基于预训练机制的自修正复杂语义分析方法

李青[1]，钟将[1]，李立力[2]，李琪[3]
（1. 重庆大学计算机学院，重庆 400044；2. 重庆大学土木工程学院，重庆 400044；
3. 绍兴文理学院计算机科学与工程系，浙江 绍兴 312000）

摘 要：面向知识服务过程中内容资源的智能化、知识化、精细化和重组化的碎片性管理需求。深层分析并挖掘语义隐层知识、技术、经验与信息，突破已有传统文本到结构化查询语言（SQL）的语义分析技术瓶颈，提出基于预训练机制的自修正复杂语义分析方法 PT-Sem2SQL。设计结合 Kullback-Leibler 差异技术的 MT-DNN 预训练机制，以加强上下文语义理解深度；设计专有增强模块，捕获句内上下文语义信息的位置；并通过自修正方法优化生成模型的执行过程，以解决解码过程中的错误输出。实验结果表明，PT-Sem2SQL 能够有效提高复杂语义的解析性能，准确度优于相关工作。

关键词：文本到 SQL；语义分析；自然语言处理；复杂事件处理
中图分类号：TP302.1
文献标识码：A
doi: 10.11959/j.issn.1000−436x.2019195

Self-correcting complex semantic analysis method based on pre-training mechanism

LI Qing[1], ZHONG Jiang[1], LI Lili[2], LI Qi[3]
1. College of Computer Science, Chongqing University, Chongqing 400044, China
2. School of Civil Engineering, Chongqing University, Chongqing 400044, China
3. Department of Computer Science and Engineering, Shaoxing University, Shaoxing 312000, China

Abstract: In the process of knowledge service, in order to meet the fragmentation management needs of intellectualization, knowledge ability, refinement and reorganization content resources. Through deep analysis and mining of semantic hidden knowledge, technology, experience, and information, it broke through the existing bottleneck of traditional semantic parsing technology from Text-to-SQL. The PT-Sem2SQL based on the pre-training mechanism was proposed. The MT-DNN pre-training model mechanism combining Kullback-Leibler technology was designed to enhance the depth of context semantic understanding. A proprietary enhancement module was designed that captured the location of contextual semantic information within the sentence. Optimize the execution process of the generated model by the self-correcting method to solve the error output during decoding. The experimental results show that PT-Sem2SQL can effectively improve the parsing performance of complex semantics, and its accuracy is better than related work.

Key words: Text-to-SQL, semantic parsing, natural language processing, complex event processing

---

收稿日期：2019−04−24；**修回日期**：2019−10−30
通信作者：钟将，zhongjiang@cqu.edu.cn
基金项目：中央高校研究生科研创新基金资助项目（No.2018CDYJSY0055）；国家重点研发计划基金资助项目（No.2017YFB1402400）；重庆市研究生科研创新基金资助项目（No.CYB18058）；重庆市技术创新与应用示范基金资助项目（No.cstc2018jszx-cyzdX0086）

Foundation Items: Fundamental Research Funds for the Central Universities (No.2018CDYJSY0055), The National Key Research and Development Program of China (No.2017YFB1402400), Graduate Research and Innovation Foundation of Chongqing (No.CYB18058), Chongqing Technological Innovation and Application Demonstration Project (No.cstc2018jszx-cyzdX0086)

1 引言

随着现代知识服务业的发展，海量跨领域知识信息封装存储于关系数据库中，面向内容资源的知识信息存在严重过载现象。如何创新现代服务科学，攻克关键核心技术，重塑现代知识服务业技术体系和价值链，提高内容资源在现代知识服务业增加值中的贡献度，创新发展现代知识服务新生态已成为研究热点与难点。针对内容资源的智能化、知识化、精细化和重组化的碎片性管理需求，建立新技术范式下的复杂语义分析方法成为重要的研究目标。

复杂语义分析的任务是将人类自然语言转换为对应的结构化查询语言（SQL, structured query language），即 Text-to-SQL。如何高效地表达隐层知识、技术、经验与信息，则是复杂语义分析领域的研究热点与难点。同时，复杂语义分析是自然语言处理中重要的子任务之一，可为智能问答[1-3]、机器翻译[4]和复杂事件处理[5-6]等重要应用提供理论基础。因此，本文重点关注如何将自然语言映射到结构化查询语言 SQL 语句。

一直以来，复杂语义分析模型因缺乏高标准的标注数据集而难以训练发展。2018 年，来自耶鲁大学的 Yu 等[7]成功构造了第一个具有复杂跨领域文本到 SQL 的标记数据集——Spider。2019 年，Yu 等[8]创新性地构造具有连贯查询的另一大型复杂跨领域文本到 SQL 的标记数据集——SParC。Spider 数据集中语义解析任务的示例如图 1 所示。在此之前，几乎所有传统数据集（WikiSQL[9]、ATIS[10-11]、GeoQuery[12]）都仅关注简单的 SQL 查询，进而导致训练模型仅满足匹配语义解析结果的需求，无法真正理解自然语言的含义[12]。

鉴于以上分析，本文提出自然语言查询的形式化语义表示模型——PT-Sem2SQL（pretraining semantic parsing to SQL）。为证明在面向真实内容资源中模型是有效的，采用 Spider 数据集和 SParC 数据集进行测试。本文模型构建思路是在以 BERT[13]为主干的 MT-DNN[14-15]预训练技术基础上，结合 KL（Kullback-Leibler）差异技术[16]设计预训练模块。同时，为了捕获顺序信息满足复杂的 SQL 查询，提出带有多个子句和附加句内上下文信息的增强模块。最后，采用自修正学习优化的思想生成优化模型的执行过程，解决解码过程中的错误输出。

