。测试方法是将问题条件值与一组列值进行匹配,测试指标包括匹配出的真实条件值准确率和匹配时间。结果如表 5 所示,加粗数字表示最优结果。可以看出,在处理 TableQA 中的文本匹配时,神经网络的方法(Synonyms)效果并不是很好,几种编辑距离的方法无论在匹配时间还是准确率上都有较大的优势,本文使用的文本匹配方法结合了词典与编辑距离,在时间上略微比单纯编辑距离方法长一些,但在准确率上提升了 3.8 百分点。

表 5　文本匹配方法在 TableQA 数据上的准确率和消耗时间

方法	准确率/%	时间/s
Levenshtein Distance	88.8	**1.72**
Levenshtein Ratio	90.6	1.97
Jaro Distance	85.5	1.86
Jaro-Winkler Distance	80.3	1.82
Synonyms	49.6	20 425.68
本文方法	**94.4**	2.56

3.3 消融实验

为了观察本文方法各个部分的重要性,本文进行了消融实验,结果如表 6 所示。

表 6 消融实验结果(%)

模型	LF	Exec
本文方法	90.53	93.44
- RoBERTa-wwm-ext-large + BERT-wwm-ext	89.67	92.35
- RoBERTa-wwm-ext-large + BERT-wwm	88.40	91.16
- [TYPE] + [COS]	87.68	90.39
- 文本匹配	79.68	85.86
- 条件值标注	87.55	90.68

在编码器方面,本文尝试使用不同的预训练模型。从测试结果中可以看出,即使使用 BERT-wwm-ext 作为预训练模型,本文方法效果依旧比其他方法要好。使用改进的 RoBERTa 语言模型比使用 BERT 的效果更好。相比使用 BERT-wwm-ext 作为预训练模型,使用 RoBERTa-wwm-ext-large 在测试集上分别提升了 0.86 百分点和 1.09 百分点的逻辑形式准确率和执行准确率。

在输入信息上,本文将编码输入的列类型分割符统一换成[COS],使其既不根据列字段的类型改变也不根据分割的三个位置改变。由结果可以看出,使用融合列字段类型信息的[TYPE]进行提取特征对整体在测试集贡献了 2.85 百分点、2.28 百分点的逻辑形式准确率和执行准确率。

为了验证文本匹配方法的必要性,本文在子任务 W-VAL-S 后不使用文本匹配方法做进一步处理,直接在生成 SQL 语句中进行使用。从结果中可以看出文本匹配方法对于整体在测试集贡献了 10.85 百分点和 7.58 百分点的逻辑形式准确率和执行准确率,一方面这取决于数据集中问题不直接包含 SQL 语句所需条件值的比例,另一方面,文本匹配方法在测试集上提供了更大的提升,也体现加入文本匹配方法能够让训练的模型更具有鲁棒性,更有利于应用。

为了验证问题条件值标注的有效性,本文使用未进行问题条件值标注的数据集进行模型训练,使用程序依据真实条件值从问句中查找起止位置的方式来得到条件值的起止位置以计算问句条件值的损失。从测试结果中可以看出问题条件值标注相对程序查找对整体在测试集贡献了 2.98 百分点和 2.76 百分点的逻辑形式准确率和执行准确率。

4 结 语

为了提高中文自然语言转 SQL 语句的解析质量,本文提出一种融合字段类型与文本匹配的中文问句解析方法。使用了更为优秀的 RoBERTa 语言模型,通过在输入中融合列类型信息作为分隔符更充分地利用表格结构,并且不需要在编码器输出后再额外添加其他计算来提取列特征。针对问句可能并不直接包含条件值的问题,通过增加问题条件值的标注信息来让子任务学习更加准确,并与后续的文本匹配分隔开来。使用了结合词典与编辑距离的文本匹配方法,对条件值预测子任务以及整体任务有了一定的提升效果。通过在中英文单表数据集上的一些测试对比,本文提出的整体模型在 TableQA 数据集上取得了最高的逻辑形式准确率和执行准确率,结合了文本匹配方法对英文数据集的处理方法也取得了很大的提高。然而本文的问句解析方法只能针对单表普通问句进行处理,后续可以针对时序类型问句以及涉及多表信息问句对模型作进一步研究,以扩展关系型数据库的自然语言接口的应用广度。

参 考 文 献

[1] Giordani A, Moschitti A. Automatic generation and reranking of SQL-derived answers to NL questions[C]//International Workshop on Eternal Systems, 2012: 59 - 76.

[2] Zhong V, Xiong C M, Socher R. Seq2SQL: Generating structured queries from natural language using reinforcement learning[EB]. arXiv: 1709.00103, 2017.

[3] Dong L, Lapata M. Language to logical form with neural attention[C]//54th Annual Meeting of the Association for Computational Linguistics, 2016: 33 - 43.

[4] Xu X J, Liu C, Song D W. SQLNet: Generating structured queries from natural language without reinforcement learning [EB]. arXiv: 1711.04436, 2017.

[5] Liu X D, He P C, Chen W Z, et al. Multi-Task deep neural networks for natural language understanding[C]//57th Annual Meeting of the Association for Computational Linguistics, 2019: 4487 - 4496.

[6] Liu Y H, Ott M, Goyal N, et al. Roberta: A robustly optimized BERT pretraining approach[EB]. arXiv: 1907.11692, 2019.

