基于 BiLSTM 的 NL2SQL 模型

邰伟鹏[1,2] 刘 杨[1] 王小林[1,2] 郑 啸[1] 钟 亮[3]

[1](安徽工业大学计算机科学与技术学院 安徽 马鞍山 243000)

[2](安徽工业大学信息技术学院 安徽 马鞍山 243000)

[3](上海量镜空间信息技术有限公司 上海 200000)

摘 要 随着互联网技术的发展,很多应用为大众提供金融量化服务,而大部分用户不具备金融或计算机专业知识,他们期望使用自然语言查询数据,因此自然语言转 SQL(NL2SQL)被迫切需要。针对此问题,提出一种基于双向长短期记忆模型(BiLSTM)的中文金融 NL2SQL 算法,分为编码和解码阶段。在编码阶段,利用 BiLSTM 和注意力机制生成特征向量。在解码阶段,根据 SQL 的语法规则,将 SQL 生成解耦为九个分类任务,各个任务间相互依赖联合学习,之后生成复杂的 SQL 语句。除模型外,还训练出包含金融词汇的向量库,构建金融领域的数据集。通过在此数据集上实验验证,结果表明,该方法准确率更高,能有效解决金融领域 SQL 生成问题,并在某金融量化分析系统中实现。

关键词 NL2SQL BiLSTM 注意力机制 向量库 数据集

中图分类号 TP3 **文献标志码** A DOI:10.3969/j.issn.1000-386x.2024.03.006

NL2SQL MODEL BASED ON BILSTM

Tai Weipeng[1,2] Liu Yang[1] Wang Xiaolin[1,2] Zheng Xiao[1] Zhong Liang[3]

[1](*School of Computer Science and Technology, Anhui University of Technology, Maanshan 243000, Anhui, China*)

[2](*Institute of Information Technology, Anhui University of Technology, Maanshan 243000, Anhui, China*)

[3](*Shanghai Measuring Mirror Space Information Technology Co., Ltd., Shanghai 200000, China*)

Abstract With the development of Internet technology, many applications provide financial quantification services for the public, but most users do not have financial or computer professional knowledge, they expect to use natural language to query data, so natural language to SQL (NL2SQL) is urgently needed. To solve this problem, a Chinese financial NL2SQL algorithm based on BiLSTM is proposed, which is divided into encoding and decoding stages. In the encoding stage, feature vectors were generated by BiLSTM and attention mechanism. In the decoding stage, the SQL generation was decoupled into nine classified tasks according to the SQL syntax rules, and each task was interdependent and joint learning, and then the complex SQL statement was generated. In addition to the model, a vector library containing financial vocabulary was trained, which built data sets for the financial domain. The experimental verification on this data set shows that the method has higher accuracy, can effectively solve the problem of SQL generation in the financial field, and is implemented in a financial quantitative analysis system.

Keywords NL2SQL BiLSTM Attention mechanism Vector library Dat set

0 引 言

NL2SQL 作为语义分析的子任务受到了广泛关注,常用于医学报告分析、智能搜索引擎等领域。如今量化金融应用发展迅速,应用中往往包含大规模的数据,用户希望快速准确地实现与数据库交互,不过查询数据库中的数据需要通过 SQL 语句,这需要懂 SQL 的

收稿日期:2021-01-12。安徽高校自然科学研究重大项目(KJ2019ZD09);安徽省重点研究与开发计划项目(202004a07020028);安徽省高校协同创新项目(Grant No. GXXT-2019-025)。**邰伟鹏**,副教授,主研领域:时空数据库,自然语言处理。**刘杨**,硕士生。**王小林**,教授。**郑啸**,教授。**钟亮**,博士。

专业技术人员来执行这一操作。而大多数用户不懂计算机相关的专业知识,为了让非专业用户也可以按需查询数据库,NL2SQL 技术在金融应用中被迫切需要。它可以改善用户与数据库之间的交互方式,提升查询数据的效率,充分挖掘了数据的价值。早期的 NL2SQL 的研究主要面向英文文本进行分析,直到近期中文版本 NL2SQL 的研究才有新的进展。

NL2SQL 经过多年的研究可以分为两大类:(1) 基于传统机器学习非端到端的方法,先将自然语言查询(Query)转为中间表达再生成 SQL 语句;(2) 基于深度学习的端到端[1]方法,直接输入自然语言查询,经由训练好的神经模型输出预测结果。非端到端传统的方法需要借助中间表达,要求 Query 规则化,不能处理复杂的用户输入,且每一步是独立的任务,结果会影响下一步骤而影响整个模型的结果。端到端的方法基于深度学习模型,比较方便用户与数据库交互的处理,但也有一定的局限性,模型可解释性及灵活性较低,无法明确模型各组件对最终结果的影响,数据标注的成本较高。目前还没有应用于中文金融领域的端到端模型,且没有专属于金融领域的数据集,无法支持如图 1 所示的中文金融问题的多表查询。

例:

Query:总市值为142000亿的股票编码有哪些?

