arXiv:2305.06164v2 [cs.CL] 7 Dec 2023

Conversational Semantic Parsing using Dynamic Context Graphs

ParagJain Mirella Lapata语言、认知与计
算研究所 信息学院，爱丁堡大学 10
Crichton街，爱丁堡 EH89AB
parag.jain@ed.ac.uk mlap@inf.ed.ac.uk

Abstract

本文研究的是在包含数百万实体和数千种关系类型的通用知识图谱（KG）上进行对话式语义解析的任务。我们重点关注能够结合对话历史中的上下文，将用户话语交互式地映射为可执行逻辑形式（例如SPARQL）的模型。我们的核心思想是通过一个动态创建的子图来表示话语及其上下文的信息，即节点数量随话语而变化。我们不将子图视为一个序列，而是利用其底层结构，并通过图神经网络对其进行编码，从而进一步实现对大量（未见过的）节点的表示。实验结果表明，动态上下文建模优于静态方法，在各方面均带来了性能提升（即对于简单问题和复杂问题均如此）。我们的结果还进一步证实，对上下文结构进行建模能够更好地处理话语信息（即处理省略和消解共指），并更好地应对较长的交互。

1 Introduction

通用知识图谱（KG）Vrandeˇci´c 和 Krötzsch,2014）将信息组织成由实体、属性和关系构成的语义网络，使机器能够利用庞大的事实知识库。知识库问答（KBQA）旨在根据自然语言问题从知识图谱中检索答案。KBQA 的一种流行方法（参见 Gu 等，2022 及其中的参考文献）基于语义解析，将问题映射为逻辑形式查询（例如，SPARQL 查询），这些查询在知识图谱上执行后即可返回答案。

<escape>现有研究（例如，Bogin 等，2019；Ravis-hankar 等，2021；Yin 等，2021）主要关注</escape>

1. Mathias Kneissl 的主演是谁？ `SELECT ?xWHERE{ wd:Q3298576 wdt:P161?x .?xwdt:P31 wd:Q502895. }`Rainer Werner Fassbinder、Volker Schlöndorff、Hanna Schygulla2. 这部艺术作品的导演是谁？ `SELECT ?x WHERE { wd:Q3298576wdt:P57 ?x. ?xwdt:P31 wd:Q502895 .}` Reinhard Hauff3.Dubashi 是否有此人担任演员？ `ASK{ wd:Q76025wdt:P161wd:Q24807818 . }`否 4.Rainer Werner Fassbinder 或 Laura Esquivel 担任哪些艺术作品的编剧？ `SELECT ?xWHERE {{ ?x wdt:P58wd:Q44426 . ?xwdt:P31 wd:Q838948 .} UNION{ ?xwdt:P58wd:Q230586 . ?x wdt:P31wd:Q838948 . }}`《美国士兵》、《莉莉·玛莲》、《爱比死更冷》.....Q3298576：Mathias Kneissl，Q76025：Reinhard Hauff，Q24807818：Dubashi，Q44426：Rainer Werner Fassbinder，Q230586：Laura Esquivel，Q838948：艺术作品，Q502895：通用名称，P161：演员，P31：实例，P57：导演，P58：编剧

图 1：来自 SPICE 数据集（Perez-Beltrachini 等人，2023年）的示例交互，其中包含话语、对应的 SPARQL 查询，以及答案，这些答案是在 Wikidata 图引擎上执行查询后返回的。底部模块展示了此次交互中涉及的知识图谱元素（即图节点）。

语义解析。具体而言，我们的研究重点在于构建能够在先前话语的上下文 中，以交互方式将用户话语映射为可执行逻辑形式的系统。图1 展示了一个用户与系统交互的示例，该示例摘自 SPICE （Perez-Beltrachini 等人，2023年），这是一个近期发布的对话语义解析数据集。每次交互都由一系列话语组成，这些话语构成连贯的语篇，并被转换为可执行的语义解析结果（在本例中为 SPARQL 查询）。随着对话的进行，理解每个话语并将其映射到正确的解析结果，需要结合特定的上下文。为了回答话语 2 中的问题，系统需要