[7] Hwang W, Yim J, Park S, et al. A comprehensive exploration on WikiSQL with table-aware word contextualization [EB]. arXiv: 1902.01069, 2019.

[8] Devlin J, Chang M W, Lee K, et al. BERT: Pre-training of deep bidirectional transformers for language understanding

[C]//Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies,2019:4171 – 4186.

[9] He P C, Mao Y, Chakrabarti K, et al. X-SQL: Reinforce schema representation with context[EB]. arXiv:1908.08113, 2019.

[10] Lyu Q, Chakrabarti K, Hathi S, et al. Hybrid ranking network for text-to-SQL[EB]. arXiv:2008.04759,2020.

[11] 袁志祥,任冬冬,洪旭东,等. 结合数据库结构及内容的问句理解方法研究[J]. 计算机工程,2020,47(3):71 – 79.

[12] 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.

[13] Cui Y M, Che W X, Liu T, et al. Pre-training with whole word masking for Chinese BERT[EB]. arXiv:1906.08101v1, 2019.

[14] Levenshtein V. Binary codes capable of correcting deletions, insertions, and reversals[J]. Soviet Physics Doklady,1965, 163(4):845 – 848.

[15] Jaro M A. Advances in record-linkage methodology as applied to matching the 1985 census of Tampa, Florida[J]. Journal of the American Statistical Association, 1989, 84 (406):414 – 420.

[16] Winkler W E. The state of record linkage and current research problems[C]//Statistical Research Division,1999.

[17] Le Q, Mikolov T. Distributed representations of sentences and documents[C]//31st International Conference on Machine Learning,2014:1188 – 1196.

[18] Lastra-Díaz J, Goikoetxea J, Taieb M A, et al. A reproducible survey on word embeddings and ontology-based methods for word similarity: Linear combinations outperform the state of the art[J]. Engineering Applications of Artificial Intelligence,2019,85:645 – 665.

[19] Dong L, Lapata M. Coarse-to-fine decoding for neural semantic parsing[C]//56th Annual Meeting of the Association for Computational Linguistics,2018:731 – 742.

(上接第 158 页)

[6] Asim M, Ming Z, Javed M Y. CNN based spatio-temporal feature extraction for face anti-spoofing[C]//2nd International Conference on Image, Vision and Computing,2017: 234 – 238.

[7] 郝枢华. 基于多特征融合的假冒人脸检测算法[D]. 北京:华北电力大学,2019.

[8] 王志斌,肖艳姣,吴涛,等. 改进变分光流法并行算法实现[J]. 计算机应用与软件,2019,36(1):105 – 110.

[9] Wadhwa N, Rubinstein M, Durand F, et al. Phase-based video motion processing[J]. ACM Transactions on Graphics, 2013,32(4):1 – 10.

[10] 张芳芳,李楠. 基于独立成分分析的 fMRI 数据分类[J]. 计算机应用与软件,2019,36(11):107 – 111.

[11] Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition[EB]. arXiv:1409.1556,2015.

[12] Chingovska I, Anjos A, Marcel S. On the effectiveness of local binary patterns in face anti-spoofing[C]//IEEE International Conference of the Biometrics Special Interest Group, 2012:1 – 7.

[13] Zhang Z W, Yan J, Liu S F, et al. A face anti-spoofing database with diverse attacks[C]//5th IAPR International Conference on Biometrics,2012:26 – 31.

[14] Shafer S A. Using color to separate reflection components [J]. Color Research & Application, 2010, 10(4):210 – 218.

[15] Yu H, Ng T, Sun Q B. Recaptured photo detection using specularity distribution[C]//15th IEEE International Conference on Image Processing,2008:3140 – 3143.

[16] Simoncelli E P, Freeman W T. The steerable pyramid: A flexible architecture for multi-scale derivative computation [C]//International Conference on Image Processing,1995: 444 – 447.

[17] Gupta P, Bhowmick B, Pal A. Accurate heart-rate estimation from face videos using quality-based fusion[C]//IEEE International Conference on Image Processing,2017:4132 – 4136.

[18] Lucas B D, Kanade T. An iterative image registration technique with an application to stereo vision[C]//7th International Joint Conference on Artificial Intelligence,1981:674 – 679.

[19] Hu J, Shen L, Sun G. Squeeze-and-excitation networks[C]// IEEE Conference on Computer Vision and Pattern Recognition,2018:7132 – 7141.

[20] 付晓静. 基于头部微弱运动的心率信号检测方法研究[D]. 天津:河北工业大学.

[21] Patel K, Han H, Jain A K. Secure face unlock: Spoof detection on smartphones[J]. IEEE Transactions on Information Forensics & Security,2016,11(10):2268 – 2283.

[22] 郑欣洋. 面向人脸认证的活体检测方法研究[D]. 南京:东南大学,2018.

[23] Rehman Y A U, Po L M, Liu M Y. LiveNet: Improving features generalization for face liveness detection using convolution neural networks[J]. Expert Systems with Applications, 2018,108:159 – 169

[24] 廖迪,黄奥运,李科,等. 基于图像色彩纹理的人脸活体检测算法研究[J]. 现代计算机,2019(18):59 – 63.

[25] Wen C, Chen F M, Xie K, et al. Face liveness detection: Fusing colour texture feature and deep feature[J]. IET Biometrics,2019,8(6):369 – 377.