arXiv:2301.07507v1 [cs.CL] 18 Jan 2023

Graphix-T5：将预训练 Transformer 与图感知层结合，用于文本到SQL解析

Jinyang Li[1, 2*], **Binyuan Hui**[2], **Reynold Cheng**[1, 5†], **Bowen Qin**[3], **Chenhao Ma**[4], **Nan Huo**[1], **Fei Huang**[2], **Wenyu Du**[1], **Luo Si**[2], **Yongbin Li**[2 †]

[1]香港大学 [2]达摩院，阿里巴巴集团 [3]深圳先进技术研究院，中国科学院[4]香港中文大学（深圳） [5]粤港澳联合实验室{jl0725,huonan,wenyudu}@connect.hku.hk, ckcheng@cs.hku.hk, bw.qin@siat.ac.cn, machenhao@cuhk.edu.cn,{binyuan.hby,f.huang,luo.si,shuide.lyb}@alibaba-inc.com

摘要

文本到SQL解析任务旨在将自然语言问题转换为可执行的 SQL 查询。近年来，这项任务受到越来越多的关注，因为它能够帮助终端用户在不具备技术背景的情况下，高效地从数据库中提取关键信息。文本到SQL解析面临的一项主要挑战是领域泛化，即如何在未见过的数据库上实现良好的泛化。近期，预训练文本到文本 Transformer 模型 T5 虽然并非专为文本到SQL解析设计，却在面向领域泛化的标准基准测试上取得了最先进的性能。在本研究中，我们探索如何通过专用组件进一步增强预训练 T5 模型，以用于文本到SQL解析。这些组件有望为文本到SQL解析器引入结构归纳偏置，从而提升模型进行（可能涉及多跳的）推理的能力；这对于生成结构丰富的 SQL 语句至关重要。为此，我们提出了一种新架构 **GRAPHIX-T5**，这是一种混合模型，在标准预训练 Transformer 的基础上加入了若干专门设计的图感知层。大量实验和分析证明，GRAPHIX-T5 在四个文本到SQL基准测试上均表现有效：**SPIDER**、**SYN**、**REALISTIC**和**DK**。GRAPHIX-T5 的表现显著优于其他所有基于 T5 的解析器，创下新的最先进水平。值得注意的是，GRAPHIX-T5-large 的精确匹配（EM）准确率和执行准确率（EX）分别比原始 T5-large 高出 5.7% 和 6.6%。其表现甚至比 T5-3B 在 EM 和 EX 上分别高出 1.2% 和 1.5%。

1 引言

关系数据库是许多领域中用户进行决策的重要资源，例如医疗保健领域，
体育和娱乐等领域因大数据时代的到来而频繁出现。对于数据用户而言，通过结构化查询语言例如SQL 从数据库中访问信息十分高效。尽管 SQL 语句有效且高效，但其复杂的性质导致非技术用户需要付出极高的学习成本。因此，文本到SQL（Cai 等，2018；Zelle 和 Mooney，1996；Xu、Liu 和 Song，2017；Yu 等，2018a；Yaghmazadeh 等，2017）旨在将自然语言指令或问题转换为 SQL 查询，因而受到广泛关注。

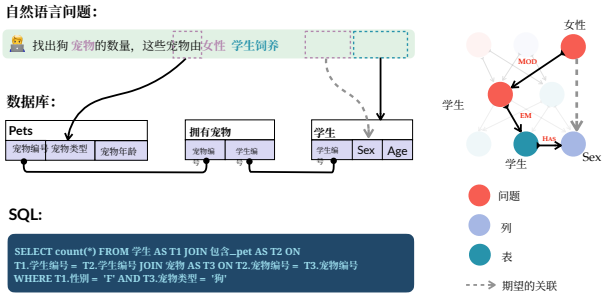


图 1：这是一个跨领域文本到 SQL 挑战的示意图。目标列“性别”与词元“女性”之间的关联对模型来说非常重要，但也极难捕捉，尤其是在缺少领域特定数据或有效规则时。不过，多跳推理路径（女性 $_{\text{MOD}}\longrightarrow$学生 $_{\text{EM}}\longrightarrow$ 学生 $_{\text{包含}}\longrightarrow$ 性别）可以缓解这一难题。

在这项工作中，我们探讨了具有挑战性的跨领域设置：文本到SQL解析器需要实现领域泛化，即 泛化到训练期间未见过的领域。从理论上讲，实现这一目标将有助于构建通用的自然语言接口，让用户能够与任意领域的数据进行交互。实现领域泛化的主要挑战（Wang 等人，2020a；Cao 等人（2021）；Wang 等人，2022；Cai 等人，2021；Hui 等人（2022））在于，生成结构丰富的 SQL 语句需要进行（可能是多跳的）**推理**，即 通过考量多种显式关系（例如，数据库模式中指定的表-列关系）和隐式关系（例如，判断某个短语指的是列还是表），恰当地结合给定数据库的上下文来理解用户问题。图1展示了文本到SQL解析中多跳推理的入门示例，图6则给出了另外两个更详细的案例。

---

* 在阿里巴巴达摩院实习期间完成的工作。

† 通讯作者为Reynold Cheng和Yongbin Li。版权所有 © 2023 年，人工智能促进协会 (www.aaai.org)。保留所有权利。

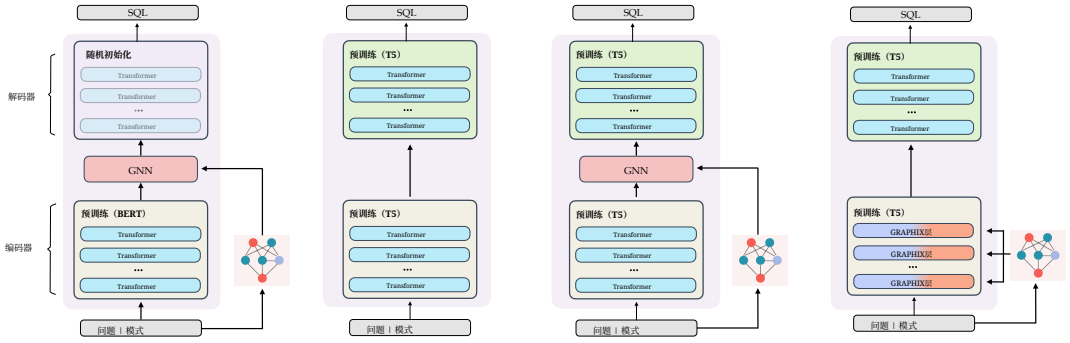


图2：现有方法的图示 **(a) RATSQL** [预训练的 BERT编码器 → 基于图的模块 → 随机初始化的解码器]。**(b) T5** [预训练的 T5编码器 → 预训练的 T5解码器] 以及提出的变体**(c) GNN-T5** [预训练的 T5编码器 → 基于图的模块 → 预训练的 T5解码器] **(d) GRAPHIX-T5** [半预训练的 graphix 模块 → 预训练的 T5解码器]。