SQL:SELECT T_基本信息.股票编码 FROM T_基本信息 JOIN T_股票行情 ON T_基本信息.股票名称 = T_股票行情.股票名称 WHERE T_股票行情.总市值 = 142000;

图 1　自然语言查询生成 SQL 语句的实例

从上述研究问题出发,本文针对中文金融应用中自然语言生成 SQL 语句的问题提出解决方法,主要工作和贡献可以概括为以下三个方面:

(1) 构建金融领域的中文数据集,共有 21 899 条自然语言查询和包含多表、嵌套查询等的复杂 SQL;在数据库设计中实现多表关联;训练出含有金融词汇的词向量库,如股票名称、股票编码等。

(2) 提出深度学习的分类模型,通过该模型生成复杂的 SQL 语句,包括 JOIN、EXISTS、GROUP BY 等关键词;在预测 SQL 中的选择列时,将单个扩充为多个;条件值的生成模块改善了 Query 与数据库列值表述不相同的问题;条件关系模块解决了传统模型只能以 AND 连接的问题。

(3) 通过在本文搭建的数据集上生成 SQL 语句,经对比证明该方法有更高的准确度。

1　相关工作

NL2SQL 在数据库的数据规模和复杂度日益增大的背景下发挥着重要的作用,用户通过自然语言准确查询数据库中的目标关系和数据集合成为了新兴的研究热点。早期的 NL2SQL 需要人工制定固定的规则,近年来的 NL2SQL 开始使用深度学习的方法,自动生成 SQL 语句。下面将概述已有的 NL2SQL 的研究工作及成果。

第一种是非端到端的方法,即不直接生成 SQL 语句,通过中间表达推断出 SQL 语句。Dong 等[2]在 2018 年提出 Coarse-to-fine 的模型,将 NL2SQL 任务分为两个阶段,给定输入的问句,首先生成粗略的草图,然后进一步填补缺失的细节。Guo 等[3]在 2019 年提出了 IRNET 的方法,该方法将 SQL 生成分为三个阶段,先将一个问题和一个数据库执行模式连接,然后采用神经模型合成 SEMQL 查询,最终从合成的 SEMQL 查询中确定 SQL 语句。但是在预测合成、最终确定 SQL 的过程中,只要有一个过程预测出错,这个错误会一直延续,导致最终的结果出现偏差。

随着深度学习的不断发展,端到端的方法也逐步走向成熟。Zhong 等[4]在 2017 年提出的 SEQ2SQL 模型借鉴了 Seq2Seq[5]的思想,利用基于策略的强化学习方法生成 SQL。然而同一个 SQL 语句可能会有多个等价序列化,Seq2SQL 采用强化学习来选择其中一个的效果不理想。于是 Xu 等[6]提出了 SQLNET 的模型,他们使用一种基于 Sketch 的方法,预先定义好 SQL 的通用模板,然后将 SQL 的生成细分为 6 个子模型,每个子模型负责生成 SQL 语句相应的部分,但无法预测条件子句之间的关系,选择列的数目也只能为 1。He 等[7]提出了 X-SQL 模型,利用了一种预训练模型 MT-DNN[8],将输入的自然语言变为全局上下文表示用于下游任务,使用 K-L 散度优化模型提高模型的准确度,该模型虽然解决了 SQLNET 模型的问题,但是只针对英文数据集。Zhang 等[9]在追一科技竞赛中以 X-SQL 模型为基线,提出了一种多任务联合学习框架 M-SQL,该模型使用 TableQA 中文数据集[10]和 BERT-WWM-EXT 的预训练模型[11],然而局限于单表查询,不能用于中文金融领域复杂 SQL 的生成。本文构建的数据集、提出的模型能够解决这个问题,从而实现中文金融领域非结构化自然语言查询向 SQL 语句的转化。

2 问题定义

中文金融领域自然语言生成 SQL 语句形式化定义如下：

$$(Q,S)\xrightarrow{M}SQL \tag{1}$$

$$Q=\{w_1,w_2,\cdots,w_i,\cdots,w_n\} \tag{2}$$

$$S=\{S_1,S_2,\cdots,S_i,\cdots,S_m\} \tag{3}$$

式中：Q(Query)表示中文金融领域自然语言查询，共由 n 个词语组成，w_i 表示 Query 第 i 个词语；M 为本文各模块的分类模型；S 表示数据库集合，由 m 个表构成。

$$S_i=\{(T_i,C_1,t_1),(T_i,C_2,t_2),\cdots,(T_i,C_j,t_j),\cdots,(T_i,C_k,t_k)\} \tag{4}$$

