透过复杂查询的视角看细粒度表检索

Wojciech Kosiuk[1], **XingyuJi**[2], **Yeounoh Chung**[3],**Fatma ¨Ozcan**[3],**MadelonHulsebos**[4*]阿姆斯特丹大学, [2]加州大学伯克利分校, [3]Google, [4]CWI

arXiv:2603.07146v1 [cs.IR] 7 Mar 2026

摘要

以自然语言实现对表和数据库的问答，已成为普及表格数据源洞见的一项关键能力。这些系统首先需要检索与给定自然语言查询相关的数据，为此人们已提出多种方法。在本研究中，我们提出并研究一种表格检索机制，通过设计细粒度的类型化查询分解并感知全局连通性（DCTR），以应对在复杂使用场景中对关系数据库进行开放域问答所带来的挑战。我们从检索复杂度的角度评估这两种机制的有效性，并沿查询复杂度和数据复杂度两个维度进行衡量。我们对行业对齐基准数据集的分析表明，DCTR 能够稳健地处理高度复合查询以及连接稠密的数据库。

1 引言

面对大型异构表集合，开放域问答和文本转 SQL 面临一项根本性的检索挑战：用户很少知道哪些表与问题相关，而且仅凭查询无法判断所需的连接路径（Chen 等人，2025c）。属性名称含糊且术语不一致，导致自然语言查询与模式组织之间出现不匹配（Lei 等人，2020）。多约束查询可能存在多种合理解释（Gomm 等人，2025年），因此难以在 SQL 生成前识别正确的表。检索错误会传递到下游 SQL 生成环节；当不同的数据模型设计允许同一查询对应多种有效 SQL 结构时，问题会进一步加剧（F¨urst 等人，2024年），而且即使可靠地评估查询是否正确仍然颇具挑战（Kim 等人，2024年）。此外，尽管近期方法为表格检索设计了查询分解，单跳检索仍是常见的默认做法，即使用完整查询嵌入进行检索（Chen 等人，2025b；Zou 等人，2025年），这促使我们深入考察单跳检索在复杂组合式查询中的稳健性。

尽管表格检索机制近期取得了进展，单向量表嵌入仍无法表示多约束相关性信号（Weller 等人，2025年），并且当查询语义偏离模式约定时，性能会下降（Chen 等人，2025b；Lei 等人，2020）。行业工作负载使这些挑战更加突出：查询冗长且具有组合性，数据库涵盖数百张语义未对齐的表，而相关数据则分散在许多连接相连表中。尽管现有研究揭示了不同方法和数据集在检索性能上的模式（Ji 等人，2025年），但对于此类复杂检索特征，仍缺乏更深入的见解。

针对复杂检索设置，我们提出基于分解的连通性表检索（DCTR），其 1）将查询分解为不同类型的语义单元，通过多向量嵌入实现细粒度的查询-模式对齐；2）整合全局表连通性感知检索，从语义相似度之外发掘相关表。我们在开放域分析型表格问答的典型行业基准上评估 DCTR，结果表明，与常见的单向量稠密嵌入检索相比，其召回率始终更高。最后，我们从查询和数据维度对检索复杂度进行形式化，并分析查询信噪比、模式异质性及连通性的影响。研究结果表明，在涉及高度组合式查询和大型、高度密集连接的关系数据库的复杂检索设置中，细粒度查询分解与全局连通性感知必不可少。

*联系邮箱：madelon@cwi.nl

2 相关工作

2.1 文本到SQL与模式链接

模式链接以不同粒度将自然语言查询与模式元素对齐。RASL（Eben 等人，2025年）通过将模式分解为语义单元，为数据库引入多阶段检索。Nahid 等人（Nahid 等人，2025年）提出一种上下文感知双向模式检索方法，利用大语言模型引导的选择将查询与表和列相链接。近期研究对显式链接提出质疑：Maamari 等人（Maamari 等人，2024年）认为，高性能大语言模型即使面对干扰性上下文，也能识别相关模式；而 RESDSQL（Li 等人，2023a）则将链接视为一种通过排序增强的编码，并将其与 SQL 生成解耦。这些研究通常假设数据上下文已知且检索范围有限，因此对复杂数据上下文中的开放域场景仅提供了狭隘的视角。

2.2 开放领域文本到SQL中的检索

开放域问答面临意图歧义和异构语料库的挑战。MURRE（Zhang 等人，2024年）通过查询改写进行迭代式多跳检索；Abacus-SQL（Xu 等人，2025年）则以数据增强作为补充；LinkAlign（Wang 等人，2025年）通过多轮查询改写和响应过滤，将模式链接扩展至大型数据库。EnrichIndex（Chen 等人，2025年b）和 Pneuma（Balaka 等人，2025年）展示了利用大语言模型生成的元数据丰富检索索引的价值；GenEdit（Maamari 等人，2025年）则通过专用算子和用户反馈，在 SQL 生成中融入领域知识。其他互补技术包括基于图的分层检索（Zou 等人，2025年）和弱监督（Liang 等人，2025年）。为应对数据库规模庞大的问题，R2D2（Bodensohn 和 Binnig，2024年）检索表片段，而非完整的表。JAR（Chen 等人，2025年c）引入连接感知重排序，以平衡表与查询的相关性和表间兼容性；Boutaleb 等人将其改用于迭代检索。我们以这些原则为基础，但引入全局连通性感知检索，以找出语义相似度之外的相关表，并采用类型化查询分解实现细粒度模式对齐。我们从复杂查询的角度分析其效果，并通过数据库为复杂场景下的性能提供新颖视角。

3 问题设定

开放域表检索 令 $T = \{T_1, \ldots, T_n\}$ 表示关系数据库中的表集合，其中每个表 T_i 由一组列 $C(T_i) = \{c_1, \ldots, c_m\}$构成，这些列之间可能通过外键关系相连。在开放域场景中，自然语言查询q 表达了一项信息需求；为了从主题多样的大型表集合中检索相关表，需要据此推断数据范围。对于 q，相关表的真实标注集合记为 $T^*(q)$。表检索任务旨在返回一个包含前k 个候选表的排序列表 $R_k(q)$，使封顶召回率（CR）（Thakur 等人，2021）最大化：CR@k(q) $= \frac{|R_k(q) \cap T^*(q)|}{\min(k, |T^*(q)|)}$。