综上所述，本文的主要贡献如下。

首先，本文设计结合 KL 差异技术增加[Zero]列的 MT-DNN 预训练模块，构建结合多任务学习与标记数据的 PT-Sem2SQL 模型。这部分模块的建立可有效解决文本到 SQL 任务的零列[WHERE]子句的预测挑战，满足复杂跨领域文本到 SQL 的数据集查询任务要求。

其次，本文提出增加额外的增强模块来捕获句内上下文语义信息。通过增强模块，子任务可采用细粒度语义分析方式进行刻画，同时底层子任务的构建可为上层任务表示提供基础。通过实验验证，句内上下文语义信息对于结构化数据的语言任务

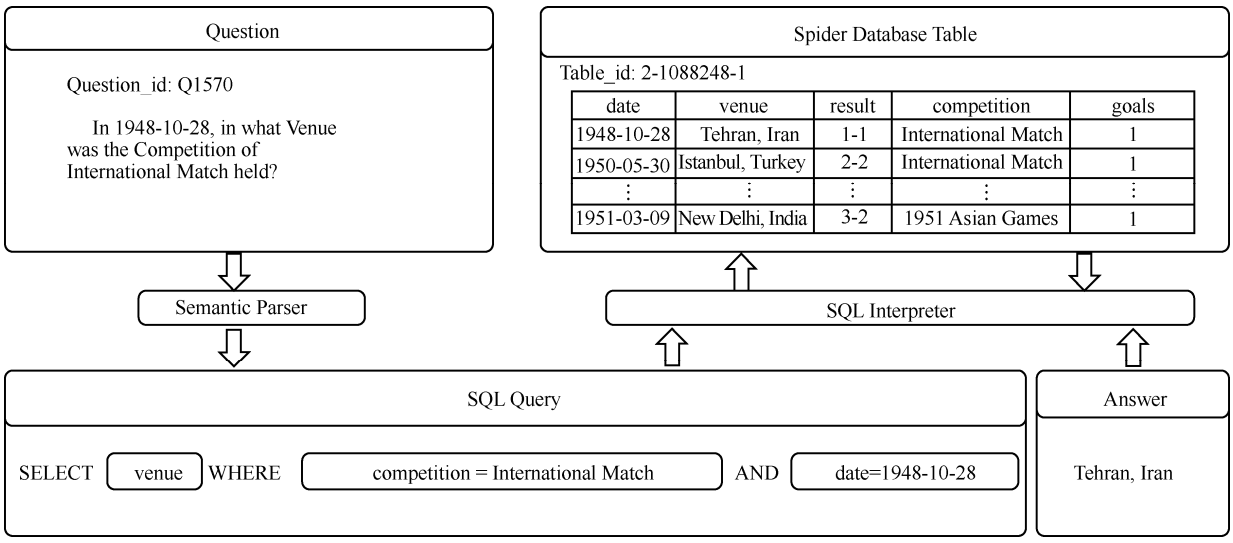


图 1 Spider 数据集中语义解析任务的示例

同样起到至关重要的作用。

最后，针对解码过程中的错误输出问题，PT-Sem2SQL 模型构建自修正方法优化生成模型的执行过程。将通过模型产生伪 Text-to-SQL 查询与真实的 Text-to-SQL 查询视为一对修正任务，采用自修正的思想优化真伪数据间差距，进而达到自修正优化的目的。

2　相关工作

针对 Text-to-SQL 语义解析问题的研究已经持续几十年。同时，相关领域专家已经提出各种对应的语义解析器模型[17-22]。

早期研究重点是将 Text-to-SQL 任务映射为序列生成的建模问题，主要构建具有自注意机制的神经网络序列到序列模型[23-24]。虽然此类方法仅取得初步效果，但无法确保生成语法的有效输出。其中，以 TypeSQL 为代表，它是 2018 年由 Yu 等[25]提出的依存语法分析模型，可以满足基础查询语言需求独立生成目标 SQL 查询的 SELECT 和 WHERE 子句。同年，Dong 等[26]提出 Coarse2Fine 模型，通过优先输出一组草图，采用插槽填充的方法优化子句解码的结果。另一种 Pointer-SQL 模型则展现出新思路，提出 sequence-to-action 的方法，该方法使用基于注意机制的复制方法和基于值构造丢失函数的方法[27]。通过构造具有注释功能的 seq2seq 模型，试图确保模型在解码过程中各个阶段语法的正确性。

尽管上述以自然语言为基础的语义解析器模型成功地解决了简单语义到形式化 SQL 查询语句的问题，但因简单语义的单一性缺陷问题导致难以扩展，无法生成复杂的 SQL 查询语句。同时，此类仅在传统的 WikiSQL 数据集[9]上训练的模型难以捕获各种自然语言变体。2018 年，耶鲁大学的 Yu 等成功开发 Spider 数据集[7]，囊括困难层面的 SQL 查询（同时包含 2 个以上的[SELECT]、[WHERE]和[GROUP BY]子句）。2019 年，Yu 等为弥补 Spider 数据集中未关注上下文语境信息的不足，进一步地开发出大型上下文相关跨领域 SParC 数据集。它包含 138 个领域、具有复杂上下文依存关系、囊括复杂语义多样性的 Text-to-SQL 数据集。本文将 Spider 数据集与 SParC 数据集进行对比，如表 1 所示。

由此可见，解决复杂跨领域的数据集上 Text-to-SQL 语义解析问题，需要模型训练生成复杂的文本到 SQL 查询。同时，此类任务也更类似于自然场景下的查询。2019 年提出的预处理技术，极大地增强了以词表示为主的外部语料库（如 Glove 模型[28]）。受到此类预处理技术的发展启发，Hwang 等[29]针对文本到 SQL 查询建立新型预训练的 BERT 模型。此外，一些工作也同样证明预训练外部语料库的模型在文本到 SQL 任务中具有显著改进的价值[13-14,30]。最优预训练技术 MT-DNN[14]则更显著地体现了这一优势，成功用多任务学习将 2 类语料库（标注和未标注的语料库）进行深度融合。由此可知，未标注语料库的训练可有效增强模型的通用性。