要记住，*Mathias Kneissl* 仍然是当前对话的主题，不过，用户不再关心这部电影的主演是谁，而是转而关注导演是谁。用户省略先前提及的信息（例如，通过省略或共指），这也是很自然的现象，而要获得完整的语义解析结果，就必须对这些省略的信息进行补全。

除了处理上下文信息所带来的挑战外，语义解析任务本身还涉及将实体、类型和谓词链接到知识图谱中的元素（例如 *Mathias Kneissl*到 `Q3298576`），而知识图谱的拓扑结构通常十分复杂，包含大量节点。此外，与关系数据库不同，实体的模式并非静态，而是动态实例化的（Gu 等人，2022）。例如，实体类型 人物 可以拥有数百种关系，但其中只有一部分与特定话语相关。因此，为了生成忠实的查询，我们不能依赖记忆，而应利用局部模式实例化。总体而言，缩小实体和关系的范围，对于解析需要复杂推理的话语至关重要（即数值运算符和逻辑运算符作用于实体集合的情况）。

现有研究（Perez-Beltrachini 等人，2023年）通过采用各种简化方法和捷径来应对上述挑战。例如，由于无法对整个知识图谱（KG）进行编码，因此仅提取与当前话语相关的子图，随后将其线性化并作为序列处理。无法通过相邻子图直接获取的实体类型信息，则通过全局查找（本质上是 KG 中所有类型的反向索引）来获取。这种解决方案的计算开销较大，因为实际上每处理一条用户话语都需要执行一次查找，而且扩展性较差（每当 KG 发生变化时，都必须重新创建索引）。

本文提出了一种对话语义解析建模方法，该方法依托于动态上下文图。我们的核心思路是通过动态生成的子图来表示话语及其上下文的信息，其中节点数量因话语而异。此外，我们不将子图视为序列，而是利用其底层结构，并使用图神经网络对其进行编码（Scarselli 等人，2009年；Gori 等人，2005）。为提升泛化能力，我们通过聚合邻居节点的信息来学习隐式节点嵌入，而这些邻居节点的嵌入则通过预训练模型进行初始化（Devlin 等人，2019年）。此外，我们引入了依赖上下文的类型链接方法，该方法基于实体及其周围上下文，进一步有助于类型消歧。

在 SPICE 数据集（Perez-Beltrachini 等人，2023年）上的实验评估表明，动态建模上下文优于静态方法，能够全面提升性能（即在简单问题和需要比较推理或定量推理的复杂问题上均有提升）。我们的结果进一步证实，对上下文结构进行建模更有利于处理话语信息（即处理省略和消解共指），也更适用于包含多个轮次的较长交互。

2 相关工作

以往关于知识库问答（KBQA）语义解析的研究（Guet al.，2022）主要聚焦于将独立的话语映射为逻辑形式查询。为此，人们提出了多种方法，大体上可分为三种建模范式。排序方法首先从知识库中枚举候选查询，然后选择与话语最相似的查询作为语义解析结果（Ravishankaret al.，2021年；Hu etal.，2021年；Bhutaniet al.，2019年）。由粗到精的方法（Dongand Lapata，2018年；Ding etal.，2019年；Ravishankar et al.，2021年）分两个阶段进行语义解析：先预测查询草图，再填补缺失的细节。最后，生成方法（Yin et al.，2021年）先对候选解析结果进行排序，再以话语和检索到的最佳逻辑形式为条件，预测最终解析结果。

我们的任务与对话式文本到 SQL 解析最为相关，这类任务体现在 SParC（Yu 等人，2019年）、CoSQL（Yu 等人，2019年）和 ATIS（Dahl 等人，1994年；Suhr 等人，2018年）等数据集中。SParC 和 CoSQL 涵盖多个领域，并包含用户与系统之间的多轮交互。这些数据集的挑战在于要求模型泛化到未见过的数据库，但对话长度相对较短，数据库规模也相对较小。ATIS 包含与美国航班预订任务相关联的话语及其对应的 SQL 查询；该数据集体现了若干长距离话语现象（Jain 和 Lapata，2021年），但它仅涵盖一个领域，且数据库模式较为简单。

基于图的方法此前已主要应用于语义解析中，用于编码数据库模式，从而使解析器能够从全局角度推理输出查询的结构。

(Bogin et al., 2019).