检索复杂度 端到端（分析型）问答的实际检索任务通常涉及查询与数据的组合，其复杂程度超出了现有评测所能呈现的范围。为从这一角度进行考察，我们从两个维度刻画检索复杂度：查询复杂度和 数据复杂度。我们将查询复杂度定义为 由查询在语义密度和功能组合方面的表述方式所引发的复杂度 (Gomm 等人，2025年)。我们将数据复杂度定义为 由于数据库规范化和模式规模，数据库中的数据结构所产生的复杂度(Chen 等人，2025c）。这些复杂度维度指导我们分析复杂问答任务中的细粒度检索。

4 细粒度表检索

我们描述了分解与连通性感知的表格检索方法（DCTR），该方法针对复杂检索任务在两个维度上进行了定制：1）用于细粒度模式对齐的功能查询分解，以及2）用于恢复语义相似度范围之外相关表的全局连通性感知检索。

执行查询之前，首先构建数据索引。为了支持在包含数千张表和相应数量列的大型模式上进行可扩展的检索，我们构建了两个稠密索引：一个对表名进行编码，另一个对列名进行编码。通过比较查询元素的嵌入与这些不同模式元素的嵌入来执行检索。

4.1 DCTR 检索流水线

图 1展示了推理阶段的检索流水线，涵盖从组件提取和按组件检索，到连接图构建和基于外键的扩展，以考虑全局连通性。

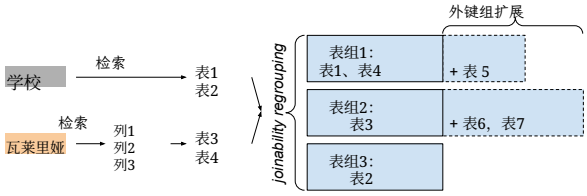


图1：DCTR 概览，包括按组件检索、组形成和外键组扩展。

语义单元中的类型化查询分解在开放域场景中，自然语言查询反映了对不同实体、属性和操作的约束（Gomm 等人，2025年）。为了将这些约束与具体的模式元素和算子对应起来，我们首先执行类型化查询分解步骤，将输入查询 q分割为原子语义单元，为检索和下游任务执行（例如 SQL 生成）奠定基础。分解过程使用具有固定少样本指令提示和解码配置的语言模型完成，以确保跨数据集行为一致（附录A）。DCTR 将每个语义单元分类为三种组件类型之一：

- 表或列名，并在检索过程中与数据库模式进行匹配。
- 查询过滤条件的实体或字面值。
- 合或比较操作（例如平均值、最小值、最大值）。

例如，查询“*2025* 年 *Luka Donˇci´c* 球衣的平均销售额是多少？”可分解为模式组件球衣 和销售额、值组件 Luka Do 和 ，以及聚合组件 平均值 。在检索阶段，仅使用模式组件和值组件，而聚合组件则用于下游任务，例如文本到SQL。

首轮检索 在检索的第一阶段，会针对每个数据库分别独立处理查询的各个组件。对于每个组件，我们会同时对表和列索引执行稠密相似度搜索。检索各组件对应的候选项。此次初始检索会生成未排序的表和列列表。随后，将所有组件和所有数据库的结果合并为一个候选集。通过列索引检索到的列会映射回其所属表，以确保将所有相关表都纳入候选集，并传递至基于图的分组阶段。

模式图与分组构建 正如（Chen 等人，2025c）所讨论的，可连接表与表格检索相关，但不一定会在第一阶段检索中出现；该阶段基于输入查询组件的语义相似性，因此需要沿连接路径进行隐式多跳推理。给定第一阶段检索得到的候选表集合，我们将每个数据库模式表示为一个无向图，其中节点对应表，边表示外键连接关系。

我们将全局模式图限制为与检索到的候选表对应的节点，并且只考虑这些节点之间有效的外键（FK）连接关系。该子图刻画了与查询相关的候选表之间的可连接结构，同时排除了模式中无关的部分。每个连通分量定义一个表组。通过全局外键扩展，我们扩展每个表组，将所有通过外键与组内任一表相连、但未通过语义相似度检索出的表纳入其中，从而恢复多跳连接上下文。

组评分每个表组根据其对提取出的查询组件的覆盖情况进行评分。对于每个组件，DCTR 首先计算其嵌入与组内所有表的嵌入之间的余弦相似度。

我们引入参数*votek* ，用于限制表组评分中纳入的表数量。最终的组覆盖分数通过对所有组件分别取相似度得分最高的*votek* 个表，并将这些得分求和得到：组覆盖分数(T, C) $= \sum_{j=1}^{|C|} \text{top}^{\text{vote}_k}(\{\text{sim}(t_1, \bar{\ } C_j) \ldots \ldots \text{sim}(t_n, \bar{\ } C_j)\})$。

更高的组覆盖分数表明，组内的表共同覆盖了查询的更多组件，因此更有可能包含准确回答该查询所需的信息。在当前公式中，所有组件对组分数的贡献相等。

最终检索 最终检索阶段通过保留每个数据库中得分最高的前n个表组来选择最相关的表。在每个选定的组内，选取个体覆盖度分数最高的 k 张表作为最终候选集。这两个超参数控制检索的广度与粒度，使流程能够根据查询复杂度和模式规模在召回率与精确率之间取得平衡。

5 实验设置

5.1 方法与基线实现

检索实现我们使用 Qdrant 实现检索后端，并建立两个集合。模式表示分别建立索引：表标识符存储在表嵌入中，而列标识符及其语义类型变体存储在列嵌入中。对于查询分解，我们使用 gpt-4.1-mini 和 gemini-2.5-pro 进行文本到SQL评估。