从建模角度来看，我们可以沿两个关键维度区分当前的文本到SQL解析器。第一个维度是如何有效地将图形式的关系结构（显式和隐式）融入神经网络，第二个维度是如何最大限度地利用预训练模型（例如T5（Raffel 等人（2020））。这两个维度相互关联，共同构成一系列方法。在这一系列方法的一端，PICARD（Scholak、Schucher 和 Bahdanau，2021）使用原始的预训练 T5 模型，将数据库模式线性化为序列，希望 T5 能够成功捕获其底层关系结构。另一端的 RAT-SQL（Wang 等人，2020a）仅使用预训练编码器（例如，BERT（Devlin等，2019）），并通过专门的关系感知模型显式捕获所需关系。然而，该框架没有利用更强大的基于编码器-解码器的预训练模型，而对关系结构的支持也仅限于此。在这项工作中，我们探索两者结合的方向，将基于编码器-解码器的预训练模型（具体而言是 T5）与关系感知编码深度结合，以更好地实现领域泛化。我们首先发现，在 T5 中间直接添加一个基于关系图的模块，形成“T5编码器 → 基于图的模块 → T5解码器架构”（另见图2(c)，即 GNN-T5），在标准基准测试上的表现并不理想。推测其不足源于中间的基于图的模块破坏了 T5 内部原有的信息流。

为了解决这一问题，我们提出了一种名为 **GRAPHIX-T5** 的新型架构，能够有效地对关系结构信息进行建模，同时保留预训练 T5 强大的上下文编码能力。**首先**，我们设计了一个 GRAPHIX 层，用于同时编码语义信息与结构信息。具体而言，由问题和数据库构成的输入，其隐藏状态通过上下文语义编码进行建模；结构表示则通过关系GNN模块注入每个 Transformer 层。该模块利用消息传递增强多跳推理能力（Fang et al. 2020；Velickovic et al. 2018），以捕获显式和隐式关系。**其次**，我们通过堆叠 GRAPHIX 层并替换原有的 T5 编码器，构建了一个新的编码器。在每个 GRAPHIX 层中，语义模块的参数仍由 T5 初始化，以尽可能保留预训练带来的上下文编码能力。与彼此分离的 GNN-T5（图2.(c)）不同，GRAPHIX-T5（图2.(d)）能够从底层开始就实现语义与结构之间的密集交互。

我们通过实验证明了 GRAPHIX-T5 在多个跨领域文本到 SQL 基准上的有效性，即 SPI- DER 、 SYN 、 DK 和 REALISTIC。在这些数据集上，所提出的模型取得了新的最先进性能，大幅领先于所有现有模型。具体而言，GRAPHIX-T5-large 的表现出人意料地优于原始T5-3B。此外，我们验证了 GRAPHIX-T5 在低资源和组合泛化场景中也能取得显著提升，这显然得益于结构偏置的引入。需要注意的是，尽管本文仅关注文本到SQL解析，我们相信 ***GRAPHIX-T5*** 的通用方法可以扩展到结构化知识落地任务，例如，表格问答（Pasupat 和 Liang，2015）、数据到文本（Nan 等人，2021）和知识库问答（Talmor 和 Berant，2018）。

2 任务表述与符号说明

2.1 任务定义

给定一个自然语言问题 $\mathcal{Q} = \{q_1, ..., q_{|\mathcal{Q}|}\}$ 及其对应的数据库模式 $\mathcal{D} = \langle \mathcal{C}, \mathcal{T} \rangle$，其中 $\mathcal{C} = \{c_1, ..., c_{|\mathcal{C}|}\}$ 和 $\mathcal{T} = \{t_1, ..., t_{|\mathcal{T}|}\}$ 分别表示列和表，$|\mathcal{C}|$ 和 $|\mathcal{T}|$ 分别表示每个数据库中的列数和表数。文本到SQL的目标是生成对应的 SQL 查询 y。

2.2 原始 T5 架构

模型输入 T5 执行文本到 SQL 任务最经典、有效的输入格式是 *PeteShaw*（Shaw 等人（2021）），它统一了自然语言问题 $\mathcal{Q}$ 和

数据库模式 $\mathcal{D}$ 作为一个联合序列，如下所示：

$$x = [q_1, ..., q_{|Q|} \mid \mathcal{D}_{name} \mid t_1 : c_1^{t_1}, ..., c_{|\mathcal{C}|}^{t_1} \mid ... \mid t_{|\mathcal{T}|} : c_1^{t_{|\mathcal{T}|}}, ..., c_{|\mathcal{C}|}^{t_{|\mathcal{T}|}} \mid *], \tag{1}$$

其中 q_i 是问题中的词元，t_j 表示 $\mathcal{D}$中的表，而 $c_k^{t_j}$ 指的是 j^{th} 表中的 k^{th} 列。

$*$ 是数据库中的特殊列标记。$\mathcal{D}_{name}$ 是每个数据库的名称。

编码器-解码器训练机制遵循 Shaw 等人（2021）的做法，T5（Raffel 等人（2020））采用编码器-解码器机制来生成 SQL 语句。首先，双向编码器学习输入 x的隐藏状态 h，然后解码器基于 h 生成 SQL 语句，如下所示：

$$h = \mathbf{Enc}_\Theta(x); y = \mathbf{Dec}_\Upsilon(h), \tag{2}$$

其中，Θ 和 Υ 分别指编码器和解码器的参数，h 连接编码器和解码器。模型使用预训练的 T5 参数进行初始化，并通过以下目标函数进行优化。

$$\max_{\Theta,\Upsilon} \log p_{\Theta,\Upsilon}(y \mid x) = \sum_{i=1}^{|y|} \log p_{\Theta,\Upsilon}(y_i \mid y_{1:i-1}, x), \tag{3}$$

其中，x和 y 分别表示输入词元和输出词元，$|y|$是生成 SQL 的最大长度。

3 提出的方法：GRAPHIX-T5

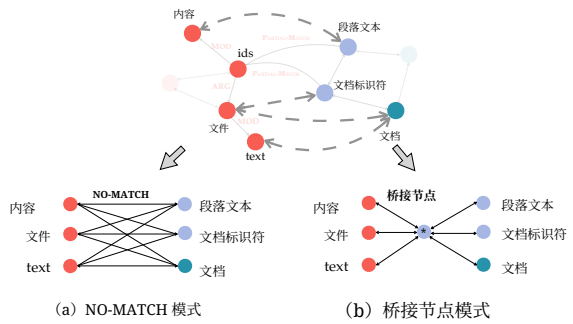
3.1 模型输入

上下文编码 我们继续按照公式（1）所示，将问题和数据库模式一并输入原始 T5，以编码上下文信息。

图构建 联合输入的问题和模式可表示为异构图 $\mathcal{G} = \langle \mathcal{V}, \mathcal{R} \rangle$，由三种类型的节点 $\mathcal{V} = \mathcal{Q} \cup \mathcal{C} \cup \mathcal{T}$和多种类型的关系 $\mathcal{R} = r_1$、.....、$r_{|\mathcal{R}|}$构成，其中每个 r_i表示节点之间的单跳关系，而多跳关系 r^k定义为单跳关系的复合：$r^k = r_1 \circ r_2 \cdots \circ r_I$，如图 1 所示，其中 I表示每个 r^k的长度。受（Wang 等人，2020a；Cao 等人（2021）；Qin 等人（2022b）；Hui 等人（2022））的启发，我们列举了一系列预定义关系来连接节点。关系集合可分为三个主要类别：

- 模式关系：FOREIGN-KEY、PRIMARY-KEY 和 SAME-TABLE 指的是原始 T5 无法从线性输入中获取的特定显式模式关系。
- 模式链接关系：EXACT-MATCH、PARTIAL- MATCH 和 VALUE-MATCH 是问题节点与模式节点之间隐含的链接关系。本文引入了一种新的关系类型 BRIDGE。
- 问题关系：MODIFIER 和 ARGUMENT 是问题中词元之间隐含的依存关系。