3　PT-Sem2SQL 模型

3.1　模型概述

本文将模型的总体结构划分为 4 个描述模块深入解构 PT-Sem2SQL 模型，即编码模块、增强模块、输出模块和自修正模块。PT-Sem2SQL 模型的整体结构与基础模块如图 2 所示。

3.2　编码模块

为了使 PT-Sem2SQL 模型更适合于复杂跨领域查询任务，模型重新设计 MT-DNN 预处理模块，增加[Zero]和[CON-TI]增强句内语义信息。编码模块主要为以下 3 个部分，如图 3 所示。

1) 句内语义信息部分（[CON-TI]）

本文设计[CON-TI]来捕获句内上下文语义信息的位置。同时，每个构造的 Token 包括 3 个部分：Token embedding（E_T）、Type embedding（E_Y）和 Position embedding（E_P）。

表 1　**Spider 与 SParC 数据集对比**

数据集	CONTEXT	CROSS DOMAIN	QUESTION /个	TURN	QUESTION LENGTH	QUESTION VOCABULARY	SELECT	WHERE	GROUP BY	ORDER BY	KEYWORDS
Spider	NO	YES	11 840	1.0	13.4	4 818	98.9%	55.2%	24.0%	21.5%	68.6%
SParC	YES	YES	12 726	3.0	8.1	3 794	97.4%	42.8%	20.1%	17.0%	41.2%

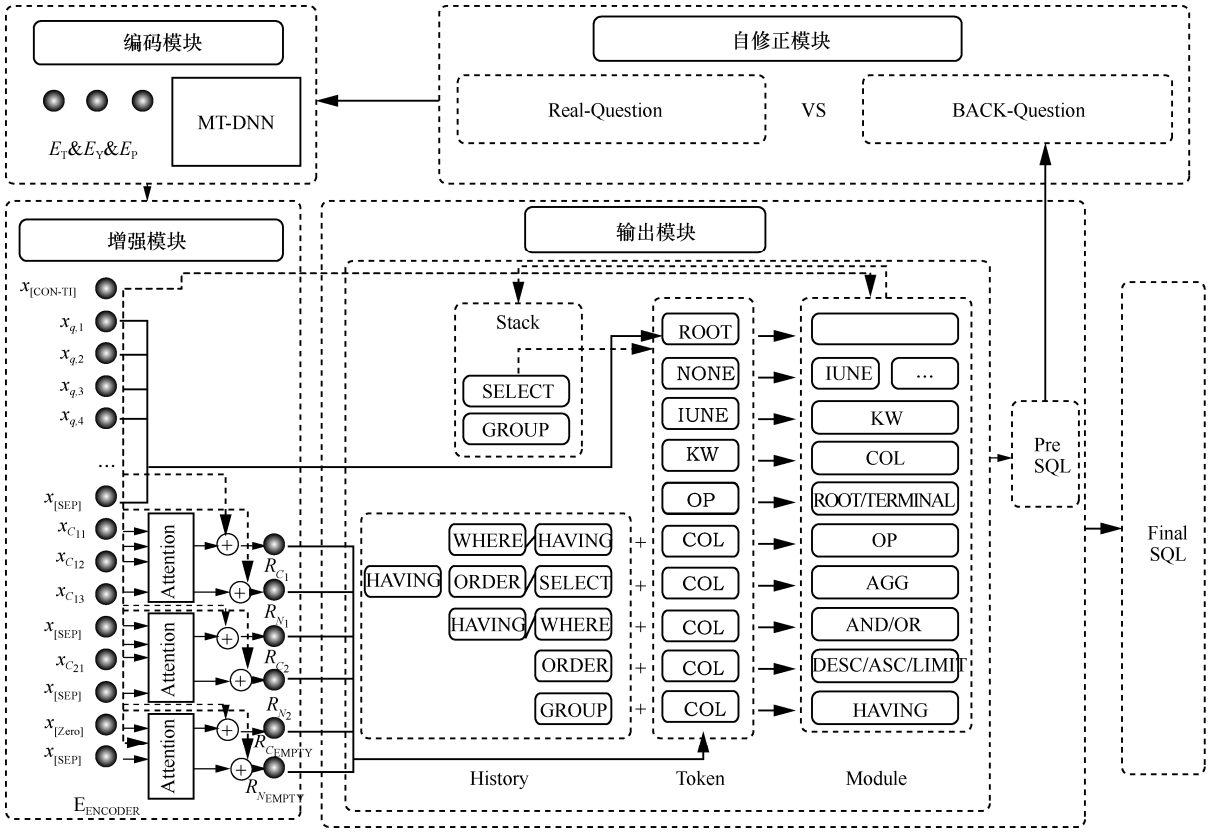


图 2　PT-Sem2SQL 模型的整体结构与基础模块

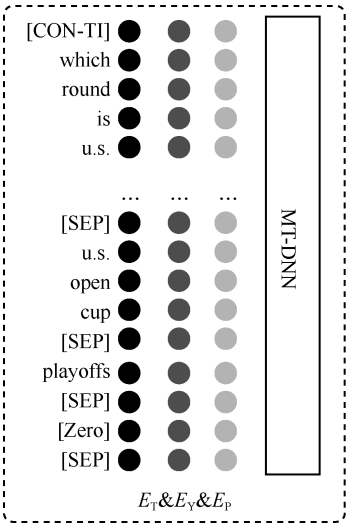


图 3　PT-Sem2SQL 模型的编码模块

2) 零列部分（[Zero]）

本文扩展原有用于编码自然语言查询的预训练模型，添加表头列表。同时采用[SEP]将查询子句与表头列表分离，在每个列表模式中增加[Zero]部分。同时，本文也对 KW-COL 的交叉熵损失函数进行优化，重新定义 KL 值，使其 KL 值介于 $D(Q\,|\,P^{\text{KW-COL}})$ 之中，即

$$\begin{cases} Q_{[\text{Zero}]}=1,\ \ n_{[\text{KW}]}=\text{NULL} \\ Q_{[\text{KW-COL}]}=\dfrac{1}{n},\ \ n_{[\text{KW}]}\geqslant 1 \end{cases} \tag{1}$$