超参数 DCTR 引入了三个关键超参数，我们将在各项实验中进行评估：

- 投票 k：组评分期间，每个组件选取用于投票的组内候选项数量。
- n 组：每个数据库保留用于下游排序的表子组数量。
- 扩展组：在评分之前，是否通过模式图中的连接路径扩展高分子组。

稠密检索基线 我们将本方法与常见的稠密检索基线（Karpukhin 等人，2020）进行比较。给定一个自然语言查询 q，基线将完整查询嵌入为单个向量，并以表名的嵌入表示表。检索时，根据给定的查询嵌入，计算其与每个数据库中各表嵌入的余弦相似度，并选取相似度最高的表。

嵌入模型 所有检索组件均使用稠密文本嵌入来表示查询单元和模式元素。我们评估了以下三种模型，它们在容量与效率之间各有不同的权衡。 **Stella-large** （Zhang 等人，2025年）：一种高容量模型（4亿参数，1024维），专为细粒度语义表示而设计。 **BGE-small** （Xiao 等人，2024年）：一种轻量级通用模型（3300万参数，384维），作为注重效率的基线。 **E5-small** （Wang 等人，2024年）：一种轻量级模型（3300万参数，384维），通过弱监督对比目标进行训练，可用于大规模检索。

5.2 数据集与指标

数据集 我们在通过 TARGET（Ji 等人，2025年）获取的三个文本转 SQL 基准测试上进行评估。这些数据集旨在反映行业检索面临的难点：大型模式、含义模糊的列名、组合式查询和多层嵌套连接。总体而言，它们涵盖企业级设置，兼具复杂的查询与数据（第 5.2 节）。**BEAVER** (Chen 等人，2025a)：来自真实数据仓库的企业级基准测试，具有大型模式（每个数据库平均包含 77.2 张表）和冗长的查询（平均 31.9 个词），涉及多表连接与聚合。**FIBEN** (Sen 等人，2020)：金融领域基准测试，包含针对单个稠密模式（含 152 张表）的 300 个自然语言问题，需要对领域专有术语和复杂嵌套查询进行推理。**BIRD** (Li 等人，2023b)：涵盖 11 个数据库的跨领域基准测试，其模式规模较小但连接紧密（平均每个数据库含 6.8 张表，每表有 1.4 个外键）。我们报告 BEAVER 和 FIBEN 测试集以及 BIRD 验证集上的结果。完整的数据集特征见表 7 （附录 D）。

检索指标 我们使用封顶召回率 RC@k来评估 $k \in \{5, 10, 25\}$ 所有嵌入模型和基准数据集上的表格检索流水线（第 3 节）。第一阶段候选范围固定为 30，并对超参数进行调优，以最大化封顶召回率@k。为考虑大语言模型查询分解带来的随机性，结果基于多次运行报告。稠密基线具有确定性，不存在方差。

复杂度度量 根据第3节中定义的检索复杂度概念，我们研究不同程度的查询复杂度和数据复杂度下的检索性能，因为典型的行业工作负载会使这些属性面临更高要求。对于每个查询，我们记录下述属性，并分析跨数据集和嵌入模型汇总的截断召回率@25。

对于查询复杂度，我们衡量：1）查询的词元长度；2）提取出的功能性查询组件数量。对于数据复杂度，我们衡量通过外键与查询涉及的金标准表相连的表总数，以反映模式规范化程度和连通性。数据集在这些维度上的分布统计数据见附录D。

6 结果

6.1 总体检索性能

表 1-3 报告封顶召回率@k，针对 $k \in \{5, 10, 25\}$，覆盖各数据集、嵌入模型和检索设置。对于较小的检索范围（$k = 5$），较小的投票 k 和 n 组设置表现最佳。随着 k 增大，中等的投票 k和 n 组相较于稠密基线提升了召回率，在 BEAVER 上以及较小嵌入模型上的增益最大。值得注意的是，小型嵌入模型从 DCTR 中获益最多，缩小了与基线中高容量模型的差距。BEAVER 反映了模式规模更大的行业场景，而 BIRD 由较小的孤立数据库组成。结构化的连通性感知检索在类企业场景中尤其有益，有助于从更大规模的模式中检索表。DCTR 设置的良好先验与检索范围成比例：对于较小的 k（例如 5），增大投票$_k$ 或 n_{groups} 并无效果；而对于较大的 k，即 10 或 25，将投票$_k$ 和n_{groups} 分别设为 2 和 5，可带来稳定的召回率增益（附录 B）。

组扩展表现出依赖于数据集的行为。它提升了 BIRD 上的召回率（+3–5%），尤其是在更大的 k时，但会显著降低 FIBEN 和 BEAVER 上的性能（最高达 30%）。这与各基准数据集之间的结构差异相一致（附录 D表 7）。也就是说，BIRD 数据库规模较小（平均 6.8 张表），但连接良好（每表外键数 1.4），因此扩展能有效找回语义上并不相似但相关的表。相比之下，FIBEN 和 BEAVER 包含大得多的模式（每个数据库分别有 152 张和 77.5 张表），从而产生显著更多的候选表，这些候选表可能会被 k 上限评估截断。这些数据集特征表明，在实践中为 k 采用动态取值是可行的，从而充分利用全局外键扩展。

表 1：各数据集和模型的截断 Recall@5（均值 ±标准差）。粗体值表示每个数据集和嵌入模型对应的最佳结果。

设置	模型	BEAVER	FIBEN	BIRD
基线	stella	0.181	0.297	**0.901**
	bge-small	0.154	0.198	0.809
	e5-small	0.108	0.062	0.737
投票$_k=1$, $n_{\text{组}}=1$	stella	**0.231**$_{\pm.004}$	**0.317**$_{\pm.001}$	0.834$_{\pm.001}$
	bge-small	**0.223**$_{\pm.005}$	**0.294**$_{\pm.003}$	**0.826**$_{\pm.002}$
	e5-small	**0.199**$_{\pm.009}$	**0.257**$_{\pm.004}$	**0.817**$_{\pm.007}$