其他研究（Hui 等人，2022 年）使用关系图网络联合表示数据库模式和问题中的句法依赖信息。在对话语义解析的上下文中，Cai 和 Wan(2020 年）使用图编码器对数据库模式中的元素如何与先前上下文中的信息交互进行建模。类似地，Hui 等人(2021 年）使用带有动态记忆衰减机制的图神经网络，对话进行过程中数据库模式与上下文中的话语之间的交互进行建模。这些方法都对关系数据库的模式进行编码，而关系数据库的规模（例如实体数量和关系类型）远小于大型知识图谱；对于大型知识图谱，将整个图编码到内存中并不可行。

Closest to our work are methods which cast conversational KGQA as a semantic parsing task (Kacupaj et al., 2021; Marion et al., 2021). These approaches build hand-crafted grammars that are not directly executable to a KG engine. Furthermore, they assume the KG can be fully encoded in memory which may not be feasible in real-world settings. Perez-Beltrachini et al. (2023) develop a parser which is executable with a real KG engine (e.g., Blazegraph) but simplify the task by considering only limited conversation context.

3 Problem Formulation

Given a general purpose knowledge graph, such as Wikidata, our task is to map user utterances to formal executable queries, SPARQL in our case. We further assume an *interactive* setting where users converse with the system in natural language and the system responds while taking into account what has already been said (see Figure 1). The system's response is obtained upon *executing* the query against a graph query engine.

令 G 表示底层知识图谱，I 表示一次交互。I 由一系列轮次组成，每轮表示为 $\langle X_t, A_t, Y_t \rangle$，分别表示时间 t的话语-答案-查询三元组（参见图 1 中的方框）。用户话语 X_t 是由词元 $\langle x_1, x_2, \ldots, x_{|X_t|} \rangle$组成的序列，其中 $|X_t|$ 是序列长度，每个 x_i，$i \in [1, |X_t|]$都是一个自然语言词元。查询字符串 Y_t 是由词元 $\langle y_1, y_2, \ldots\ldots, y_{|Y_t|} \rangle$组成的序列，其中 $|Y_t|$ 是序列长度，每个 y_i，$i \in [1, |Y_t|]$要么是 SPARQL 语法词汇中的词元，要么是知识图谱元素表中的词元（例如 SELECT、WHERE），要么是知识图谱元素 $\in G$（例如 Q3298576）。时间 t 的答案 A_t 是对 Y_t 执行操作后得到的结果，执行对象为 G。给定交互历史 $I[:$ 第 t 轮的话语 $t-1]$ 以及当前话语 X_t，我们的目标是生成 Y_t。这需要理解 X_t 在 $X_{:t-1}$、$A_{:t-1}$和 G的上下文中的含义，并学习基于编码后的上下文信息生成 Y_t。

4 Model

我们的建模方法结合了三个组成部分。首先，我们将用户话语中的命名实体与知识图谱中的实体进行关联，并利用这些已链接的实体提取一个子图，作为上下文（第 4.1 节）。第二个组成部分负责在当前及先前提及的命名实体的上下文中进行类型链接（第4.2 节）。最后，我们的语义解析器学习将用户话语映射为 SPARQL 查询（第 4.3 节）。

4.1 Entity Grounding and Disambiguation

我们旨在将用户话语 X_t映射到图 G。由于无法对整个知识图谱进行编码，我们从 G 中提取与当前轮次相关的子图。为此，我们首先使用现成的命名实体识别（NER）系统进行命名实体识别。在我们的实验中，我们使用 AllenNLP（Gardner 等人，2018年）。我们通过高效的字符串匹配来执行命名实体链接[1]（Aho 和 Corasick，1975），而不同于 Perez-Beltrachini 等人（2023年），他们部署了一个 ElasticSearch 服务器来查询倒排索引。

一个字符串可能存在歧义，即对应多个实体。例如，*RainerWernerFassbinder*可以对应电影制作人(Q44426) 和电影(Q33561976)。为了解决歧义并提高召回率，Perez-Beltrachini 等人(2023年) 从不将其限定为单个实体，而是纳入排名靠前的K 匹配项；然而，这会引入噪声并增加计算成本。相反，我们根据实体在训练集中的流行度进行消歧（Shen 等人，2015年），并在第6节比较这两种方法。