3) MT-DNN 初始化部分

将编码器与 MT-DNN 预训练技术共同使用，使其编码器具有多个自然语言查询任务的功能。最终为模型对齐的有效查询提供支持。

3.3　增强模块

利用 PT-Sem2SQL 模型的编码模块输出用于编码的向量，即

$$\boldsymbol{x}_{[\text{CON-TI}]},\boldsymbol{x}_{q,1},\cdots,\boldsymbol{x}_{q,n},\boldsymbol{x}_{[\text{SEP}]},\boldsymbol{x}_{C_{11}},\cdots,\boldsymbol{x}_{[\text{SEP}]},$$

$$\boldsymbol{x}_{C_{21}},\cdots,\boldsymbol{x}_{[\text{SEP}]},\boldsymbol{x}_{[\text{Zero}]},\boldsymbol{x}_{[\text{SEP}]}$$

其中，$\boldsymbol{x}_{[\text{CON-TI}]}$ 代表捕获的句内上下文语义信息；每个查询句输入为 $\boldsymbol{x}_{q,1},\cdots,\boldsymbol{x}_{q,n}$（$n$ 是查询字的数量）；$\boldsymbol{x}_{C_{ij}}$ 是第 i 列第 j 个 Token 的输出；R_{C_i} 表示增强模块的输出。所有向量都属于 $\mathbb{R}^d$，d 是 MT-DNN 编码器的隐藏维度（在大型 MT-DNN 模型中 d=1 024）。

由于在编码模块中各部分捕获上下文影响力不够强，本文建立增强模块以加强句内的上下文影响。增强模块使用 $\boldsymbol{x}_{[\text{CON-TI}]}$ 来更新架构内各个部分，同时与输出模块不同模块相互对齐连接。用 softmax 分类器将对齐模型分类，如式(2)所示。

$$\alpha_{ij} = \text{softmax}(W_{[\text{CON-TI}]}\boldsymbol{x}_{[\text{CON-TI}]}W_{C_{ij}}\boldsymbol{x}_{C_{ij}}) \quad (2)$$

其中，α_{ij} 为匹配全局上下文的列的第 j 个标记的输出；$W_{[\text{CON-TI}]}, W_{C_{ij}} \in \mathbb{R}^d$。总结每列结果，采用式(3)计算增强模块。

$$R_{C_i} = \boldsymbol{x}_{[\text{CON-TI}]} + \boldsymbol{x}_{C_i} = \boldsymbol{x}_{[\text{CON-TI}]} + \sum_{j=1}^{n_i}\alpha_{ij}\boldsymbol{x}_{C_{ij}} \quad (3)$$

本文采用与式(2)和式(3)相类似的方式，统计增强模块预测关键字数。为了防止出现嵌套模块，本文添加终端模块的预测，具体将在第 3.4 节中详述。

$$\beta_{ij} = \text{softmax}(W_{[\text{CON-TI}]}\boldsymbol{x}_{[\text{CON-TI}]}W_{N_{ij}}\boldsymbol{x}_{N_{ij}}) \quad (4)$$

$$R_{N_i} = \boldsymbol{x}_{[\text{CON-TI}]} + \boldsymbol{x}_{N_i} = \boldsymbol{x}_{[\text{CON-TI}]} + \sum_{j=1}^{n_i}\beta_{ij}\boldsymbol{x}_{N_{ij}} \quad (5)$$

其中，$W_{N_{ij}} \in \mathbb{R}^d$，$\boldsymbol{x}_{N_{ij}}$ 表示第 i 列第 j 个 Token 的输出，R_{N_i} 表示 Token 句内上下文关键字数的增强输出。

3.4 输出模块

本文引入 Yu 等[31]在 Text-to-SQL 任务的分解方式，将其分解为 9 个子模块。各模块都预测最终 SQL 查询语句的一部分。但是，与 Yu 等所提 SyntaxSQLNet 模型不同的是，本文的 PT-Sem2SQL 模型定义编码模块和增强模块，各个模块的计算方式也相应产生变化。同时，也注意到使用一个堆栈来运行本文的解码过程，直到其置空。

1) $ IUEN 子模块

其关键字选自{INTERSECT, UNION, EXCEPT, NONE}中及概率计算式，为

$$P_{\text{IUEN}}(C_i) = \text{softmax}(W_{\text{IUEN}}R_{C_i}) \quad (6)$$

2) $ KW 子模块

为结合复杂跨领域查询的特性，PT-Sem2SQL 模型需要先预测 SQL 查询语句中的关键字数，并在 3 种可能的关键词中进行选择{WHERE, GROUP BY, ORDER BY}。

$$P_{\text{KW}}^{\text{num}}(N_i) = \text{softmax}(W_{\text{KW}}^{\text{num}}R_{N_i}) \quad (7)$$

$$P_{\text{KW}}^{\text{val}}(C_i) = \text{softmax}(W_{\text{KW}}^{\text{val}}R_{C_i}) \quad (8)$$

3) $ OP 子模块

子模块关键字选自{=, >, <, >=, <=, !=, LIKE, NOT IN, IN, BETWEEN}。$ OP 子模块同样需要先预测关键字数，其概率计算式为

$$P_{\text{OP}}^{\text{num}}(N_i) = \text{softmax}(W_{\text{OP}}^{\text{num}}R_{N_i}) \quad (9)$$

$$P_{\text{OP}}^{\text{val}}(C_i) = \text{softmax}(W_{\text{OP}}^{\text{val}}R_{C_i}) \quad (10)$$