表 2：不同数据集和模型上的截断召回率@10（均值 ±标准差）。粗体数值表示每个数据集和嵌入模型对应的最佳结果。

设置	模型	BEAVER	FIBEN	BIRD
基线	stella	0.277	**0.413**	**0.961**
	bge-small	0.231	0.262	0.909
	e5-small	0.175	0.140	0.850
投票$_k=2$, $n_{\text{组}}=2$	stella	**0.313**$_{\pm.007}$	0.396$_{\pm.006}$	0.899$_{\pm.004}$
	bge-small	**0.326**$_{\pm.002}$	**0.331**$_{\pm.006}$	0.893$_{\pm.003}$
	e5-small	**0.273**$_{\pm.007}$	**0.346**$_{\pm.004}$	0.882$_{\pm.006}$
投票$_k=2$, $n_{\text{groups}}=2$, 扩展组	stella	0.073$_{\pm.014}$	0.202$_{\pm.002}$	0.936$_{\pm.003}$
	bge-small	0.095$_{\pm.001}$	0.091$_{\pm.014}$	**0.942**$_{\pm.001}$
	e5-small	0.109$_{\pm.001}$	0.078$_{\pm.008}$	**0.904**$_{\pm.005}$

6.2 检索复杂度分析

查询复杂度。 图2 展示了检索性能随 查询长度 （以词元数计）的变化。我们发现，对于较长的查询，DCTR 始终优于稠密基线，且差距在超过 40 个词元后进一步扩大。尤其是对于较小的嵌入模型，基线比 DCTR 对查询中的噪声更为敏感，这体现了细粒度查询分解对复杂查询的有效性。

考察每个查询提取的组件数量对检索性能的影响（图 3）时，我们发现，对于轻度组合查询，两种方法的表现相当；但随着组件数量增加，基线的召回率持续下降，而 DCTR 则保持稳定。所有嵌入模型上的基线性能衰减都表明，高容量模型无法解决高度复合查询，但对无关噪声的鲁棒性更强（图 2）。这证实了针对高度组合式查询进行细粒度、组件级检索的必要性。

表 3：不同数据集和模型上的截断召回率@25（均值 ± 标准差）。粗体数值表示每个数据集和嵌入模型的最佳结果。

设置	模型	BEAVER	FIBEN	BIRD
基线	stella	0.413	**0.584**	0.986
	bge-small	0.350	**0.461**	0.976
	e5-small	0.281	0.283	0.949
投票$_k=5$, $n_{\text{组}}=5$	stella	**0.435**$_{\pm.002}$	0.513$_{\pm.007}$	0.971$_{\pm.002}$
	bge-small	**0.429**$_{\pm.003}$	0.448$_{\pm.001}$	0.972$_{\pm.002}$
	e5-small	**0.408**$_{\pm.008}$	**0.516**$_{\pm.003}$	0.965$_{\pm.001}$
投票$_k=5$, $n_{\text{groups}}=5$, 扩展组	stella	0.228$_{\pm.001}$	0.259$_{\pm.001}$	**0.991**$_{\pm.000}$
	bge-small	0.232$_{\pm.002}$	0.127$_{\pm.014}$	**0.988**$_{\pm.002}$
	e5-small	0.228$_{\pm.001}$	0.011$_{\pm.003}$	**0.975**$_{\pm.001}$

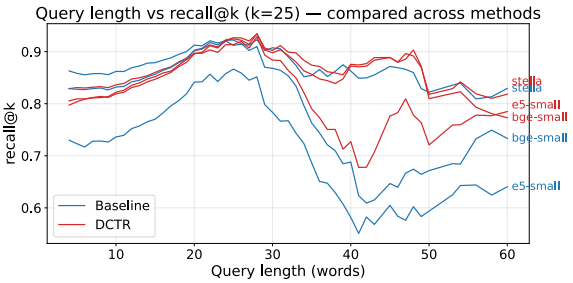


图 2：各数据集汇总后的查询长度与 recall@25。比较三种嵌入模型的结果表明，DCTR 优于基线，尤其是在查询较长时（≥40词元）。

数据复杂度。 我们通过黄金表的连通性评估数据复杂度，具体以它们在数据库中的连接关系数量衡量。图 4 表明，DCTR 在涉及连接紧密的表的查询上表现更佳，这说明连接性感知分组能够提升对可连接表集合的检索效果。

6.3 下游文本到SQL性能

我们评估 DCTR 在下游文本到SQL任务中的效果，以衡量在复杂检索任务中，相较于使用长上下文，检索的实际效用。对于基线，我们将完整的模式作为上下文传递给 Gemini（Chung 等人，2025年）；对于 DCTR，则传递通过 DCTR 检索到的表，并使用*stella*嵌入 $k \in \{5, 10, 25\}$。性能通过执行准确率（EX）衡量。结果见表4。DCTR 在 BEAVER 上将执行准确率提升了 +3%，在 FIBEN 上提升了 +5%，同时在 BIRD 上达到与基线相当的表现。不同数据集的性能表现 k 各不相同，且与检索难度相关：BIRD 在较小的 k 时就达到最高准确率，因为此时召回率已经很高；FIBEN 需要中等程度的 k；而检索最具挑战性的 BEAVER，则受益于更大的 k。

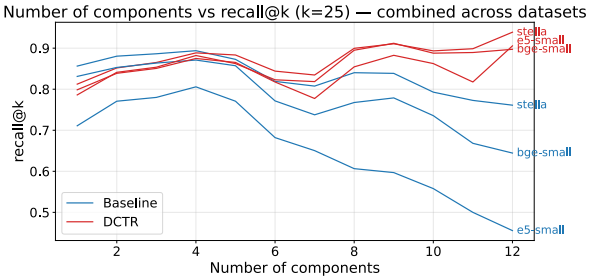


图 3：各数据集上组件数量与召回率@25 的关系。通过三种嵌入模型的比较表明，DCTR 对组合式查询具有稳健性，而单向量基线无论模型容量如何都无法解决组合式查询。