NO-MATCH 模式与桥接关系模式以往研究（Cao 等人（2021）；Hui 等人（2022））通过添加称为 NO-MATCH 的虚拟边，表明问题中存在
词元和模式词元，它们应当彼此关联



（a）NO-MATCH 模式　　（b）桥接节点模式

图 3：该图展示了问题中的实体难以与模式项进行字符串匹配的情况。（a）为通过 NO-MATCH 模式解决此问题的策略，即将模式节点与全部标记节点全连接。（b）为我们的解决方案：添加一个桥接节点，以较少的边连接问题节点和模式节点。

但由于现有的字符串匹配规则，无法将它们连接起来。然而，如图 3 所示，NO-MATCH 可能会导致过平滑问题（Chen 等人，2020a），因为它会引入过多噪声邻居来计算注意力分数。假设问题中有 A 个词元，且有 B 个在语义上相关、但未被规则连接的模式项，那么需要连接为 NO-MATCH 的边数为 $A \times B$。相比之下，我们利用特殊词元 * 作为桥接节点，使问题标记节点能够到达所有模式节点，并将边数从 $A \times B$ 大幅减少至 $A + B$。

3.2 GRAPHIX 层

GRAPHIX层旨在将每个 Transformer 块获得的语义信息与关系图神经网络（GNN）块的结构信息相结合。

语义表示隐藏状态的语义表示首先由 Transformer（Vaswani 等，2017）块编码，该块包含两个重要组件：多头自注意力网络（**MHA**）和全连接前馈网络（**FFN**）。在 l^{th} GRAPHIX 层中，隐藏状态表示 $\mathcal{H}_{\mathcal{S}}^l = \left\{h_1^{(l)}, \ldots, h_N^{(l)}\right\}$，$N$是输入的最大长度。MHA 首先将查询矩阵 $\mathbf{Q} \in \mathbb{R}^{m \times d_k}$、键矩阵和数值矩阵 $\mathbf{K} \in \mathbb{R}^{n \times d_k}$、$\mathbf{V} \in \mathbb{R}^{n \times d_v}$ 通过自注意力机制映射为注意力向量，如公式（4）所示

$$\mathbf{Attn}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathbf{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V}, \tag{4}$$

其中，m 是查询向量的数量，n 是键向量或值向量的数量。MHA 使用 h 个头执行自注意力，每个头 i 均由 $\mathbf{W}_i^Q \in \mathbb{R}^{d_m \times d_k}$、$\mathbf{W}_i^K \in \mathbb{R}^{d_m \times d_k}$、$\mathbf{W}_i^V \in \mathbb{R}^{d_m \times d_v}$ 独立参数化，并将输入映射为查询、键值对。通常，$d_k = d_v = d_m/h$ 表示 T5 的 Transformer 块中的参数，d_m 表示 T5 的维度。随后，MHA 计算每个头的注意力输出并将其拼接

它们如下：

$$\mathbf{head_i} = \mathbf{Attn}(\mathbf{QW}_i^Q, \mathbf{KW}_i^K, \mathbf{VW}_i^V), \quad (5)$$

$$\mathbf{MHA}(\mathcal{H}_{\mathcal{S}}^{(l)}) = \mathbf{Concat}\left(\mathbf{head}_1, \cdots, \mathbf{head}_h\right)\mathbf{W}^O, \quad (6)$$

$$\widehat{\mathcal{H}}_{\mathcal{S}}^{(l)} = \mathbf{MHA}(\mathcal{H}_{\mathcal{S}}^{(l)}), \quad (7)$$

其中，$\mathbf{W}^O \in \mathbb{R}^{d_m h \times d_m}$ 是一个可训练参数矩阵。语义隐状态需要通过另一个组件 FFN 获取，即 即，FFN 的应用方式如公式 (8) 所示。

$$\mathbf{FFN}(\widehat{\mathcal{H}}_{\mathcal{S}}^{(l)}) = \mathbf{max}\left(0, \widehat{\mathcal{H}}_{\mathcal{S}}^{(l)}\mathbf{W_1} + \mathbf{b_1}\right)\mathbf{W_2} + \mathbf{b_2}, \quad (8)$$

其中，线性权重矩阵分别表示$\mathbf{W_1} \in \mathbb{R}^{d_m \times d_{ff}}$和$\mathbf{W_2} \in \mathbb{R}^{d_{ff} \times d_m}$。实验发现，较大的 d_{ff}更理想，通常设为 $d_{ff} = 4d_m$。最终，经过层归一化和残差连接后，得到语义隐状态，如下所示

$$\tilde{\mathcal{H}}_{\mathcal{S}}^{(l)} = \mathbf{LayerNorm}(\widehat{\mathcal{H}}_{\mathcal{S}}^{(l)} + \mathbf{FFN}(\widehat{\mathcal{H}}_{\mathcal{S}}^{(l)})), \quad (9)$$

结构表示 在每个 GRAPHIX 层中，结构表示由关系图注意力网络（RGAT）（Wang等人，2020b）在预定义的问题-模式异构图上生成。具体而言，给定 i^{th} 节点的初始节点嵌入[1] e_i^{init}，以及通过特定类型关系与其相连的 j^{th} 个邻居 e_j^{init}，可通过以下方式计算：

$$\vec{\alpha}_{ij} = \frac{e_i^{init}\widetilde{\mathbf{W}}_Q\left(e_j^{init}\widetilde{\mathbf{W}}_K + \phi\left(r_{ij}\right)\right)^\top}{\sqrt{d_z}}, \quad (10)$$

$$\alpha_{ij} = \mathbf{softmax}_j\left(\vec{\alpha}_{ij}\right), \quad (11)$$

$$\hat{e}_i^{init} = \sum_{j \in \tilde{\mathcal{N}}_i} \alpha_{ij}\left(e_j^{init}\widetilde{\mathbf{W}}_V + \phi(r_{ij})\right), \quad (12)$$

$$\hat{e}_i^{(l)} = \mathbf{LayerNorm}(e_i^{init} + \hat{e}_i^{init}\widetilde{\mathbf{W}}_O), \quad (13)$$

$$\tilde{e}_i^{(l)} = \mathbf{LayerNorm}(\hat{e}_i^{(l)} + \mathbf{FFN}(\hat{e}_i^{(l)})), \quad (14)$$

随后，将输出节点嵌入收集为 $\tilde{\mathcal{E}}_{\mathcal{G}}^{(l)} = \left\{\tilde{e}_1^{(l)}, \ldots, \tilde{e}_N^{(l)}\right\}$，其中 $\widetilde{\mathbf{W}}_Q$、$\widetilde{\mathbf{W}}_K$、$\widetilde{\mathbf{W}}_V$、$\widetilde{\mathbf{W}}_O \in \mathbb{R}^{d \times d}$是 RGAT 中可训练的参数。$\phi$（$r_{ij}$）是一个映射函数，可生成一个 d维嵌入，用于表示 i^{th} 节点与 j^{th} 节点之间的每种关系。更重要的是，$\mathcal{N}_i$ 表示关系接收域，即 RGAT 在通过消息传递更新每个节点的表示时所考虑的 i^{th} 节点邻居数量。