式中：T_i 表示第 i 个表名，C_j 表示表 i 的第 j 个列名；t_j 表示表 i 中第 j 列的类型(数据类型、主键或外键)。生成的 SQL 语句规则如图 2 所示。

```
(SELECT    ($AGG    $COL)*
FROM    $TAB    ($JOIN    $TAB)*
$WHERE    ($WOP    $COL    $OP    $VAL)*
          / $NOT    $EXISTS    ($SELECT...)
$GROUP BY    $COL
$HAVING    ($AGG    $COL)    $OP    $VAL
$ORDER BY    $COL    $DESC/ASC)
$UNION/$EXCEPT/$INTERSECT
($SELECT...)
```

图 2 SQL 语句规则

图 2 中 $ 之后的名称为预测类型[12]，只需要预测以 $ 开头标记的位置便可以生成完整的 SQL 语句，$ 后预测值不存在时，取值为 NONE。COL 表示数据库表中选取的列名；AGG 是聚合函数，聚合函数取值集合是[NONE, SUM, COUNT, MAX, MIN, AVG]；OP 是列的操作符，候选集合为[==, ! =, >, <, <=, >=]；WOP 表示条件列之间的关系操作符，候选集合为[AND, OP]；VAL 是 WHERE/HAVING 条件的值，该值为数据库表中的列值。当然，Query 中不一定会出现与表中列值完全相同的词语，下文将结合自然语言，对应数据表中的列名及列值共同预测 VAL。TAB 是数据库中的表，* 表示括号中内容若存在则至少有一个。

3 方法和模型

本文针对传统的 NL2SQL 算法不能解决中文金融领域自然语言生成复杂 SQL 的问题，分析并提出解决算法，由两个部分构成，编码器和各个子模块的解码，总体框架如图 3 所示。

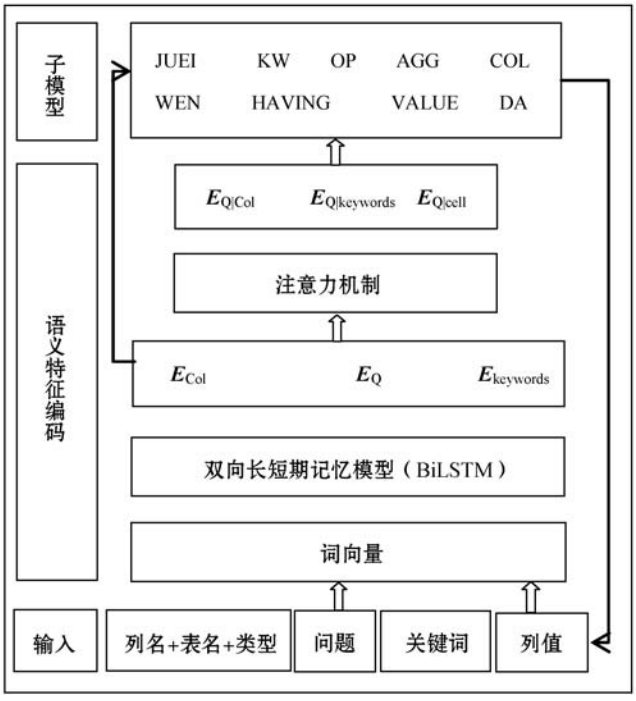


图 3 模型的整体架构

在编码器中，使用双向 LSTM 模型(BiLSTM)，相比 LSTM 能够充分学习自然语言上下文的语义信息，捕获文本中长距离的依赖关系。由于中文文本语义丰富，词与词之间没有明显的间隔，对 Query、数据库中的表名及列名预处理后，将 Query 中文词向量载入到 BiLSTM 模型中，输出特征向量 $\boldsymbol{E}_Q$；对于列名，连接相应的表名、列的类型(数据类型、主键或外键)得到列名向量，经过 BiLSTM 编码后生成特征向量 $\boldsymbol{E}_{Col}$；SQL 中 JOIN、GROUP BY、EXISTS 等统称为关键词，经过 BiLSTM 后输出 $\boldsymbol{E}_{keywords}$ 向量，下文将结合具体的任务生成关键词特征向量。与 SQLNET 类似，使用注意力机制[13]来增强不同的列名、关键词在 Query 中的重要程度，获得特征向量 $\boldsymbol{E}_{Q|Col}$、$\boldsymbol{E}_{Q|keywords}$ 后作为解码器的输入。解码器即 SQL 语句的生成，分为多个模块，实质上是多个分类任务，将各个任务的结果组装成符合语法规则的 SQL 语句。与 SQLNET 不同的是，模型训练时，由于是有监督学习，各个部分对应的标签是已知的，因此使用并行的方式训练各个任务。在模型测试时，许多任务的结果依赖于其他任务的预测结果，这部分进行顺序执行，不能并行处理。

具体而言，整个模型分为九个模块，JUEI、KW、COL、OP、AGG、WEN、DA、HAVING、VALUE。JUEI 模块预测 JOIN、UNION、EXCEPT、INTERSECT，决定是否生成多表嵌套的 SQL 语句；KW 模块预测 GROUP BY、