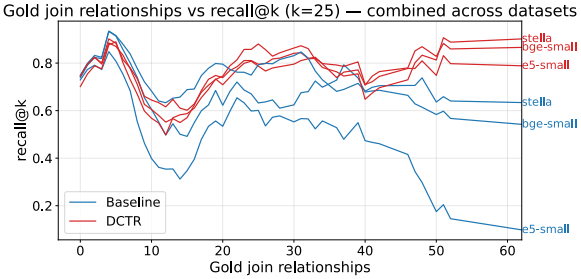


图 4：与黄金表相连的表数量与召回率@25 的关系（跨数据集汇总），针对不同嵌入模型。DCTR 在黄金表连接更密集的查询上提升更大。

表 4：DCTR 相对于长上下文基线的下游 SQL 执行准确率。

设置		BEAVER	FIBEN	BIRD
基线		0.2730	0.1370	0.6630
DCTR	k=5	0.2150	0.0970	**0.6656**
	k=10	0.2400	**0.1870**	0.6630
	k=25	**0.2980**	0.1270	0.6623

7 结论

面向大规模数据库的自然语言接口日益普及，这凸显了检索机制的重要性：它们必须能够适应复杂的查询特征和数据特征。在此背景下，我们提出了基于分解的连通性感知表格检索（DCTR）。我们表明，在具有行业代表性的基准数据集上，与常见的单向量稠密检索基线相比，DCTR 取得了显著提升。从查询复杂度和数据复杂度两个维度分析检索复杂度后发现，带类型的查询分解是处理高度组合式查询的关键，这类查询需要进行细粒度的多表模式对齐；而具备全局连通性感知能力的检索则能在连通性密集的关系型数据库中，发现语义相似表之外的相关表。

参考文献

Muhammad Imam Luthfi Balaka、David Alexander、Qiming Wang、Yue Gong、Adila Krisnadhi 和 Raul Castro Fernandez。2025年。Pneuma：利用大语言模型实现表格数据表示与检索的端到端系统。*arXiv* 预印本 *arXiv:2504.09207*。

Jan-Micha Bodensohn 和 Carsten Binnig。2024年。重新思考数据湖中的表格检索。载于第七届数据管理人工智能技术应用国际研讨会论文集，*aiDM 2024*，圣地亚哥，智利，*2024*年*6*月*14*日，第 2:1–2:5 页。ACM。

Allaa Boutaleb、Bernd Amann、Rafael Angarita 和 Hubert Naacke。通过迭代搜索探索多表检索。载于 *EurIPS 2025*研讨会：表格数据人工智能。

Peter Baile Chen、Fabian Wenz、Yi Zhang、Devin Yang、Justin Choi、Nesime Tatbul、Michael Ca- farella、Cs̆gatay Demiralp 和 Michael Stonebraker。2025a。Beaver：面向文本转 SQL 的企业级基准测试。预印本，arXiv:2409.02038。

Peter Baile Chen、Tomer Wolfson、Michael Ca- farella 和 Dan Roth。2025b。Enrichindex：利用大语言模型离线丰富检索索引。预印本，arXiv:2504.03598。

Peter Baile Chen、Yi Zhang 和 Dan Roth。2025年c。表格检索是否已成为一个已解决的问题？探索感知连接的多表检索。预印本，arXiv:2404.09889。

Yeounoh Chung、Gaurav T. Kakkar、Yu Gan、Bren¨ton Milne 和 Fatma Ozcan。2025年。长上下文是否就是所需的一切？利用大语言模型的扩展上下文进行自然语言转SQL。*VLDB*基金会会刊，18(8):2735–2747。

Jeffrey Eben、Aitzaz Ahmad 和 Stephen Lau。2025年。RASL：面向大型数据库文本到SQL的检索增强模式链接。预印本，arXiv:2507.23104。

Jonathan F¨urst、Catherine Kosten、Farhad Nooralahzadeh、Yi Zhang 和 Kurt Stockinger。2024年。基于真实用户查询评估文本到SQL系统的数据模型鲁棒性。预印本，arXiv:2402.08349。

Daniel Gomm、Cornelius Wolff 和 Madelon Hulse- bos。2025年。我们提出的问题正确吗？关于用于表格数据分析的自然语言查询中的歧义。载于 *EurIPS 2025* 表格数据人工智能研讨会。

Xingyu Ji、Parker Glenn、Aditya G. Parameswaran 和 Madelon Hulsebos。2025年。TARGET：面向生成任务的表格检索基准。预印本，arXiv:2505.11545。

VladimirKarpukhin、Barlas O˘guz、SewonMin、Patrick Lewis、Ledell Wu、Sergey Edunov、Danqi Chen 和 Wen tau Yih。2020。面向开放域问答的稠密段落检索。预印本，arXiv:2004.04906。

Heegyu Kim、Taeyang Jeon、Seunghwan Choi、Seung- taek Choi 和 Hyunsouk Cho。2024年。Flex：面向可靠文本到 SQL 基准的专家级无误执行指标。预印本，arXiv:2409.19014。

Wenqiang Lei、Weixin Wang、Zhixin Ma、Tian Gan、Wei Lu、Min-Yen Kan 和 Tat-Seng Chua。2020。重新审视模式链接在文本到 SQL 中的作用。载于*2020*年自然语言处理经验方法会议论文集（*EMNLP*），第6943–6954页，线上。计算语言学协会。

Haoyang Li、Jing Zhang、Cuiping Li 和 Hong Chen。2023a。Resdsql：解耦文本到 SQL 的模式链接与骨架解析。预印本，arXiv:2302.05965。