接下来Perez-Beltrachini 等人 (2023年)，对于每个已识别的知识图谱实体 e，我们提取三元组 (e, r, o) 以及 (s, r, e)，其中 s和 o表示关系 r的主语或宾语。例如，实体*Dubashi* 对应的三元组为 *(Dubashi, countryof origin, India)*。主语和宾语进一步

[1]我们的实现基于 pyahocorasick https://pypi.org/project/pyahocorasick/。

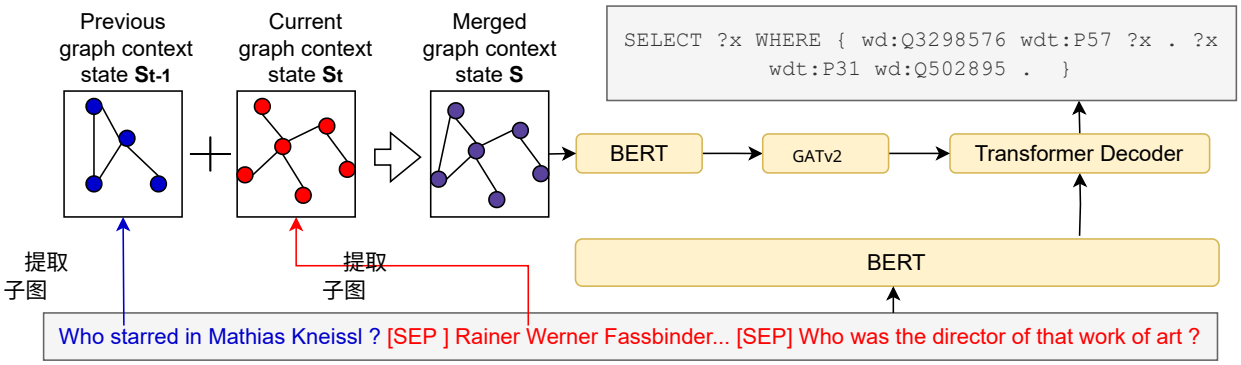

Figure 2: Model architecture. The previous and current utterance are concatenated and their subgraphs are merged and encoded in a graph neural network. The subgraphs represent the entity neighborhood and type linking.

映射为其对应的类型，而非实际实体（即（e，r，o_{type}）和（s_{type}，r，e））。在我们的示例中，*Dubashi* 对应的三元组随后变为（*Dubashi*，原籍国，国家），其中国家 是类型 Q6256。我们将类型化三元组的集合表示为 G_t^{ent}。

4.2 Context-Dependent Type Linking

S 中的实体PARQL 查询具有类型，例如，在图 1中，知识图谱元素Q502895 是类型“通用名称”的占位符。类型实例通常存在于实体的一跳邻域中 G_t^{ent}，但也可能位于更远的跳数之外。Perez-Beltrachini 等人 (2023年) 对 所有 知识图谱类型建立索引，并执行 全局 查找，这种方法计算开销较大，且仅适用于他们所使用的知识图谱。相比之下，我们根据当前 上下文中提及的实体执行类型链接。我们扩展已完成实体链接的实体，以提取包含类型信息的三元组。[2] 由于考虑多跳邻域会产生规模极大的子图，且不利于内存效率，因此我们根据这些子图与用户话语之间的字符串重叠程度进行剪枝，从而大幅减少三元组的数量。剪枝后的图 G_t^{type} 与先前获得的实体图 G_t^{ent} 合并，使得 $G_t = G_t^{ent} \cup G_t^{type}$。

4.3 动态上下文图模型

图 2 展示了我们的动态上下文图模型（简称 DCG）的整体架构。DCG 的输入是一个由以下元素组成的元组：$\langle C_t,\ X_t,\ G_t \rangle$，其中 X_t 表示时间 t时的用户话语，而 G_t 则表示相应的子图。C_t 表示先前的上下文信息，其中包括前一个话语 X_{t-1}，前一个答案$\hat{}A_{t-1}$，以及子图 G_{t-1}。我们使用 G_t 来表示合并后的上下文子图 G_t 和 G_{t-1} 这样的$\hat{}$，使得 $G_t = G_t \cup G_{t-1}$。我们使用图神经网络（GNN，Scarselli 等人，2009年；Gori 等人，2005年）对上下文$\hat{}$子图 G_t 进行编码，并使用 BERT（Devlin 等人，2019年）对用户话语及其话语上下文进行编码。我们的解码器采用 Transformer 网络（Vaswani 等人，2017年），以用户话语编码和相应的图表示为条件。