ORDER BY、WHERE、SELECT,默认最外层 SQL 语句包括 SELECT;COL 模块预测列名;OP 模块预测列与值的关系,当列为文本类型时,OP 只能为“==”或“!=”,为数字类型时,OP 取值集合为[==,!=,>,<,<=,>=];AGG 预测列的聚合符,当列类型为文本时,AGG 为 NONE,为数值时,AGG 取值集合为[NONE,SUM,COUNT,MAX,MIN,AVG];WEN 模块预测 WOP、EXISTS、NOT EXISTS,WOP 表示各条件列之间的关系;DA 模块预测 DESC、ASC,即查询结果的排序方式;若 GROUP BY 存在,则预测 HAVING 模块;VALUE 预测列关联的值,传统的算法基本上都是从 Query 中提取,但是由于中文语义的丰富性,提取的值可能不在表列值中出现,因此基于此问题对 VALUE 预测特殊处理,首先对 Query 进行规则修正,将数字、年份、单位、日期统一标准化,并用 Pycorrect 包识别更正错别字,如表 1 所示。

表 1　自然语言查询预处理

自然语言查询	规则修正
八月份哪些股票走势较好?	8 月份哪些股票走势较好?
五月一日后市场有什么行情?	劳动节后市场有什么行情?
茅台在元旦后可以买吗?	贵州茅台在元旦后可以买吗?
现在有什么月度季节性规律?	2021/2 有什么月度季节性规律?

通过之前对 COL 的预测,锁定目标列名 COL 下的所有列值,将列值向量输入 BiLSTM 捕获特征,最后使用分类模型确定 VALUE。

3.1　分类模型细节

在获取 Query 的上下文特征向量 $\boldsymbol{E}_{\mathrm{Q}}$、表列名特征向量 $\boldsymbol{E}_{\mathrm{Col}}$ 及 SQL 关键词 $\boldsymbol{E}_{\mathrm{keywords}}$ 向量后,根据预测各个模块所需要的不同向量,开始构建分类模型。本文对 COL 等模块的预测,不仅依赖于 Query,与表中的列名也有关。若只使用 $\boldsymbol{E}_{\mathrm{Q}}$,只能表示自然语言查询中的信息,无法表示表中不同的列名在 Query 中的重要程度,因此,采用注意力机制[12]表示不同列名在自然语言查询中的侧重点,从而得到列名对应 Query 的特征向量 $\boldsymbol{E}_{\mathrm{Q|Col}}$。

首先计算 Query 中包含列名信息各个词语的权重 w,计算公式为:

$$w=\mathrm{softmax}(v) \tag{5}$$

$$\boldsymbol{v}_i=\boldsymbol{E}_{\mathrm{Col}}\boldsymbol{W}\boldsymbol{E}_{\mathrm{Q}}^{i}\quad i\in\{1,2,\cdots,L\} \tag{6}$$

式中:L 为自然语言查询中词语的个数;$\boldsymbol{E}_{\mathrm{Q}}^{i}$ 表示 Query 中第 i 个词语经过 BiLSTM 隐藏层的输出;$\boldsymbol{W}$ 为可训练的模型参数。之后就能得到 $\boldsymbol{E}_{\mathrm{Q|Col}}$ 的特征向量表示:

$$\boldsymbol{E}_{\mathrm{Q|Col}}=\boldsymbol{E}_{\mathrm{Q}}w \tag{7}$$

SQL 规则下的各个模块是多分类任务,采用 softmax 函数进行分类。假设已生成的不完整 SQL 语句特征向量为 $\boldsymbol{E}_{\mathrm{PH}}$,初始 SQL 语句为 NONE。预测 JUEI 模块,关键词集合为{JOIN、UNION、EXCEPT、INTERSECT、NONE},将关键词输入 BiLSTM 得到特征向量 $\boldsymbol{E}_{\mathrm{JUEI}}$,利用注意力机制生成 $\boldsymbol{E}_{\mathrm{PH|JUEI}}$、$\boldsymbol{E}_{\mathrm{Q|JUEI}}$ 向量。根据 $\boldsymbol{E}_{\mathrm{Q|JUEI}}$、$\boldsymbol{E}_{\mathrm{PH|JUEI}}$、$\boldsymbol{E}_{\mathrm{Q}}$、$\boldsymbol{E}_{\mathrm{JUEI}}$ 开始预测 SQL 语句中含有上述关键词的数量,然后预测关键词,分类概率计算公式如下:

$$P_{\mathrm{JUEI}}^{\mathrm{num}}=\mathrm{softmax}(U\tanh(U_1^{\mathrm{num}}\boldsymbol{E}_{\mathrm{Q|JUEI}}+U_2^{\mathrm{num}}\boldsymbol{E}_{\mathrm{PH|JUEI}})) \tag{8}$$

$$P_{\mathrm{JUEI}}^{\mathrm{val}}=\mathrm{softmax}(U\tanh(U_1^{\mathrm{val}}\boldsymbol{E}_{\mathrm{Q|JUEI}}+U_2^{\mathrm{val}}\boldsymbol{E}_{\mathrm{JUEI}}+U_3^{\mathrm{val}}\boldsymbol{E}_{\mathrm{PH|JUEI}})) \tag{9}$$