Jinyang Li, Binyuan Hui, Ge Qu, Jiaxi Yang, Binhua Li, Bowen Li, Bailin Wang, Bowen Qin, Rongyu Cao, Ruiying Geng, Nan Huo, Xuanhe Zhou, Chen- hao Ma, Guoliang Li, Kevin C. C. Chang, Fei Huang, Reynold Cheng, and Yongbin Li. 2023年b.大语言模型是否已经可以用作数据库接口？面向大规模数据库支撑的文本到SQL任务的大型基准测试。预印本，arXiv:2305.03111。

Hsing-PingLiang, Che-WeiChang, and Yao-Chung Fan. 2025年。通过从部分表格生成问题来改进表格检索。预印本，arXiv:2508.06168。

Karime Maamari, Fadhil Abubaker, Daniel Jaroslaw- icz, and Amine Mhedhbi. 2024年。模式链接已死？推理良好的语言模型时代的文本到SQL。预印本，arXiv:2408.07702。

Karime Maamari、Connor Landy 和 Amine Mhedhbi。2025年。GenEdit：复合算子与持续改进，以应对企业中的文本到SQL。载于 第*11*届创新数据系统研究会议（*CIDR*）论文集，阿姆斯特丹，荷兰。

Md Mahadi Hasan Nahid、Davood Rafiei、Weiwei Zhang 和 Yong Zhang。2025年。重新思考模式链接：一种面向文本到SQL的上下文感知双向检索方法。预印本，arXiv:2510.14296。

¨Jaydeep Sen、Chuan Lei、Abdul Quamar、Fatma Ozcan、Vasilis Efthymiou、Ayushi Dalmia、Greg Stager、Ashish Mittal、Diptikalyan Saha 和 Karthik Sankaranarayanan。2020。Athena++：面向复杂嵌套 SQL 查询的自然语言查询。*VLDB Endow.* 论文集，13(12):2747–2759。

Nandan Thakur、Nils Reimers、Andreas R¨uckl´e、Abhishek Srivastava 和 Iryna Gurevych。2021。Beir:

用于零样本评估信息检索模型的异构基准数据集。发表于 第三十五届神经信息处理系统大会数据集与基准赛道（第二轮）。

Liang Wang、Nan Yang、Xiaolong Huang、Binxing Jiao、Linjun Yang、Daxin Jiang、Rangan Ma- jumder 和 Furu Wei。2024年。基于弱监督对比式预训练的文本嵌入。预印本，arXiv:2212.03533。

Yihan Wang、Peiyu Liu 和 Xin Yang。2025年。Linkalign：面向真实世界大规模多数据库文本到SQL的可扩展模式链接。*arXiv* 预印本 *arXiv:2503.18596*。

OrionWeller、Michael Boratko、Iftekhar Naim 和 Jinhyuk Lee。2025年。基于嵌入的检索的理论局限。预印本，arXiv:2508.21038。

Shitao Xiao、Zheng Liu、Peitian Zhang、Niklas Muennighoff、Defu Lian 和 Jian-Yun Nie。2024年。C-pack：用于通用中文嵌入的打包资源。预印本，arXiv:2309.07597。

KeyanXu、Dingzirui Wang、Xuanliang Zhang、Qingfu Zhu 和 Wanxiang Che。2025年。Abacus-SQL：一个赋能跨领域和开放领域数据库检索的文本到SQL系统。*arXiv* 预印本 *arXiv:2504.09824*。

Dun Zhang、Jiacheng Li、Ziyang Zeng 和 Fulong Wang。2025年。Jasper 和 stella：最先进嵌入模型的蒸馏。预印本，arXiv:2412.19048。

Xuanliang Zhang、Dingzirui Wang、Longxu Dou、Qingfu Zhu 和 Wanxiang Che。2024年。MURRE：通过移除机制实现开放领域文本到SQL的多跳表格检索。*arXiv* 预印本 *arXiv:2402.10666*。

Jiaru Zou、DongqiFu、Sirui Chen、XinruiHe、Zihao Li、Yada Zhu、Jiawei Han 和 Jingrui He。2025年。GTR：用于跨表问答的图表检索增强生成。*arXiv* 预印本 *arXiv:2504.01346*。

附录

查询分解提示

我们使用固定的指令提示来提取与模式相关的查询组件。提示如下：

---目标--- 给定一个自然语言查询和一系列组件类型，从文本中识别出 ↪所有模式 ↪元素组件。

---输出--- 对于识别出的每个组件，提取以下信息：↪- 组件_名称：组件的名称，使用输入↪查询中的确切短语。↪

尽可能保持简短精炼。↪- 组件_描述：简要描述组件的↪属性，以及有助于理解组件在查询中所起作用的任何↪潜在背景信息。↪↪

---模式元素组件描述--- ↪查询中直接↪引用↪的特定数据模式元素（例如表或列名）。思考查询中的哪些部分最可能在可用数据中有↪对应项↪，并说明原因。尽可能简洁，并删去模式元素↪组件周围的↪任何↪约束或筛选短语。↪

注意：排除那些更可能指代特定单元格↪值、而非↪模式元素的短语。仅关注最可能存在匹配模式↪条目的元素。↪

---示例--- {few_shot_示例}

---真实数据--- 输入
查询：{query}

输出：

B 消融实验

我们研究关键超参数（*n groups*、*vote k* 和 *expand groups*）对检索性能的影响。由于不同数据集在数据库数量、模式规模和表间连通性方面存在差异，因此选择超参数对于平衡覆盖率和精确率至关重要。我们重点关注 $k = 25$，因为在此处，分组和投票的影响最为显著，也更容易观察。

结果。 表 5 总结了消融实验结果。在各个数据集上，增加 *ngroups* 和*vote k* 能够持续提升召回率，直至达到饱和点；超过该点后，增益会减小或消失。对于 $k = 25$，通常在 *n groups* 和 *vote k* 取值为 5–6 时可获得最佳性能，而更大的取值则不会带来进一步提升。