联合表示 在分别从语义空间和结构空间计算出表示后，l^{th} GRAPHIX层将语义信息与结构信息结合起来，以如下方式实现信息集成：

$$\tilde{\mathcal{H}}_{\mathcal{M}}^{(l)} = \tilde{\mathcal{H}}_{\mathcal{S}}^{(l)} + \tilde{\mathcal{E}}_{\mathcal{G}}^{(l)}, \quad (15)$$

[1]可以采用多种初始化策略。在本工作中，我们使用节点嵌入的语义表示对其进行初始化。

3.3 GRAPHIX-T5

下面我们正式介绍完整的 GRAPHIX-T5 模型。GRAPHIX编码器最后一层的隐藏状态可表示为：

$$h = \mathbf{Enc}_{\Theta,\Psi}\left(x, \mathcal{G}\right), \quad (16)$$

其中，$\mathcal{G}$ 表示问题-模式异构图，Ψ表示 RGAT 的额外参数，这些参数采用随机初始化。为保留预训练语义知识，我们将原始 T5 编码器中的参数 Θ 迁移过来，作为 GRAPHIX 层中语义 Transformer 块的初始参数。

3.4 训练

与原始 T5 类似，我们也采用微调策略。整个训练框架旨在优化以下对数似然。

$$\max_{\Theta,\Upsilon,\Psi} \log p_{\Theta,\Upsilon,\Psi}(y \mid x) = \sum_{i=1}^{|y|} \log p_{\Theta,\Upsilon,\Psi}\left(y_i \mid y_{1:i-1}, x, \mathcal{G}\right). \quad (17)$$

4 实验

4.1 实验设置

数据集与设置 我们在四个具有挑战性的跨领域文本到 SQL 基准测试上进行了大量实验，并采用了两种不同的训练设置。**（1）SPIDER**（Yu et al. 2018b）是一个大规模跨领域文本到 SQL 基准测试，还纳入了 9 个早期经典数据集，例如 Scholar（Iyer et al. 2017）、WikiSQL（Zhong、Xiong 和 Socher 2017）、GeoQuery（Zelle 和 Mooney 1996）等。它包含 8659 个训练样例和 1034 个开发样例，涵盖 138 个领域中的 200 个复杂数据库。测试集不对个人审查开放。**（2）SYN**（Gan et al. 2021a）将通过简单字符串匹配得到的问题词元或模式名称替换为同义词。**（3）DK**（Gan、Chen 和 Purver 2021）要求文本到 SQL 解析器具备领域知识推理能力。**（4）REALISTIC** 会删除并替换问题中对模式项的显式提及，使其更贴近真实场景。此外，我们还在 **SPIDER-SSP**（Shaw 等人 2021）上测试了模型的组合泛化能力，该基准包含 SPIDER 的三个划分：Spider-Length（根据变体长度划分数据集）；Spider-TMCD（目标最大复合差异）和 Spider-Template（根据不同解析模板划分）。最后，我们分别在使用 10%、20%、50% 数据的 **LOW-RESOURCE** 设置下评估 GRAPHIX-T5 的性能。

评估指标 按照（Yu 等，2018b）的做法，我们使用精确匹配（EM）和执行准确率（EX）这两项标准指标来衡量模型的性能。EM 用于评估生成的 SQL 与标准答案 SQL 的相似程度。EX 则反映预测的 SQL 是否有效，以及是否返回用户期望的准确结果。

模型	EM	EX
RAT-SQL + BERT ♡	69.7	-
RAT-SQL + GRAPPA ♡	73.9	-
GAZP + BERT	59.1	59.2
桥接 v2 + BERT	70.0	68.3
NatSQL + GAP	73.7	75.0
SMBOP + GRAPPA	74.7	75.0
LGESQL + ELECTRA ♡	75.1	-
S^2SQL + ELECTRA ♡	76.4	-
T5-large	67.0	69.3
GRAPHIX-T5-large	72.7 (↑ 5.7)	75.9 (↑ 6.6)
T5-large + PICARD ♣	69.1	72.9
GRAPHIX-T5-large + PICARD ♣	76.6 (↑ 7.5)	80.5 (↑ 7.6)
T5-3B	71.5	74.4
GRAPHIX-T5-3B	75.6 (↑ 4.1)	78.2 (↑ 3.8)
T5-3B + PICARD ♣	75.5	79.3
GRAPHIX-T5-3B + PICARD ♣	**77.1** (↑ 1.6)	**81.0** (↑ 1.7)

表 1：SPIDER 开发集上的精确匹配（EM）和执行（EX）准确率（%）。 ♡ 表示模型不会预测 SQL 值。 ♣ 表示模型使用 PICARD 受限解码。↑ 表示绝对提升幅度。

实现细节 我们的代码 [2]主要基于 Hugging Face Transformers 库（Wolf 等人，2020） [3]实现。我们将最大输入长度设为 1024，最大生成长度设为 128，批量大小设为 32。此外，我们采用 Adafactor （ Shazeer 和 Stern，2018）作为主要优化器，并使用线性衰减的 5e-5 学习率。实验过程中，GRAPHIX 层主要注入编码器，以学习更好的表示，从而实现结构泛化。我们在两个主要版本上评估 GRAPHIX-T5 的效果：参数量约为 8 亿的 T5-Large，以及参数量超过 30 亿的 T5-3B。所有实验均在一台 NVIDIA Tesla A100 上进行，该设备可供大多数研究中心使用。

对比方法 我们的模型主要与主流的强基线模型进行比较，例如 GNNSQL（Bogin、Berant 和 Gardner，2019）、RATSQL（Wang 等人，2020a）、GAZP（Zhong 等人，2020）、BRIDEGE（Chen 等人，2020b）、SMBOP（Rubin 和 Berant，2021）、NatSQL（Gan 等人，2021b）、LGESQL（Cao 等人，2021）、S^2SQL（Hui 等人（2022））以及 T5+PICARD（Scholak、Schucher 和 Bahdanau，2021），并在不同的数据集和设置下进行比较。

4.2 总体性能

SPIDER 上的结果 表 1 展示了 GRAPHIX-T5 与其他具有竞争力的基线模型在官方 SPIDER 基准上的表现。首先，我们证明，配备受限解码模块 PICARD（Scholak、Schucher 和 Bahdanau，2021）的 GRAPHIX-T5-3B，在这一颇具挑战性的跨领域文本到SQL基准测试中达到了最先进水平。此外，显而易见，GRAPHIX-T5 在大规模和 3B 规模下均大幅领先于原始 T5，优势显著。这表明，GRAPHIX

[2] https://github.com/AlibabaResearch/DAMO-ConvAI/tree/main/graphix [3]https://huggingface.co/

模型	SYN	DK	REALISTIC
GNN	23.6	26.0	-
IRNET	28.4	33.1	-
RAT-SQL	33.6	35.8	-
RAT-SQL +BERT	48.2	40.9	58.1
RAT-SQL +Grappa	49.1	38.5	59.3
LGESQL + ELECTRA	64.6	48.4	69.2
T5-large	53.6	40.0	58.5
GRAPHIX-T5-large	61.1 (↑ 7.5)	48.6 (↑ 8.6)	67.3 (↑ 8.8)
T5-3B	58.0	46.9	62.0
GRAPHIX-T5-3B	**66.9** (↑ 8.9)	**51.2** (↑ 4.3)	**72.4** (↑ 10.4)