式中:U_1^{num}、U_2^{num}、U、U_1^{val}、U_2^{val}、U_3^{val} 表示可训练的模型参数,各个模块之间不共享参数。预测了 JUEI 模块后,预测 KW 模块,关键词集合为{GROUP BY,ORDER BY,WHERE,SELECT,NONE},利用 BiLSTM 和注意力机制得到 $\boldsymbol{E}_{\mathrm{KW}}$、$\boldsymbol{E}_{\mathrm{Q|KW}}$ 向量,则 SQL 中 KW 关键词数量及具体值的分类概率计算公式为:

$$P_{\mathrm{KW}}^{\mathrm{num}}=\mathrm{softmax}(U\tanh(U_1^{\mathrm{num}}\boldsymbol{E}_{\mathrm{Q|KW}}+U_2^{\mathrm{num}}\boldsymbol{E}_{\mathrm{PH|KW}})) \tag{10}$$

$$P_{\mathrm{KW}}^{\mathrm{val}}=\mathrm{softmax}(U\tanh(U_1^{\mathrm{val}}\boldsymbol{E}_{\mathrm{Q|KW}}+U_2^{\mathrm{val}}\boldsymbol{E}_{\mathrm{KW}}+U_3^{\mathrm{val}}\boldsymbol{E}_{\mathrm{PH|KW}})) \tag{11}$$

在 COL 模块中,首先预测 COL 的数量,再预测使用哪些列名,分类概率计算公式为:

$$P_{\mathrm{COL}}^{\mathrm{num}}=\mathrm{softmax}(U\tanh(U_1^{\mathrm{num}}\boldsymbol{E}_{\mathrm{Q|Col}}+U_2^{\mathrm{num}}\boldsymbol{E}_{\mathrm{PH|COL}})) \tag{12}$$

$$P_{\mathrm{COL}}^{\mathrm{val}}=\mathrm{softmax}(U\tanh(U_1^{\mathrm{val}}\boldsymbol{E}_{\mathrm{Q|Col}}+U_2^{\mathrm{val}}\boldsymbol{E}_{\mathrm{Col}}+U_3^{\mathrm{val}}\boldsymbol{E}_{\mathrm{PH|COL}})) \tag{13}$$

根据列名 COL 及外键关系可确定表名。之后预测 COL 与值之间的关系符 OP。OP 不仅和 COL 有关,还与 Query 有关,假设目标列名向量为 $\boldsymbol{E}_{\mathrm{CS}}$,则 OP 的数量及具体值的分类概率计算公式为:

$$P_{\mathrm{OP}}^{\mathrm{num}}=\mathrm{softmax}(U\tanh(U_1^{\mathrm{num}}\boldsymbol{E}_{\mathrm{Q|CS}}+U_2^{\mathrm{num}}\boldsymbol{E}_{\mathrm{PH|CS}})) \tag{14}$$

$$P_{\mathrm{OP}}^{\mathrm{val}}=\mathrm{softmax}(U\tanh(U_1^{\mathrm{val}}\boldsymbol{E}_{\mathrm{Q|CS}}+U_2^{\mathrm{val}}\boldsymbol{E}_{\mathrm{CS}}+U_3^{\mathrm{val}}\boldsymbol{E}_{\mathrm{PH|CS}})) \tag{15}$$

HAVING 通常与 GROUP BY 联合使用,当 GROUP BY 存在时,HAVING 的分类概率计算公式为:

$$P_{\mathrm{HAVING}}=\mathrm{softmax}(U\tanh(U_1\boldsymbol{E}_{\mathrm{Q|CS}}+U_2\boldsymbol{E}_{\mathrm{CS}}+U_3\boldsymbol{E}_{\mathrm{PH|CS}})) \tag{16}$$

在 AGG 模块中,对于列的聚合符,首先预测 AGG 的数量,然后预测具体使用哪些聚合符,分类概率公式为:

$$P_{\mathrm{AGG}}^{\mathrm{num}}=\mathrm{softmax}(U\tanh(U_1^{\mathrm{num}}\boldsymbol{E}_{\mathrm{Q|CS}}+U_2^{\mathrm{num}}\boldsymbol{E}_{\mathrm{PH|CS}})) \tag{17}$$

$$P_{\mathrm{AGG}}^{\mathrm{val}}=\mathrm{softmax}(U\tanh(U_1^{\mathrm{val}}\boldsymbol{E}_{\mathrm{Q|CS}}+U_2^{\mathrm{val}}\boldsymbol{E}_{\mathrm{CS}}+U_3^{\mathrm{val}}\boldsymbol{E}_{\mathrm{PH|CS}})) \tag{18}$$