在 BEAVER 和 FIBEN 上，启用组扩展会导致所有配置下的性能显著下降。相比之下，BIRD 在若干设置下受益于组扩展，

召回率常接近完美。不同嵌入模型下的趋势一致，这表明超参数的表现基本不受嵌入模型选择的影响，而主要由数据集特征决定。

k 的消融结果=5,10 对于 $k \in 5{,}10$，我们评估了更大的超参数配置（n_{groups},投票$_k$），其中 $k = 5$ 考虑设置 1–3，$k = 10$考虑设置 1–4，并且每种设置都分别在启用和不启用扩展组的情况下进行评估。

表 6中的结果展示了 $k = 5$ 和 $k = 10$的超参数选择所产生的影响，评估仅使用 `stella-large`，因为未观察到各模型之间存在显著差异。

最优的（$n_{\text{groups}}, , vote_k$）配置因数据集而异。对于 BEAVER，在 $k = 5$时（1, 1）表现最佳，在 $k = 10$时（2,2）表现最佳。在 FIBEN 中，（1,1）在 $k = 5$ 和 $k = 10$上均取得最高召回率。对于 BIRD，在 $k = 5$时（3,3）结果最佳。请注意，BIRD 的评估使用了 200个查询的子样本，因此这些最优值与基于完整数据集得出的结果不同。

C k=5,10 的详细逐查询分析结果

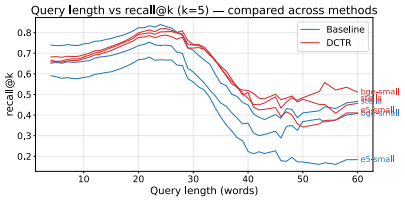
图 [5, 6, 7, 8] 报告了在 $k = 5$ 和 k =10 条件下，针对查询复杂度和数据复杂度的逐查询分析。

除了正文中讨论的结果外，表7 还展示了召回率随黄金表数量变化的情况。不出所料，所需表的数量越多，检索难度越大，召回率也随之持续下降。我们的方法呈现出与稠密基线相似的趋势，这表明需要处理大量多表的情况仍然具有挑战性。

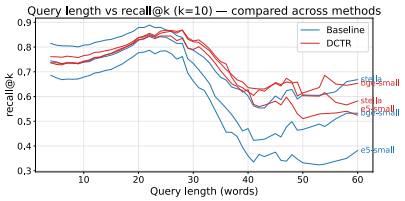
其余分析也重现了正文部分观察到的趋势。在查询复杂度方面，随着查询长度增加以及提取出的组成部分增多，性能有所提升。在数据复杂度方面，对于涉及密集连接表的查询，该方法表现得更为稳健；然而，当必须联合检索大量黄金表时，性能会下降。

D 数据集与查询分布

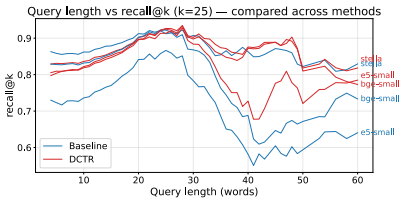
图 9 展示了所评估查询中查询和数据复杂度属性的分布。大多数查询长度少于 40 个词元，包含



(a) 截断召回率@5



(b) 截断召回率@10



(c) 截断召回率@25

图5：合并所有数据集后的查询长度与 recall @k。比较所提方法与基线在三种嵌入模型上的表现。对于所有 k 值，所提方法始终优于基线，尤其是在较长查询（≥40词元）上。

1-6 个提取的组件，需要 1-4 张黄金表，并涉及这些表之间 1-10 个连接关系。这些分布为正文中的逐查询分析提供了背景，并阐明了所评估工作负载的典型结构特征。

此外，表7总结了基准测试特征。BEAVER 和 FIBEN 包含的表数量远多于 BIRD，且每个数据库的平均表数也更高，因此在这些企业级场景中，检索挑战要大得多。相比之下，BIRD 的每个数据库外键数量最多。这一结构特性与我们关注连通性的设计更加契合，也提高了从组扩展中获益的可能性。

数据集(n_{groups}, $vote_k$)		嵌入模型					
		扩展组=False			扩展组=True		
BEAVER	(3,3)	0.3901	0.3869	0.3782	0.2231	0.2338	0.2155
	(4,4)	0.4106	0.4073	0.3933	0.2306	0.2392	0.2392
	(5,5)	0.4332	0.4321	0.4106	0.2284	0.2306	0.2284
	(6,6)	0.4504	0.4332	0.4138	0.2285	0.2349	0.2371
	(7,7)	0.4569	0.4353	0.3966	0.2381	0.2403	0.2371
FIBEN	(3,3)	0.4526	0.3890	0.4091	0.2324	0.1149	0.1227
	(4,4)	0.4769	0.4051	0.4508	0.2524	0.1271	0.1044
	(5,5)	0.5152	0.4491	0.5178	0.2594	0.1366	0.1131
	(6,6)	0.5466	0.4700	0.5535	0.2672	0.1271	0.1262
	(7,7)	0.5648	0.4795	0.5614	0.2715	0.1393	0.1184
BIRD (200 查询)	(3,3)	0.9214	0.9337	0.9509	1.0000	1.0000	0.9951
	(4,4)	0.9582	0.9754	0.9730	1.0000	1.0000	0.9975
	(5,5)	0.9754	0.9705	0.9902	1.0000	1.0000	0.9951
	(6,6)	0.9754	0.9803	0.9926	1.0000	1.0000	0.9877
	(7,7)	0.9754	0.9779	0.9926	1.0000	0.9926	0.9975

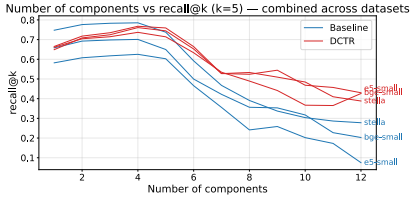
表 5：超参数消融结果。不同 (n_{groups}, $vote_k$) 设置下的封顶召回率@25 s.