4) $ AGG 子模块

子模块关键字选自{MAX, MIN, SUM, COUNT, AVG, NONE}。同样，它取决于聚合器的数量，计算式为

$$P_{\text{AGG}}^{\text{num}}(N_i) = \text{softmax}(W_{\text{AGG}}^{\text{num}}R_{N_i}) \quad (11)$$

$$P_{\text{AGG}}^{\text{val}}(C_i) = \text{softmax}(W_{\text{AGG}}^{\text{val}}R_{C_i}) \quad (12)$$

5) $ COL 子模块

采用$ COL 子模块来预测表中各列，其概率计算式为

$$P_{\text{COL}}^{\text{num}}(N_i) = \text{softmax}(W_{\text{COL}}^{\text{num}}R_{N_i}) \quad (13)$$

$$P_{\text{COL}}^{\text{val}}(C_i) = \text{softmax}(W_{\text{COL}}^{\text{val}}R_{C_i}) \quad (14)$$

6) $ ROOT/TERMINAL 子模块

为了结合复杂跨领域查询的特性，本文添加预测跨度以方便预测是否有新的子模块。这种方法能有效预测跨度的开始和结束位置。同时，模型需要首先调用$ OP 子模块，然后确定它何时是$ ROOT 子模块。

$$P_{\text{R/T}}^{\text{begin}}(C_i) = \text{softmax}(W^{\text{begin}}x_{q_i} + W_{C_i}^{\text{begin}}R_{C_i}) \quad (15)$$

$$P_{\text{R/T}}^{\text{end}}(C_i) = \text{softmax}(W^{\text{end}}x_{q_i} + W_{C_i}^{\text{end}}R_{C_i}) \quad (16)$$

其中，$W^{\text{begin}}, W_{C_i}^{\text{begin}}, W^{\text{end}}, W_{C_i}^{\text{end}} \in \mathbb{R}^d$。

7) $ AND/OR 子模块

子模块关键字选自{AND, OR}中，其概率计算式为

$$P_{\text{AND/OR}}(C_i) = \text{softmax}(W_{\text{AND/OR}}R_{C_i}) \quad (17)$$

8) $ DESC/ASC/LIMIT 子模块

同样，模块选自母模块 ORDER BY 下的{DESC, ASC, DESC LIMIT, ASC LIMIT}中，概率计算式为

$$P_{\text{D/A/L}}(C_i) = \text{softmax}(W_{\text{D/A/L}}R_{C_i}) \quad (18)$$

9) $ HAVING 子模块

模块选自母模块 GROUP BY 下的{HAVING}，概率计算式为

$$P_{\text{HAVING}}(C_i)=\text{softmax}(W_{\text{HAVING}}R_{C_i}) \tag{19}$$

3.5 自修正模块

将输出模块生成的 SQL 查询语句反向生成伪查询问题（back-question），传入自修正模块。通过伪查询与真实的查询问题（real-question）间进行二元极大博弈，以达到模型自修正的目的。自修正模块解析如图 4 所示。

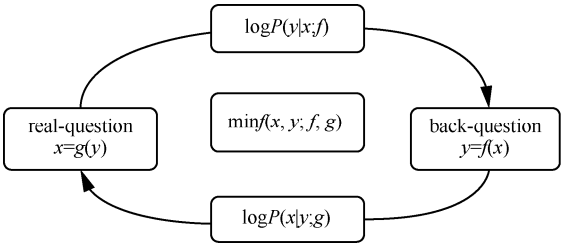


图 4 PT-Sem2SQL 模型的自修正模块

修正函数定义为

$$\min f(x,y;f,g)=\left[\log P(x)+\log P(y\mid x;f)-\log P(y)-\log P(x\mid y;g)\right]^2 \tag{20}$$

4 实验

4.1 实验环境和数据集

本文使用复杂跨领域的 Spider 数据集[7]进行初步验证，包括 11 840 个查询问题，其中有 6 445 个独特的复杂跨领域 SQL 查询和 206 个具有多个表的数据库。同时，这也是一个具有复杂跨域 SQL 查询的新型 Text-to-SQL 数据集。Spider 数据集同时汇聚 6 个现有数据集中的数据，分别为 Restaurants[32-33]、GeoQuery[34]、Scholar[35]、Academic[36]、Yelp 和 IMDB[37]。本文将 Spider 数据集随机划分为 3 个部分进行实验，即训练集（8 659 个查询问题）、验证集（1 034 个查询问题）、测试集（2 147 个查询问题）。为进一步验证 PT-Sem2SQL 模型在上下文相关跨领域 Text-to-SQL 数据集中的效果，本文使用 SParC 数据集进行更进一步实验。在 SParC 数据集中同样采用随机划分进行实验，即训练集（3 024 个查询问题）、验证集（422 个查询问题）、测试集（842 个查询问题）。

本文的 PT-Sem2SQL 模型是在 Python 3.6 上采用 PyTorch 并在 MT-DNN 之上构建实现的。具体来说，模型使用全局学习率为 10^{-5} 的 Adam 优化器，其中，$\beta_1=0.9$ 或 $\beta_1=0.999$。同时，根据 2017 年 Smith 等[38]提出的增加训练过程中的 Batch Size，能够在训练集和测试集上取得类似学习率衰减表现的思想，设置 Batch Size 大小为 32。PT-Sem2SQL 模型采用全链接层的注意力机制，Dropout 参数选自{0.1，0.2，0.3，0.4，0.5，0.6}，并通过参数调整性实验训练 10 轮，选择在验证集上的最佳匹配模型将 Dropout 设置为 0.2。