表 2：SYN、DK 和 REALISTIC 基准测试上的精确匹配（EM）准确率（%）。

模型	模板	长度	TMCD
T5-base	59.3	49.0	60.9
T5-3B	64.8	56.7	69.6
NQG-T5-3B	64.7	56.7	69.5
GRAPHIX-T5-3B	**70.1** (↑ 5.4)	**60.6** (↑ 3.9)	**73.8** (↑ 4.3)

表 3：组合数据集 SPIDER-SSP 上的精确匹配（EM）准确率（%）。

层的结构泛化能力对于 T5 这样的文本到文本预训练语言模型执行文本到 SQL 任务至关重要。

更具挑战性设置下的零样本结果 如表 2 所示，我们进一步展示了 GRAPHIX-T5 在 SYN、DK 和 REAL-ISTIC 上面对更具挑战性且更贴近现实的评估时的稳健性，且无需额外训练。首先，结果表明，GRAPHIX-T5-3B 在这三个数据集上的表现均优于其他基线模型。此外，我们观察到，GRAPHIX-T5-large 和 GRAPHIX-T5-3B 的表现分别明显优于原始 T5-large 和 T5-3B。这表明，在处理来自现实场景、更加灵活复杂的文本到 SQL 问题时，原始 T5 对结构推理的需求很大。而 GRAPHIX 可以缓解这一问题。

组合泛化结果如表3所示，在 SPIDER-SSP 上，Shaw 等人（2021）提出的基于语法的归纳式 T5 模型 NQG-T5 并不明显优于原始 T5，这表明自然语言的语法无助于增强 T5 的组合生成能力。然而，GRAPHIX-T5 通过有效融合结构信息，帮助 T5 获得 SQL 知识，并使其更不易受到这些修改的影响。

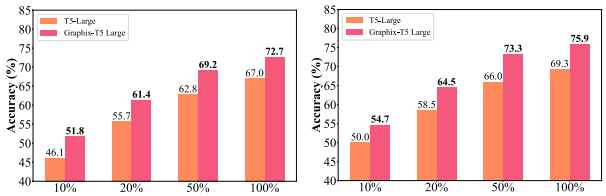


图 4：SPIDER 低资源设置下的精确匹配（EM）（左）和执行（EX）（右）准确率（%）。

模型	SPIDER					SYN					DK					REALISTIC				
	easy	中等	hard	额外	all	easy	中等	hard	额外	all	easy	中等	hard	额外	all	easy	中等	hard	额外	all
T5-large	85.5	70.9	55.2	41.6	67.0	69.0	56.8	46.3	30.2	53.6	**64.1**	44.3	22.9	18.1	40.0	79.8	68.0	44.4	28.9	58.5
GRAPHIX-T5-large	**89.9**	**78.7**	**59.8**	**44.0**	**72.6**	**75.8**	**67.5**	**50.6**	**33.1**	**61.1**	63.6	**54.5**	**33.8**	**29.5**	**48.6**	**88.1**	**77.3**	**50.5**	**40.2**	**67.3**
T5-3B	89.5	78.3	58.6	40.4	71.6	74.2	64.5	48.0	27.8	58.0	**69.9**	53.5	24.3	24.8	46.9	85.3	73.4	46.5	27.8	62.0
GRAPHIX-T5-3B	**91.9**	**81.6**	**61.5**	**50.0**	**75.6**	**80.6**	**73.1**	**52.9**	**44.6**	**66.9**	69.1	**55.3**	**39.2**	**31.4**	**51.2**	**93.6**	**85.7**	**52.5**	**41.2**	**72.4**

表 4：精确匹配 （EM）准确率：根据四个基准测试中推理数据的难度等级进行划分。

模型	EM	EX
（a）RAT-SQL + BERT	69.7	-
（b）T5-large	67.0	69.3
（c）GNN-T5-large	51.6	54.5
(d) GRAPHIX-T5-large		
使用 BRIDGE 模式	**72.7**	**75.9**
使用 NO-MATCH 模式	71.1	74.2
使用 DOUBLE-GRAPH	72.0	74.7

表5：针对跨领域文本到SQL语句的变体 GNN + PLM 策略进行的消融研究，对应图2；(a) 是 RAT-SQL，(b) 是原始 T5，(c) 是 GNN-T5，(d) 是 GRAPHIX。

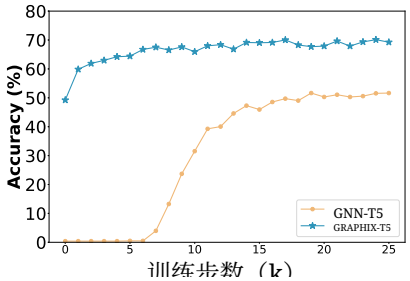


图5：GRAPHIX-T5 和 GNN-T5 在 SPIDER 上收敛期间验证集的性能。可以清楚地看出，由于灾难性遗忘，GNN-T5 的性能极不理想。

低资源设置下的结果图4记录了 GRAPHIX-T5-large 和 T5-large 在不同低资源设置下的表现。结果显示：1）在每种低资源设置下，GRAPHIX-T5-large 的表现都显著优于原始 T5-large。这表明，人类构建的结构知识能够弥补低资源数据导致的学习不足（Ye 等人，2022）；2）值得注意的是，即使只使用**50%** 的数据，GRAPHIX-T5-large 的表现也明显优于使用**100%** 数据训练的原始 T5-large。这进一步证实了 GRAPHIX-T5 在低数据资源条件下训练的优势。

复杂查询上的结果如表4所示，为了更好地理解性能提升，我们还比较了 GRAPHIX-T5 与原始 T5 在 SPIDER 官方划分的四个 SQL 难度等级上的更精确性能结果。我们发现，GRAPHIX-T5 更擅长处理难度更高的文本到 SQL 案例，如**困难**和**极难**示例所示，这表明结构偏置训练有助于文本到文本预训练语言模型对复杂场景进行推理。

4.3 消融研究

如表5所示，为了更好地验证 GRAPHIX-T5 各组件的作用，我们在大型版本上进行了消融研究，旨在回答以下问题。

[1] B 的效果有多好? 采用 BRIDGE 模式的 GRAPHIX-T5-large 比采用 NO-MATCH 模式性能更好。这表明，NO-MATCH 模式会大幅增加噪声邻居的数量，从而提高出现过平滑问题的风险（Chen 等，2020a）。

[2] 能否将 GRAPHIX 融入解码器? DOUBLE-GRAPH 表示 GRAPHIX-T5 同时将 GRAPHIX 层融入编码器和解码器。结果表明，将 GRAPHIX 层添加到解码器并不会带来任何改进。由于解码器是一种自回归模型，在生成当前词元时只考虑历史词元。然而，GRAPHIX-T5 可以通过全局链接预测未来词元的信息，这可能会破坏这一特性，进而对解码器产生负面影响。因此，我们认为最佳策略是仅将 GRAPHIX 层融入编码器。

[3] G 是否RAPHIX 优于其他架构变体? 呼应图2，我们使用 SPIDER 上的预训练语言模型，对4类模型的性能进行了评估。根据表5（c），与 GRAPHIX-T5 相比，GNN-T5 的性能下降了约20%，证明 GNN-T5 的训练策略并不有效。此外，我们注意到，这种受损的 GNN-T5 在训练过程中出现了灾难性遗忘问题（French 1999）。如图5所示，由于 GNN-T5 的准确率在最初几千个训练步数中一直为0，因此显然 T5 的全部预训练知识都被遗忘了。收敛后，GNN-T5 的性能显著低于 GRAPHIX-T5，表明它只利用了 T5 中一小部分语义信息。相比之下，GRAPHIX-T5 在最初的1000个训练步数内即可达到近50%的准确率，并且收敛后比 GNN-T5 提高了20%以上，这验证了 GRAPHIX-T5 能够避免灾难性遗忘并增强泛化能力的优势。