WEN 模块预测 WOP、EXISTS 及 NOT EXISTS。当条件为多个,条件关系操作符 WOP 可以取 AND 或

OR。WEN 取值集合为{AND、OR、EXISTS、NOT EXISTS、NONE},SQL 含有 WEN 模块中关键词的数量及具体值的分类概率计算公式为:

$$P_{\text{WEN}}^{\text{num}} = \text{softmax}(U\tanh(U_1^{\text{num}}\boldsymbol{E}_{\text{Q|WEN}} + U_2^{\text{num}}\boldsymbol{E}_{\text{PH|WEN}})) \quad (19)$$

$$P_{\text{WEN}}^{\text{val}} = \text{softmax}(U\tanh(U_1^{\text{val}}\boldsymbol{E}_{\text{Q|WEN}} + U_2^{\text{val}}\boldsymbol{E}_{\text{WEN}} + U_3^{\text{val}}\boldsymbol{E}_{\text{PH|WEN}})) \quad (20)$$

在 DA 模块中,预测 ORDER BY 按指定列排序的方式,取值集合为{DESC, ASC, NONE},分类概率公式为:

$$P_{\text{DA}} = \text{softmax}(U\tanh(U_1\boldsymbol{E}_{\text{Q|CS}} + U_2\boldsymbol{E}_{\text{CS}} + U_3\boldsymbol{E}_{\text{PH|CS}})) \quad (21)$$

完成上述分类模型后,只需要预测 VALUE,就可以根据规则组装成完整的 SQL 语句。对 VALUE 的预测,由于 COL 已经预测完成,之后可以得到目标 COL 下的所有列值,经过 BiLSTM 捕获特征向量后,得到 $\boldsymbol{E}_{\text{cell}}$向量,利用这个上下文特征表达,构建分类任务,解决了 VALUE 可能不在 Query 中的问题。同样使用注意力机制,与 $\boldsymbol{E}_{\text{Q|Col}}$ 的计算方法相同,计算得到 $\boldsymbol{E}_{\text{Q|cell}}$。则 P_{VALUE}分类概率计算公式为:

$$P_{\text{VALUE}} = \text{softmax}(U\tanh(U_1^{\text{val}}\boldsymbol{E}_{\text{Q|cell}} + U_2^{\text{val}}\boldsymbol{E}_{\text{CS}} + U_3^{\text{val}}\boldsymbol{E}_{\text{PH|CS}})) \quad (22)$$

由于 SQL 的构造有一定的限制条件,如文本类型的列不能有数值操作。因此,在预测过程中删除组装不合理的 SQL 语句,提高 SQL 生成的准确率。

3.2 目标函数

在训练中,模型的损失函数为各个模块的损失之和,之后定义损失函数来优化模型,通过最小化损失函数更新参数,预测除 COL、VALUE 外的损失函数直接使用交叉熵[14],公式如下:

$$L_{\text{oss}} = \frac{1}{N}\sum_{i=1}^{N} -\sum_{c=1}^{M} y_{ic}\log(P_{ic}) \quad (23)$$

式中:N 为样本总数;M 为类别的数量;y_{ic}取值集合为{0,1},当预测类别与样本 i 的类别相同时是 1,反之为 0;P_{ic}表示预测样本 i 属于 c 类的概率。

对于列名 COL 以及 VALUE 的预测,由于数据库表中一般含有大量的列名和对应的列值,事实上正样本只有少量的列名和列值,其余没有在 SQL 语句中出现的都是负样本,为了排除正负样本数量差异带来的干扰,针对上述任务损失的计算,使用负数带权重的对数似然函数,计算公式为:

$$L_{\text{oss}} = -(\sum_{i=1}^{N} \alpha y_i \log P_i + (1 - y_i)\log(1 - P_i)) \quad (24)$$

式中:P 为 COL 或 VALUE 预测概率;y 是真正的值;α 是保持正负样本平衡的超参数,经过多次训练最终选取 α 等于 3。

4 实验与结果分析

4.1 实验数据集

本文使用中文金融领域的表格数据作为数据源,数据集包括 Query、Query 对应的表及 SQL 语句,如图 4 所示。

Query:总市值为142000亿的股票编码有哪些?

股票编码	股票名称	类型	是否上市
600519	贵州茅台	A股	是
…	…	…	…

T_基本信息

股票名称	总市值	市盈率	成交量
贵州茅台	26583亿	59.63	3.37 万
…	…	…	…

T_股票行情

SQL: SELECT T_基本信息.股票编码 FROM T_基本信息 JOIN T_股票行情 ON T_基本信息.股票名称 = T_股票行情.股票名称 WHERE T_股票行情.总市值 = 142000;

图 4 本文数据集的一个数据样本

与过去主流数据集 WIKISQL[4]、Spider[15]相比,该数据集更复杂,引入了 JOIN、GROUP BY 等高阶操作,并在各条件列的关系中增加了“或”的逻辑关系,查询的中文自然语言表达丰富,包括很多口语化的说法,自然语言查询中的词语可能不会出现在数据库的列值中,同时利用数据增强的方法,最后得到 21 899 个 Query 和 SQL 对。该数据被分为三类:训练集、验证集、测试集,其中:验证数据 3 500,测试数据 3 200。通过训练得到该数据集上最优的模型参数,验证集用来调整模型的超参数,初步评估模型的泛化能力,最终使用测试集评估:根据以上数据集得到的参数及超参数是否具有较强的泛化能力,即模型对于与训练集完全不同的数据也可以有较高的准确度。本文的模型是针对该中文金融数据集提出的新的解决方案。