4.2 Spider 数据集

4.2.1 Spider 准确度测量

本文通过对比先前的模型来评估 PT-Sem2SQL 模型的执行效果。表 2 为模型在验证集和测试集上文本到 SQL 查询的准确性。为了对比展现自修正模块的影响程度，在表 2 中添加消融性实验结果展示行。

表 2 Spider 数据集中不同难度水平下各模型的准确度测量

模型方法	测试集					验证集
	Easy	Medium	Hard	Extra Hard	All	All
seq2seq	11.9%	1.9%	1.3%	0.5%	3.7%	1.9%
SQLNet	26.2%	12.6%	6.6%	1.3%	12.4%	10.9%
TypeSQL	19.6%	7.6%	3.8%	0.8%	8.2%	8.0%
IncSQL	29.5%	13.1%	12.3%	2.5%	14.6%	13.2%
SyntaxSQLNet	48.0%	27.0%	24.3%	4.6%	27.2%	24.8%
SQLove	56.7%	32.9%	22.3%	4.3%	29.1%	27.9%
PT-Sem2SQL-SE	68.1%	40.2%	31.6%	7.3%	36.8%	33.7%
PT-Sem2SQL	73.2%	45.1%	36.2%	9.2%	40.9%	38.1%

由表 2 实验结果可知，对比其他基线 Text-to-SQL 模型（包括最新的 SQLove 模型），在 Spider 数据集上本文模型都表现出较好的准确性。同时在 PT-Sem2SQL 模型中，使用自修正技术导致执行精度在测试集中从原先的36.8%提高到40.9%，在验证集从原先的 33.7%提高到 38.1%。由此可见，采用自修正方法可以显著改善效果。

4.2.2　各模块准确度测量

2018 年，Yu 等将 SQL 子句分解为 5 个部分（如“SELECT”“WHERE”“GROUP BY”“ORDER BY”和“KEYWORDS”）来计算组件匹配 F1 分数[7]。本文模型也效仿该方式，检查各个组件是否完全匹配，如表 3 所示。

从表3的结果可以看出，本文设计的PT-Sem2SQL模型是第一个在各个模块都超过60%准确度的模型。特别地，PT-Sem2SQL 模型在最具挑战性的[WHERE]子句上取得很好的准确率。这主要归功于重新设计增加[Zero]和[CON-TI]的编码模块。该模块在 MT-DNN 预处理的基础上重新结合 Kullback-Leibler 差异技术定义 KL 值增加[Zero]，再利用[CON-TI]部分增强句内语义信息，为模型提供更多的示例，贴合满足复杂跨领域数据的查询需求。

同时，本文认真分析错误检索表中的错误输出问题。关注到主要是因为在使用模板自动生成查询时，概率性引入一些错误。例如，问题“What is the maximum percentage grown 2000—2008 in burundi?”与“year”有关，错误输出 SQL 查询包含不必要的“COUNT”。同时，另外一个值得注意的错误是因为人们设计的训练数据中没有考虑自然语言的模糊性表达问题。虽然针对 Spider 数据集此类问题不是关键性问题，但对于其他大多数任务却是共性问题。比如针对同样的自然语言查询，一些人倾向于使用自然语言“和”，而另外一些人倾向于使用自然语言“或”进行表达。

4.2.3　各训练数据量准确度测量

为观测各模型在不同训练数据量下的表现，本文选择在{20%, 40%, 60%, 80%, 100%}训练集下刻画准确度趋势，如图 5 所示。

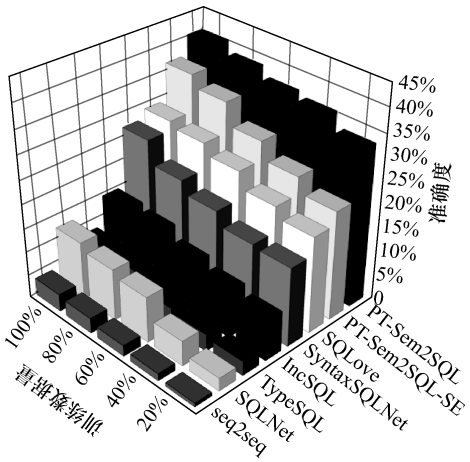


图 5　各模型在不同训练数据量下的准确度趋势

实验结果表明，在不同训练数据量下，PT-Sem2SQL 的性能都优于其他 6 个基线模型。此外，随着训练数据量的增加，PT-Sem2SQL 可以实现的性能改进尤为突出，准确度测量高于其他模型。其主要原因是随着训练数据资源的增加，PT-Sem2SQL 可以更好地训练问题分解器并进行信息提取，进而生成更准确的查询子模块以及准确填

表 3　**Spider 测试集上各组件匹配的 F1 分数**

模型方法	SELECT	WHERE	GROUP BY	ORDER BY	KEYWORDS
seq2seq	13.0%	1.5%	3.3%	5.3%	8.7%
SQLNet	44.5%	19.8%	29.5%	48.8%	64.0%
TypeSQL	36.4%	16.0%	17.2%	47.7%	66.2%
IncSQL	59.2%	41.9%	23.4%	52.7%	67.3%
SyntaxSQLNet	62.5%	34.8%	55.6%	60.9%	69.6%
SQLove	78.4%	47.8%	61.6%	66.2%	74.6%
PT-Sem2SQL-SE	81.3%	50.4%	68.0%	69.1%	77.4%
PT-Sem2SQL	89.2%	65.9%	77.1%	75.6%	86.1%

表 4 SParC 数据集中不同难度水平下各模型的准确度测量

模型方法	测试集					验证集
	Easy	Medium	Hard	Extra Hard	All	All
SyntaxSQLNet	38.4%	8.7%	2.4%	1.3%	12.7%	11.9%
CD-Seq2Seq	36.1%	8.4%	3.7%	2.1%	12.6%	11.8%
SQLove	32.2%	6.9%	1.3%	0.5%	10.2%	8.9%
PT-Sem2SQL-SE	46.3%	23.8%	12.4%	7.3%	22.5%	19.7%
PT-Sem2SQL	53.1%	28.4%	15.5%	9.2%	26.6%	24.1%