4.4 案例研究

为定性说明 GRAPHIX 的有效性，图6展示了两个从 SYN 中随机抽取的示例。图6比较了原始T5-3B 和 GRAPHIX-T5-3B 预测的 SQL 语句。我们可以观察到，即使在困难场景下，GRAPHIX 也能生成正确的

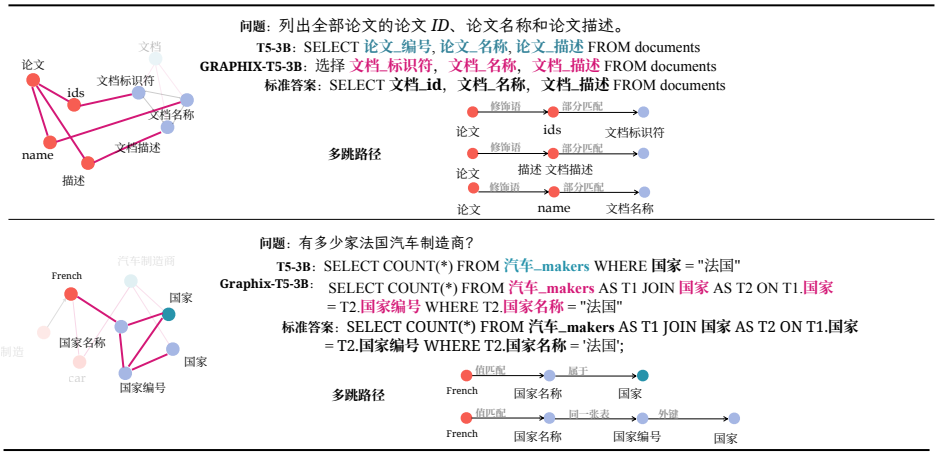


图6：案例研究：两个随机抽取的示例　自 SYN。结果表明，多跳推理有助于 GRAPHIX-T5 生成更多语义含义和数据库模式结构的SQL

SQL 语句。这是因为，即便重叠的关键词很少，GRAPHIX-T5 也能准确识别相应的列或表对象，并通过多跳推理和结构锚定生成高质量的 SQL 语句。例如，在第一个案例中，原始T5-3B 选错了列 paper id、paper name 和 paper description，而这些列甚至没有出现在表文档中。这表明，面对具有挑战性的文本到SQL任务时，原始T5-3B 缺乏结构锚定能力，因而无法找到目标模式元素。相比之下，GRAPHIX-T5-3B 能够通过图6所示的多跳路径，将问题实体对应到正确的列名称。在第二个案例中，原始T5-3B 错误地将国家识别为目标列；然而，“France”只出现在表 countries 的 countryname 列中。这说明 T5-3B 只能生成语义上有效的 SQL 语句，却未能考虑真实的数据库结构。相反，GRAPHIX-T5 在训练过程中成功融合语义信息与结构信息，因此能够根据问题和数据库生成真正有效的 SQL 语句。

5 相关工作

跨领域文本到SQL解析器的基本原理是构建一个编码器来学习问题和模式的表示，同时使用解码器根据编码器中学到的信息生成 SQL 语句（Qin等，2022a）。具体而言，IR NET（Guo等，2019）提出设计一个编码器，通过基于注意力的双向长短期记忆网络分别学习问题和模式的表示，并通过解码器利用编码后的中间表示预测 SQL 语句。后来，基于图的编码器在文本到SQL任务中的有效性得到了成功验证。例如，一些工作（Bogin、Berant 和 Gardner，2019；Chen 等，2021）构建模式图并增强输入的表示。RATSQL（Wang 等人，2020a）、SDSQL（Hui等，2021b）、LGESQL（Cao 等人（2021））、S^2SQL（Hui 等人（2022））进一步改进了结构推理，通过对模式与问题之间的关系进行建模。R^2SQL（Hui等，2021a）、SCORE（Yu等，2021）和 STAR（Cai等，2022）增强了上下文相关的文本到SQL解析中的结构推理。这些工作由预训练语言模型独立构建语义特征，随后由基于图的模块注入结构信息。然而，这种训练策略仅对基于编码器的预训练语言模型有效（即，BERT（Devlin等，2019）、ELECTRA（Clark等，2020）、等）。

近年来，文本到文本预训练语言模型 T5 已被证明在文本到SQL任务中表现有效（Shaw 等人（2021）；Qin 等人，2022c）。此外，Scholak、Schucher 和 Bahdanau（2021）设计了一种受限解码流程，即 PICARD，用于在束搜索阶段检测并拒绝错误词元。Xie 等人（2022）进一步通过多任务学习，将其他结构化知识落地任务中的知识注入 T5，以提升其在文本到SQL任务中的表现。尽管这些方法有效，但在缺乏显式和隐式结构信息的情况下，它们仍难以在更具挑战性和复杂的场景中生成 SQL 语句。而 GRAPHIX-T5 通过在编码器中进行论元图表示学习，能够克服这一问题。与此同时，RASAT（Qi 等人，2022）也尝试通过将边嵌入添加到多头自注意力中，为 T5 提供结构信息；而我们则保留预训练变换器的完整性，以最大限度地利用其先验语义知识，从而获得更好的性能。

6 结论

在本文中，我们提出了一种有效的架构，在增强 T5 结构编码能力的同时，完整保留了预训练 T5 强大的上下文编码能力。为实现这一目标，我们设计了一种图感知半预训练文本到文本预训练语言模型，即 GRAPHIX-T5，以增强其在具有挑战性的文本到SQL任务中的多跳推理能力。大量实验结果证明了 GRAPHIX-T5 的有效性，并表明结构信息对于当前文本到文本预训练语言模型处理复杂的文本到SQL任务至关重要。

致谢

我们感谢于涛博士和谢天宝对我们在 SPIDER排行榜上的工作进行评估。我们感谢王柏林博士和李博文博士提出的宝贵建议。Reynold Cheng、Jinyang Li、Nan Huo 和 Wenyu Du 获得了香港大学（项目编号：104006830）、粤港澳联合实验室计划2020（项目编号：2020B1212030009）以及创新翼二期研究基金的资助。Jinyang Li 还获得了香港大学校长博士生奖学金计划和阿里巴巴集团通过阿里巴巴研究实习生计划提供的资助。Chenhao Ma 获得了深圳市科技创新委员会科技计划项目资助，项目编号为 ZDSYS20211021111415025。