4.2 实验环境及参数设置

本文训练验证及测试的设备为一台 PC 机,Windows 10 系统,使用 Pycharm 开发软件,模型的搭建基于深度学习的框架 Pytorch。实验软件和硬件的配置环境如表 2 所示。

表 2 实验环境

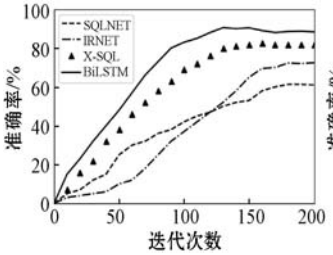
名称	配置
内存/GB	16
GPU	NVIDIA GeForce RTX 2070
Python 版本	Python 3.7.1
PyTorch 版本	PyTorch 1.4

在实验过程中,使用修改后的 Chinese Word Vectors[16] 中文词向量库表示自然语言查询和数据库中的表,由于原始的向量库中不包含金融专业词汇的词向量,通过 Ngram2vec 训练后将缺少的词向量导入到 Chinese Word Vectors 库中。构造 $n \times w \times d$、$t \times v \times d$ 三维矩阵,分别表示 Query 和列名连接表名、列名类型(数据类型、主键或外键)后的向量矩阵。其中:n 表示样本总个数;w 是 Query 分词后的词语个数;d 是隐藏层输出的维度;t 为所有表中列的个数;v 为列名中包含的词语的个数,再根据目标列名下的列值、SQL 语句中的关键词构造矩阵。将上述向量矩阵作为 BiLSTM 的输入,之后输出上下文相关的特征向量,将 BiLSTM 设为 2 层,隐藏层维度设置为 100 维,训练优化器选 Adam[17],训练的迭代次数 epoch 设置为 200,模型中两个超参数:学习率和批量大小,将学习率 learning_rate 设为 0.001,每次批量操作的数据量 batch_size 设为 64,由于模型的拟合能力较强,为了不影响模型的泛化能力,每次迭代开始前要将数据集中数据顺序打乱。

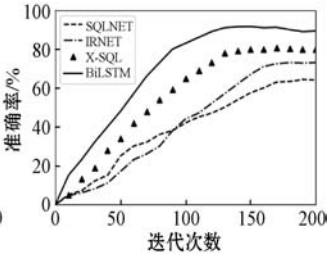
4.3　评估准则及实验结果

本文对模型的评估包括匹配准确率(MX)、生成准确率(GX)和各个子模型的准确率。匹配准确率(MX)即将模型预测的 SQL 语句和数据集中正确的 SQL 语句匹配,忽略列名的排列顺序,即不同列名顺序的 SQL 子句,但表达意思相同是相互匹配的。生成准确率(GX)是预测的 SQL 在数据表中执行结果与真实结果相比较,统计生成结果正确的个数。各个子任务的准确率即更加细粒度的比较,分为 JUEI、KW、COL、OP、AGG、WOP、DA、HAVING、VALUE 准确度计算,能更好地理解模型。

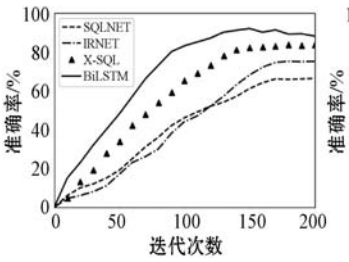
之后根据搭建的数据集,对几个经典模型进行复现。然后使用上述评估方式:匹配准确率(MX)、生成准确率(GX),在验证数据集和测试数据集的基础上,分别对复现的模型和本文提出的模型进行准确率计算,随着迭代次数的变化,各模型的准确率也发生变化,如图 5 所示。



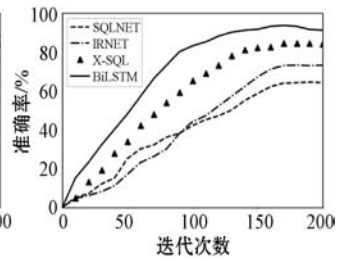
(a) DEV MX



(b) TEST MX



(c) DEV GX



(d) TEST GX

图 5　准确率变化曲线

由图 5 可得,基于 BiLSTM 的 NL2SQL 模型与其他模型相比准确率最高,收敛速度最快。在迭代 100 次左右时模型就趋近于收敛。为了更直观地显示各模型在验证集和测试集上最终的准确度,绘制了表 3。

表 3　各模型在本文数据集上的表现(%)