数据集	(n_{groups}, $vote_k$)	召回率@5		Recall@10	
		假	True	假	True
BEAVER	(1,1)	0.2333	0.0317	0.2575	0.0830
	(2,2)	0.1993	0.0294	0.3308	0.0873
	(3,3)	0.1880	0.0294	0.3114	0.0862
	(4,4)			0.2931	0.0916
FIBEN	(1,1)	0.4626	0.1737	0.4743	0.1766
	(2,2)	0.3763	0.1694	0.4034	0.2066
	(3,3)	0.3477	0.1604	0.4368	0.1988
	(4,4)			0.4359	0.1918
BIRD (200 查询)	(1,1)	0.8354	0.7641	0.8452	0.9165
	(2,2)	0.8697	0.8010	0.9042	0.9754
	(3,3)	0.8747	0.7740	0.9140	0.9754
	(4,4)			0.9435	0.9779

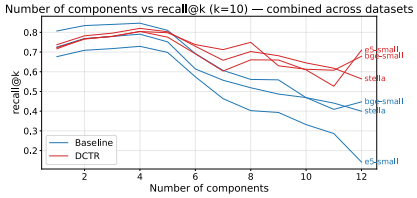
表6：stella-large 嵌入模型的超参数消融结果。报告了不同 (n_{groups}, $vote_k$) 设置下启用（True）或禁用（False）扩展组 时的封顶 Recall@5 和 @10。

指标	BEAVER	BIRD	FIBEN
查询	209	1534	300
唯一数据库	6	11	1
平均查询（词）	31.9	14.6	12.3
平均 SQL（词元）	152.3	25.4	61.9
表	463	75	152
列	4238	798	373
平均每个数据库的表数	77.2	6.8	152.0
平均每个表的列数	9.2	10.6	2.5
平均每个表的外键数	1.1	1.4	1.0

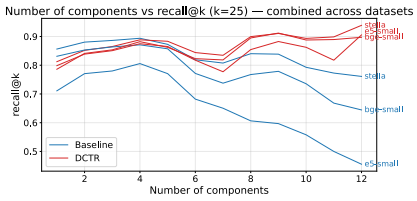
表7：基准数据集特征。



(a) 截断召回率@5

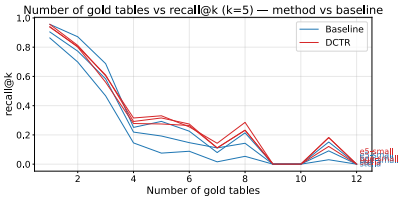


(b) 截断召回率@10

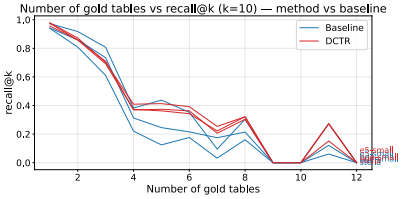


(c) 截断召回率@25

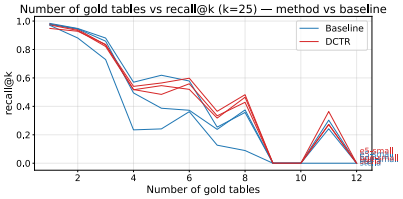
图6：跨数据集汇总的组件数量与 recall@k 的关系。比较所提方法与基线所使用的三种嵌入模型。对于组件更多的查询，我们的方法始终优于基线。随着查询组件数量增加，所提方法始终优于基线。在 k=25（图6c）中，我们的方法在高度组合查询上保持稳定性能，而基线的性能则有所下降。



(a) 截断召回率@5

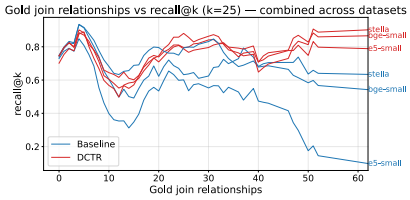


(b) 截断召回率@10

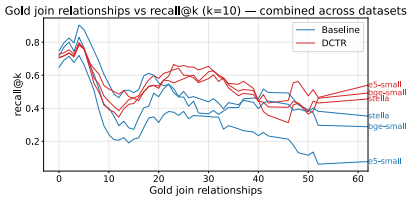


(c) 截断召回率@25

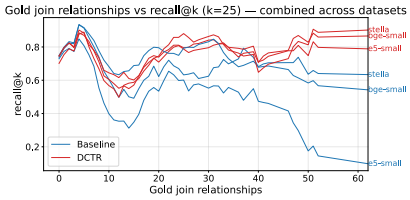
图7：黄金表数量与 recall@k 的关系，跨数据集汇总。比较所提方法与基线所用的三种嵌入模型。随着黄金表数量增加，两种方法的上限召回率均出现下降。所提方法的表现与基线相似，表明在需要检索大量相关表时优势有限。



(a) 截断召回率@5

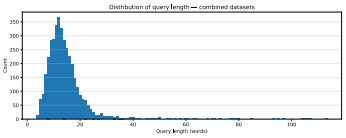


(b) 截断召回率@10

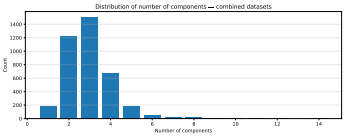


(c) 截断召回率@25

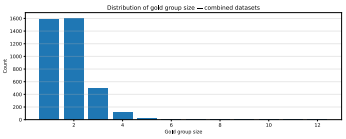
图 8：黄金表连接关系的数量与 recall@k 的关系，汇总了各数据集的结果。比较所提方法与基线采用的三种嵌入模型。当黄金表之间连接密集时，DCTR 能提升查询的性能。



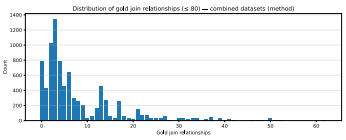
(a) 查询长度分布



(b) 提取的模式/值组件数量分布



(c) 黄金表平均组大小的分布



(d) 黄金表中连接关系数量的分布

图9：分析的属性的分布。所有数据集的结果合并展示。