参考文献

Bogin, B.; Berant, J.; and Gardner, M. 2019. 使用图神经网络表示模式结构以进行文本到SQL解析。发表于*ACL*论文集。Cai, R.; Xu, B.; Zhang, Z.; Yang, X.; Li, Z.; and Liang, Z. 2018. 将自然语言转换为数据库查询的编码器-解码器框架。发表于*IJCAI*论文集。Cai, R.; Yuan, J.; Xu, B.; and Hao, Z. 2021. SADGA：面向文本到SQL的结构感知双图聚合网络。发表于*NeurIPS*论文集。Cai, Z.; Li, X.; Hui, B.; Yang, M.; Li, B.; Li, B.; Cao, Z.; Li, W.; Huang, F.; Si, L.; and Li, Y. 2022. STAR：用于上下文相关的文本到SQL解析的SQL引导预训练。发表于*EMNLP Findings*论文集。Cao, R.; Chen, L.; Chen, Z.; Zhao, Y.; Zhu, S.; and Yu, K. 2021. LGESQL：融合混合局部与非局部关系的线图增强文本到SQL模型。发表于*ACL*论文集。Chen, D.; Lin, Y.; Li, W.; Li, P.; Zhou, J.; and Sun, X. 2020a. 从拓扑视角测量并缓解图神经网络的过平滑问题。发表于*AAAI*论文集。Chen, X.; Meng, F.; Li, P.; Chen, F.; Xu, S.; Xu, B.; and Zhou, J. 2020b. 弥合知识引导对话生成中先验知识选择与后验知识选择之间的差距。发表于*EMNLP*论文集。Chen, Z.; Chen, L.; Zhao, Y.; Cao, R.; Xu, Z.; Zhu, S.; and Yu, K. 2021. ShadowGNN：用于文本到SQL解析器的图投影神经网络。发表于*NAACL*论文集。Clark, K.; Luong, M.; Le, Q. V.; and Manning, C. D. 2020. ELECTRA：将文本编码器作为判别器而非生成器进行预训练。发表于*ICLR*论文集。Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT：面向语言理解的深度双向Transformer预训练。发表于*NAACL*论文集。Fang, Y.; Sun, S.; Gan, Z.; Pillai, R.; Wang, S.; and Liu, J. 2020. 用于多跳问答的层次图网络。发表于*EMNLP*论文集。French, R. M. 1999. 连接主义网络中的灾难性遗忘。《认知科学趋势》。Gan, Y.; Chen, X.; Huang, Q.; Purver, M.; Woodward, J. R.; Xie, J.; and Huang, P. 2021a. 提升文本到SQL模型对同义词替换的鲁棒性。发表于*ACL* 论文集。Gan, Y.; Chen, X.; and Purver, M. 2021. 探索跨领域文本到SQL泛化中尚未充分研究的局限性。发表于*EMNLP* 论文集。Gan, Y.; Chen, X.; Xie, J.; Purver, M.; Woodward, J. R.; Drake, J.; and Zhang, Q. 2021b. Natural SQL：让人们更容易从自然语言规范中推断 SQL。发表于*EMNLP Findings* 论文集。Guo, J.; Zhan, Z.; Gao, Y.; Xiao, Y.; Lou, J.-G.; Liu, T.; and Zhang, D. 2019. 借助中间表示实现跨领域数据库中的复杂文本到SQL。发表于*ACL* 论文集。Hui, B.; Geng, R.; Ren, Q.; Li, B.; Li, Y.; Sun, J.; Huang, F.; Si, L.; Zhu, P.; and Zhu, X. 2021a. 用于跨领域上下文相关语义解析的动态混合关系网络。发表于*AAAI* 论文集。Hui, B.; Geng, R.; Wang, L.; Qin, B.; Li, Y.; Li, B.; Sun, J.; and Li, Y. 2022. S^2SQL：将句法注入问题-模式交互图编码器，以用于文本到 SQL 解析器。发表于*ACL Findings* 论文集。Hui, B.; Shi, X.; Geng, R.; Li, B.; Li, Y.; Sun, J.; and Zhu, X. 2021b. 通过模式依赖学习改进文本到SQL。发表于*arXiv:2103.04399*。Iyer, S.; Konstas, I.; Cheung, A.; Krishnamurthy, J.; and Zettlemoyer, L. 2017. 从用户反馈中学习神经语义解析器。发表于*ACL* 论文集。Nan, L.; Radev, D.; Zhang, R.; Rau, A.; Sivaprasad, A.; Hsieh, C.; Tang, X.; Vyas, A.; Verma, N.; Krishna, P.; Liu, Y.; Irwanto, N.; Pan, J.; Rahman, F.; Zaidi, A.; Mutuma, M.; Tarabar, Y.; Gupta, A.; Yu, T.; Tan, Y. C.; Lin, X. V.; Xiong, C.; Socher, R.; and Rajani, N. F. 2021. DART：开放领域结构化数据记录到文本生成。发表于*NAACL* 论文集。Pasupat, P.; and Liang, P. 2015. 半结构化表格上的组合式语义解析。发表于*ACL* 论文集。Qi, J.; Tang, J.; He, Z.; Wan, X.; Cheng, Y.; Zhou, C.; Wang, X.; Zhang, Q.; and Lin, Z. 2022. RASAT：将关系结构融入预训练序列到序列模型以实现文本到SQL。发表于*EMNLP* 论文集。Qin, B.; Hui, B.; Wang, L.; Yang, M.; Li, J.; Li, B.; Geng, R.; Cao, R.; Sun, J.; Si, L.; Huang, F.; and Li, Y. 2022a. 文本到SQL解析综述：概念、方法与未来方向。发表于*arXiv:2208.13629*。Qin, B.; Wang, L.; Hui, B.; Geng, R.; Cao, Z.; Yang, M.; Sun, J.; and Li, Y. 2022b. 面向表格语义解析的链接增强预训练。发表于*arXiv:2111.09486*。Qin, B.; Wang, L.; Hui, B.; Li, B.; Wei, X.; Li, B.; Huang, F.; Si, L.; Yang, M.; and Li, Y. 2022c. SUN：探索文本到SQL解析器中的内在不确定性。发表于*COLING* 论文集。Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; and Liu, P. J. 2020. 探索统一文本到文本 Transformer 的迁移学习极限。机器学习研究杂志。