模型	DEV MX	DEV GX	Test MX	Test GX
SQLNET	62.28	66.20	63.58	68.34
IRNET	72.58	74.89	72.61	73.20
X-SQL	82.22	83.30	83.42	84.58
本文模型	90.58	91.87	91.62	93.43

可以看出,本文模型在数据集上表现优于其他模型,在匹配准确率上,本文模型比 SQLNET 的重新实现高出 28 百分点,在生成准确率上高出 25 百分点,由于 SQLNET 的原始模型是基于英文数据集,自然语言查询比较简单,没有中文的表达复杂,且未对数据预处理,解决了 SQLNET 只能生成单表操作 SQL 语句的问题。从验证集和测试集来看,测试集上的准确率要高于验证集,可能由于测试集的数据质量优于验证集。为了能更加详细了解模型的性能,计算各个模块的准确率,如表 4 所示。

表 4　各模块在本文数据集上的表现(%)

模块	JUEI	KW	COL	OP	AGG	DA	WEN	HAVING	VALUE
准确率	90.57	93.58	93.40	91.34	93.82	90.72	92.30	91.22	85.43

各个模块的准确率都在 85% 以上,最高 93.82%,最低 85.43%,VALUE 准确率较其他子模型略低,可能因为中文语义表达丰富,使用 BiLSTM 无法完全理解自然语言查询中的语义信息。

4.4　应用范例

目前,本文模型已实际应用于金融系统量股棱镜中,使用该系统的用户输入自然语言查询,系统根据模型生成 SQL 语句,返回给用户数据库中的数据。用户输入框如图 6 所示。

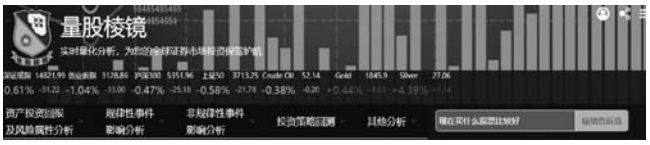


图 6 用户输入框

SQL 语句查询数据库的结果如图 7 所示。

2月胜率>=75%

资产名称	同花顺一级行业	上月末股票总市值(亿)	是否沪深300
三七互娱	信息服务	689.65	沪深300
比亚迪	交运设备	7063.88	沪深300
用友网络	信息服务	1395.01	沪深300
亨通光电	信息设备	293.62	沪深300
广联达	信息服务	985.5	沪深300
美年健康	医药生物	596.92	沪深300
荣盛石化	化工	2539.48	沪深300

图 7 查询结果

5 结 语

本文模型可以用于中文金融领域生成复杂 SQL 的任务。基于本文模型构建了金融领域的数据集，并作为实验数据。与之前存在的数据集 WIKISQL、Spider 相比，此数据集更具有针对性，也更复杂。之后提出将 SQL 语句的生成分为 9 个模块，每个模块都处理为多分类问题，然后利用 SQL 规则格式将各模块的结果组装成 SQL 语句。实验结果表明，本文模型与传统模型相比准确率更高，生成的 SQL 语句更符合实际，已经应用于金融系统中，使非 SQL 专业的用户也可以直接利用自然语言查询数据库中的金融数据。但是本文模型使用有监督学习，训练数据需要大量的人工标注，后续工作将探究如何使用少量的数据标注生成中文金融领域 SQL 语句。

参 考 文 献

[1] Baik C, Jagadish H V, Li Y. Bridging the semantic gap with SQL query logs in natural language interfaces to databases [C]//35th International Conference on Data Engineering, 2019:374 - 385.

[2] 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.

[3] Guo J Q, Zhan Z C, Gao Y, et al. Towards complex text-to-SQL in cross-domain database with intermediate representation[C]//57th Annual Meeting of the Association for Computational Linguistics, 2019:4524 - 4535.

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

[5] Bahdanau D, Cho K, Bengio Y, et al. Neural machine translation by jointly learning to align and translate[C]//3rd International Conference on Learning Representations, 2015: 1 - 15.

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

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

[8] 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.

[9] 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.

[10] Sun N Y, Yang X F, Liu Y F. TableQA: A large-scale Chinese text-to-SQL dataset for table-aware SQL generation[EB]. arXiv:2006.06434, 2020.

[11] Cui Y M, Che W X, Liu T, et al. Pretraining with whole word masking for Chinese BERT[EB]. arXiv:1906.08101, 2019.

[12] 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.

[13] Vshish A, Shazeer N, Parmar N, et al. Attention is all you need[C]//31st International Conference on Neural Information Processing Systems, 2017:6000 - 6010.

[14] Boer P, Kroese D, Mannor S, et al. A tutorial on the cross-entropy method[J]. Annals of Operations Research, 2005, 134:19 - 67.

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

[16] Li S, Zhao Z, Hu R F, et al. Analogical reasoning on Chinese morphological and semantic relations[C]//56th Annual Meeting of the Association for Computational Linguistics, 2018:138 - 143.

[17] Kingma D P, Ba J. Adam: A method for stochastic optimization[EB]. arXiv:1412.6980, 2017.