充提取的关键词信息。这些有助于 PT-Sem2SQL 在 Text-to-SQL 语义解析过程中获得更好的逻辑形式结果，提升模型准确度。

4.3 SParC 数据集

4.3.1 SParC 准确度测量

与 Spider 数据集类似，本文的模型准确度测量重点对比 2018—2019 年 Yu 等所提出的基线 SyntaxSQLNet[31]、CD-Seq2Seq[39]模型与 2019 年 Hwang 等所提出的基线 SQLove 模型[29]。Spider 数据集上不同难度水平占比分别为：Easy（40%）、Medium（37%）、Hard（12%）、Extra Hard（11%）。对比实验结果如表 4 所示。

从表 4 的结果可以看出，PT-Sem2SQL 的测试集准确度性能首次达到 26.6%。进行自修正模块消融实验后模型在验证集和测试集上的准确性仍优于其他基线模型，可以基本解决上下文相关 Text-to-SQL 任务。通过消融实验可以看出，自修正模块对模型性能贡献度较高，在测试集中执行精度从原先的 22.5%提高到 26.6%，在验证集中执行精度从原先的 19.7%提高到 24.1%。

4.3.2 问题匹配度与上下文相关交互匹配度测量

2019 年，Yu 等将上下文感知模型的性能测试分解为 2 个部分（问题匹配度测量和上下文相关交互匹配度测量），以此计算上下文相关跨领域 Text-to-SQL 的模型匹配度[8]。本文模型也效仿该方式，检查模型是否完全匹配，具体如表 5 所示。

实验结果表明，PT-Sem2SQL 模型性能优于其他基线模型，问题匹配度高达 34.1%，而上下文相关交互匹配度达 13.4%，相比最佳历史基线 CD-Seq2Seq 模型提升 5.9%。当模型进行消融实验后，问题匹配度由 34.1%降低到 32.7%，上下文相关交互匹配度由 13.4%降低到 11.9%。主要是因为模型在自修正模块可以分析历史问题差异性，经过修正模块后可解析问题间交互差异，有效扩展上下文相关交互通路。同时，通过表 5 可以看出，由于 SParC 数据集复杂度大幅增加，各模型匹配度表现欠佳，仍有很大的改进空间。

表 5 问题匹配度与上下文相关交互匹配度测量

模型方法	问题匹配度		上下文相关交互匹配度	
	验证集	测试集	验证集	测试集
SyntaxSQLNet	18.5%	20.2%	4.3%	5.2%
CD-Seq2Seq	21.9%	23.2%	8.1%	7.5%
SQLove	20.3%	22.0%	6.9%	6.4%
PT-Sem2SQL-SE	29.7%	32.7%	13.1%	11.9%
PT-Sem2SQL	32.5%	34.1%	15.8%	13.4%

5 结束语

本文提出了 PT-Sem2SQL 模型，重新解构复杂语义解析问题，强化 Text-to-SQL 任务的上下文信息。同时，本文基于最先进的 MT-DNN 预训练技术重新设计模型的编码模块，成功解决在复杂语义 Text-to-SQL 任务数据集中的列预测问题。针对解码过程中的错误输出问题，PT-Sem2SQL 模型自修正模块优化模型。通过 Spider 和 SParC 数据集的不同模型对比实验结果表明，本文的模型展示优于所有基线模型，表现出卓越性能。这些新的尝试为复杂的跨域 Text-to-SQL 任务提供有效技术支持，也希望下一步引入知识图谱表征方法，解决局部子句查询出错的问题。

参考文献：

[1] CHOI E, HE H, IYYER M, et al. QuAC: question answering in context[C]//The 2018 Conference on Empirical Methods in Natural Language Processing. 2018: 2174-2184.

[2] SUHR A, IYER S, ARTZI Y. Learning to map context-dependent sentences to executable formal queries[C]//The 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers). 2018: 2238-2249.

[3] 李青, 钟将, 李立力, 等. 领域专业知识富关联关系提取方法研究[J]. 控制与决策, 2019: 1-9.
LI Q, ZHONG J, LI L L, et al. Research on the extraction method of multiple semantic relations in domain knowledge[J]. Control and Decision, 2019: 1-9.

[4] 孙加东, 赵铁军, 梁华参. 基于结构对齐的统计机器翻译模型[J]. 通信学报, 2009, 30(7) : 124-129.
SUN J D, ZHAO T J, LIANG H S. Statistical machine translation model based on structure alignment[J]. Journal on Communications, 2009, 30(7): 124-129.

[5] LI Q, ZHONG J, TAO Y, et al. Research of the processing technology for time complex event based on LSTM[J]. Cluster Computing, 2019, 22: 9571-9579.

[6] GUO Y, SHANG X, LI Z. Identification of cancer subtypes by integrating multiple types of transcriptomics data with deep learning in breast cancer[J]. Neurocomputing, 2019, 324: 20-30.

[7] 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]//The 2018 Conference on Empirical Methods in Natural Language Processing. 2018: 3911-3921.

[8] YU T, ZHANG R, YASUNAGA M, et al. SParC: cross-domain semantic parsing in context[J]. arXiv Preprint, arXiv:1906.02285, 2019.

[9] ZHONG V, XIONG C, SOCHER R. Seq2sql: generating structured queries from natural language using reinforcement learning[J]. arXiv Preprint, arXiv:1709.00103, 2017.

[10] United States. Defense advanced research projects agency. speech and natural language: proceedings of a workshop held at Hidden Valley, Pennsylvania, June 24-27, 1990[M]. Morgan Kaufmann Pub, 1990.