Rubin, O.; and Berant, J. 2021. SmBoP：半自回归自底向上的语义解析。发表于*NAACL* 论文集。Scholak, T.; Schucher, N.; and Bahdanau, D. 2021. PICARD：从语言模型中进行约束自回归解码的增量式解析。发表于*EMNLP* 论文集。Shaw, P.; Chang, M.-W.; Pasupat, P.; and Toutanova, K. 2021. 组合泛化与自然语言变异：语义解析方法能否兼顾两者？发表于*ACL* 论文集。Shazeer, N.; and Stern, M. 2018. Adafactor：具有次线性内存开销的自适应学习率。发表于*ICML* 论文集。Talmor, A.; and Berant, J. 2018. 将网络作为回答复杂问题的知识库。发表于*NAACL* 论文集。Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. 注意力机制就是你所需要的一切。发表于*NeurIPS* 论文集。Velickovic, P.; Cucurull, G.; Casanova, A.; Romero, A.; Li`o, P.; and Bengio, Y. 2018. 图注意力网络。发表于*ICLR* 论文集。Wang, B.; Shin, R.; Liu, X.; Polozov, O.; and Richardson, M. 2020a. RAT-SQL：面向文本到SQL 解析器的关系感知模式编码与链接。发表于*ACL* 论文集。Wang, K.; Shen, W.; Yang, Y.; Quan, X.; and Wang, R. 2020b. 用于方面级情感分析的关系图注意力网络。发表于*ACL* 论文集。Wang, L.; Qin, B.; Hui, B.; Li, B.; Yang, M.; Wang, B.; Li, B.; Huang, F.; Si, L.; and Li, Y. 2022. Proto n：从预训练语言模型中探测文本到SQL解析的模式链接信息。发表于*KDD* 论文集。Wolf, T.; Debut, L.; Sanh, V.; Chaumond, J.; Delangue, C.; Moi, A.; Cistac, P.; Rault, T.; Louf, R.; Funtowicz, M.; Davi- son, J.; Shleifer, S.; von Platen, P.; Ma, C.; Jernite, Y.; Plu, J.; Xu, C.; Le Scao, T.; Gugger, S.; Drame, M.; Lhoest, Q.; and Rush, A. 2020. Transformers：自然语言处理领域的最新进展。发表于*EMNLP* 论文集。Xie, T.; Wu, C. H.; Shi, P.; Zhong, R.; Scholak, T.; Yasunaga, M.; Wu, C.-S.; Zhong, M.; Yin, P.; Wang, S. I.; Zhong, V.; Wang, B.; Li, C.; Boyle, C.; Ni, A.; Yao, Z.; Radev, D.; Xiong, C.; Kong, L.; Zhang, R.; Smith, N. A.; Zettlemoyer, L.; and Yu, T. 2022. UnifiedSKG：利用文本到文本语言模型统一结构化知识落地并执行多任务。*arXiv* 预印本。Xu, X.; Liu, C.; and Song, D. 2017. Sqlnet：无需强化学习即可从自然语言生成结构化查询。*arXiv* 预印本。Yaghmazadeh, N.; Wang, Y.; Dillig, I.; and Dillig, T. 2017. SQLizer：从自然语言合成查询。*ACM* 编程语言论文集。Ye, H.; Zhang, N.; Deng, S.; Chen, X.; Chen, H.; Xiong, F.; Chen, X.; and Chen, H. 2022. 面向小样本学习的本体增强提示微调。发表于*WWW '22*：*2022* 年 *ACM Web* 会议，线上活动，里昂，法国，*2022* 年 *4* 月 *25* 日至 *29* 日。

Yu, T.; Li, Z.; Zhang, Z.; Zhang, R.; and Radev, D. 2018a. TypeSQL：基于知识、类型感知的神经文本到SQL生成。载于*NAACL* 论文集。Yu, T.; Zhang, R.; Polozov, A.; Meek, C.; and Awadallah, A., Hassan. 2021. SCoRe：对话语义解析中的上下文表示预训练。载于*ICLR* 论文集。Yu, T.; Zhang, R.; Yang, K.; Yasunaga, M.; Wang, D.; Li, Z.; Ma, J.; Li, I.; Yao, Q.; Roman, S.; Zhang, Z.; and Radev, D. 2018b. Spider：用于复杂跨领域语义解析和文本到SQL任务的大规模人工标注数据集。载于*EMNLP* 论文集。Zelle, J. M.; and Mooney, R. J. 1996. 使用归纳逻辑编程学习解析数据库查询。载于*AAAI* 论文集。Zhong, V.; Lewis, M.; Wang, S. I.; and Zettlemoyer, L. 2020. 面向零样本可执行语义解析的基于落地的适配。载于*EMNLP* 论文集。Zhong, V.; Xiong, C.; and Socher, R. 2017. Seq2SQL：使用强化学习从自然语言生成结构化查询。载于*CoRR abs/1709.00103*。

源 x	目标 y	关系类型	描述
问题	问题	修饰语	y 是 x 的修饰语。
问题	问题	论元	在修饰语之外的句法依存关系中，y 是 x 的源词元。
问题	问题	距离-1	y 是 x 最近的（1跳）邻居。
列	列	外键	y 是 x 的外键。
列	列	同一张表	x 和 y 出现在同一张表中。
列	*	桥接	当 y 是特殊列标记“*”时，x 和 y 相连。
表	列	HAS	列 y 属于表 x。
表	列	主键	列 y 是表 x 的主键。
表	*	桥接	当 y 是特殊列标记“*”时，x 与 y 相连。
问题	表	精确匹配	x 是 y 的一部分，而 y 是整个问题的一个跨度。
问题	表	部分匹配	x 是 y 的一部分，但整个问题并不包含 y。
问题	列	精确匹配	x 是 y 的一部分，而 y 是整个问题中的一个跨度。
问题	列	部分匹配	x 是 y 的一部分，但整个问题中不包含 y。
问题	列	值匹配	x 是列 y 的候选单元格值之一。
问题	*	桥接	当 y 是特殊列标记“*”时，x 和 y 相连。

表6：Th e GRAPHIX-T5 中使用的主要关系类型清单。上述关系均为非对称关系 c.

A 细粒度句法关系

以往的工作（Cao 等人（2021）；Wang 等人，2020a）在构建图时仅将词元之间的距离作为关系，因此无法考虑词元之间确定性的关系。例如，以下两个句子的含义相同：“列出并非来自法国的学生的名称。”；“国籍不是法国的学生的名称是什么？”。这两个句子中 not 与 France 之间的关系应当相同。然而，根据其定义的关系，这种关系分别被表示为两个不同的关系：距离-2 和距离-1，这会导致预训练语言模型学习到错误的关系表示。尽管 Hui 等人（2022）提出将前向关系和后向关系作为问题词元之间的额外抽象关联，但仍难以辨别对文本到SQL有用的、更重要的关系。在本研究中，我们发现，名词以及其他可能表示名词特征的词元可以帮助模型找到对应的数据库项。为实现这一目标，我们手动将依存句法关系聚类为两个新的句法关系类别：**修饰语** 和**论元**。如表6所示，**修饰语** 表示源词元节点的某些属性受到目标词元节点的修饰。例如，在短语“女学生”中，Female 是 Students 的修饰语；在短语“Production Time”中，Production 是词元 Time 的修饰语。所有其他依存句法关系都将标记为**论元**。

B 排行榜结果

配备 PICARD 后，GRAPHIX-T5 在 SPIDER 测试排行榜上以明显优势位居第一，如表7和表8所示。

模型	**Dev**	**Test**
搭配基于编码器的预训练语言模型		
RATSQL + BERT	69.7	65.6
RATSQL + GRAPPA	73.4	69.6
GAZP +BERT	59.1	53.3
桥接 v2 + BERT	70.0	65.0
NatSQL + GAP	73.7	68.7
SMBOP +GRAPPA	74.7	69.7
LGESQL +ELECTRA	75.1	72.0
S^2SQL + ELECTRA	76.4	72.1
使用文本到文本预训练语言模型：T5		
PICARD + T5-3B	75.5	71.9
UnifiedSKG + T5-3B	71.8	-
RASAT +PICARD + T5-3B	75.3	70.9
GRAPHIX +PICARD +T5-3B	**77.1**	**74.0**

表 7：SPIDER 测试集上的精确匹配（精确匹配）结果。

模型	**Dev**	**Test**
使用基于编码器的预训练语言模型		
GAZP +BERT	59.2	53.5
桥接 v2 + BERT	68.3	64.3
NatSQL + GAP	75.0	73.3
SMBOP +GRAPPA	75.0	71.1
搭配文本到文本预训练语言模型：T5		
PICARD + T5-3B	79.3	75.1
UnifiedSKG + T5-3B	74.4	-
RASAT +PICARD + T5-3B	80.5	75.5
GRAPHIX +PICARD +T5-3B	**81.0**	**77.6**

表 8：SPIDER 测试集上的执行（EX）结果。