[11] DAHL D A, BATES M, BROWN M, et al. Expanding the scope of the ATIS task: The ATIS-3 corpus[C]//The Workshop on Human Language Technology. Association for Computational Linguistics, 1994: 43-48.

[12] FINEGAN-DOLLAK C, KUMMERFELD J K, ZHANG L, et al. Improving Text-to-SQL Evaluation Methodology[C]//The 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2018: 351-360.

[13] DEVLIN J, CHANG M W, LEE K, et al. BERT: pre-training of deep bidirectional transformers for language understanding[C]//The 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers). 2019: 4171-4186.

[14] LIU X, HE P, CHEN W, et al. Multi-task deep neural networks for natural language understanding[J]. arXiv Preprint, arXiv:1901.11504, 2019.

[15] SOLOMON J, CHARETTE F. Hierarchical multi-task deep neural network architecture for end-to-end driving[J]. arXiv Preprint, arXiv:1902.03466, 2019.

[16] KULLBACK S. Letter to the editor: the Kullback-Leibler distance[J]. American Statistician, 1987, 41(4): 340-341.

[17] WARREN D H D, PEREIRA F C N. An efficient easily adaptable system for interpreting natural language queries[J]. Computational Linguistics, 1982, 8(3-4): 110-122.

[18] WONG Y W, MOONEY R. Learning synchronous grammars for semantic parsing with lambda calculus[C]//The 45th Annual Meeting of the Association of Computational Linguistics. 2007: 960-967.

[19] ZELLE J M, MOONEY R J. Learning to parse database queries using inductive logic programming[C]//The National Conference on Artificial Intelligence. 1996: 1050-1055.

[20] ZETTLEMOYER L S, COLLINS M. Learning to map sentences to logical form: structured classification with probabilistic categorial grammars[C]//The Twenty-First Conference on Uncertainty in Artificial Intelligence. AUAI Press, 2005: 658-666.

[21] ZHAO K, HUANG L. Type-driven incremental semantic parsing with polymorphism[C]//The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. 2015: 1416-1421.

[22] HERZIG J, BERANT J. Decoupling structure and lexicon for zero-shot semantic parsing[C]//The 2018 Conference on Empirical Methods in Natural Language Processing. 2018: 1619-1629.

[23] SUTSKEVER I, VINYALS O, LE Q V. Sequence to sequence learning with neural networks[C]//Advances in Neural Information Processing Systems. 2014: 3104-3112.

[24] BAHDANAU D, CHO K, BENGIO Y. Neural machine translation by jointly learning to align and translate[C]//International Conference on Learning Representations, abs/1409.0473, 2014.

[25] YU T, LI Z, ZHANG Z, et al. TypeSQL: knowledge-based type-aware neural Text-to-SQL generation[C]//The 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers). 2018: 588-594.

[26] DONG L, LAPATA M. Coarse-to-fine decoding for neural semantic parsing[C]//The 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2018: 731-742.

[27] WANG C, BROCKSCHMIDT M, SINGH R. Pointing out SQL queries from text[J]. 2018.

[28] PENNINGTON J, SOCHER R, MANNING C. Glove: global vectors

for word representation[C]//The 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). 2014: 1532-1543.

[29] HWANG W, YIM J, PARK S, et al. A comprehensive exploration on WikiSQL with table-aware word contextualization[J]. arXiv Preprint, arXiv:1902.01069, 2019.

[30] RADFORD A, NARASIMHAN K, SALIMANS T, et al. Improving language understanding by generative pre-training[J]. 2018.

[31] YU T, YASUNAGA M, YANG K, et al. SyntaxSQLNet: Syntax tree networks for complex and cross-domain Text-to-SQL task[C]//The 2018 Conference on Empirical Methods in Natural Language Processing. 2018: 1653-1663.

[32] TANG L R, MOONEY R J. Using multiple clause constructors in inductive logic programming for semantic parsing[C]//European Conference on Machine Learning. Springer, 2001: 466-477.

[33] POPESCU A M, ETZIONI O, KAUTZ H. Towards a theory of natural language interfaces to databases[C]//The 8th International Conference on Intelligent User Interfaces. ACM, 2003: 149-157.

[34] ZELLE J M, MOONEY R J. Learning to parse database queries using inductive logic programming[C]//The National Conference on Artificial Intelligence. 1996: 1050-1055.

[35] IYER S, KONSTAS I, CHEUNG A, et al. Learning a neural semantic parser from user feedback[J]. arXiv Preprint, arXiv:1704.08760, 2017.

[36] LI F, JAGADISH H V. Constructing an interactive natural language interface for relational databases[J]. Proceedings of the VLDB Endowment, 2014, 8(1): 73-84.

[37] YAGHMAZADEH N, WANG Y, DILLIG I, et al. SQLizer: query synthesis from natural language[J]. Proceedings of the ACM on Programming Languages, 2017, 1(OOPSLA): 63.

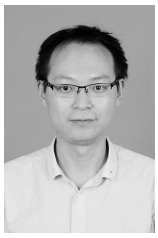
[38] SMITH S L, KINDERMANS P J, YING C, et al. Don't decay the learning rate, increase the batch size[J]. arXiv Preprint, arXiv: 1711.00489, 2017.

[作者简介]

李青（1989− ），女，陕西西安人，重庆大学博士生，主要研究方向为自然语言处理、复杂事件检测、医学信息学。

钟将（1974− ），男，重庆人，博士，重庆大学教授，主要研究方向为自然语言处理、数据挖掘。

李立力（1989− ），男，陕西铜川人，博士，重庆大学博士生，主要研究方向为桥梁健康监测、数据挖掘。



李琪（1987− ），男，江苏盱眙人，博士，绍兴文理学院讲师，主要研究方向为图计算、数据挖掘。