话语编码器 我们使用 BERT[3] (Devlin 等人，2019年) 来表示前一个话语 X_{t-1}、前一个答案 A_{t-1}和当前话语 X_t 的拼接（见图 2）。为了区分当前上下文和过去的上下文，我们使用 [SEP] 标记。更正式地说，令 $\hat{}X_t$ = [CLS] X_{t-1} [SEP] A_{t-1} [SEP] X_t 表示 BERT 的输入，其中 $X_t = (x_1, x_2 \ldots x_{|X_t|})$、$A_{t-1} = (a_1, a_2 \ldots a_{|A_{t-1}|})$ 和 $X_{t-1} = (x_1, x_2 \ldots x_{|X_{t-1}|})$ 是自然语言词元序列。我们通过 Z_t = BERT(X_t) 获得潜在表示 $Z_t\hat{}$。

图编码器 我们将时间 t 的知识图谱子图 G_t 表示为有向图 $\mathcal{G} = (\mathcal{V}, \mathcal{E})$（下文中，为简化表示，省略时间 t），其中 $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$ 是节点，满足 $v_i \in \{entities,\ relations,\ types\}$ 和 $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$。每个节点 v_i 由一系列自然语言词元组成，满足$v_i = \langle v_{i1}, v_{i2}, \ldots, v_{i|v_i|} \rangle$。我们的知识图谱包含大量不同的节点，但在训练期间不可能将它们全部纳入。为处理未见过的节点

[2]对于实体 实体，我们查询 select ?r1 ?n1 ?t1 ?r2 ?n2 ?t2 where { wd:ent ?r1 ?n1 . ?n1 wdt:P31 ?t1 . OPTIONAL {?n1 ?r2 ?n2 . ?n2 wdt:P31 ?t2} }.

[3]We employ BERT for a fair comparison with prior work. Nonetheless, our model does not have any inherent restrictions that would prevent the use of other pretrained models.

在测试时，我们为节点 v_i获取通用节点表示 h_i^0，其中 $h_i^0 = \text{AVG}(\text{BERT}(v_i))$。换言之，我们通过对 BERT 得到的各个词元编码取平均值，计算编码 h_i^0。我们不创建单独的嵌入矩阵，而是在学习过程中直接更新 BERT 表示，从而能够扩展到大量（未见过的）节点。

图神经网络（GNN）通过聚合其邻居节点的信息来学习节点表示。每个 GNN 层 l以节点表示 $\{h_i^{l-1} \mid i \in [1,n]\}$和边 $\mathcal{E}$作为输入。每一层的输出是一组更新后的节点表示 $\{h_i^l \mid i \in [1,n]\}$。我们使用图注意力网络 v2（GATv2，Brody 等人，2022）来更新节点表示，该网络用一种动态且表达能力更强的变体替代了 GAT 的静态注意力机制（Velickovic 等人，2018年）。令 $\mathcal{N}_i = \{v_j \in \mathcal{V} \mid (j, i) \in \mathcal{E}\}$ 分别表示节点 v_i 的邻居，以及 α_{ij}与节点 h_i 之间的注意力分数。我们将注意力计算为一个单层前馈神经网络，其参数由权重向量 a和权重矩阵 W确定：

$$\alpha_{ij} = \frac{\exp\left(\psi\left(h_i^{l-1}, h_j^{l-1}\right)\right)}{\sum_{k \in \mathcal{N}_i} \exp\left(\psi\left(h_i^{l-1}, h_k^{l-1}\right)\right)} \quad (1)$$

The scoring function ψ is computed as follows:

$$\psi\left(h_i^{l-1}, h_j^{l-1}\right) = a^T \,\text{LeakyReLU}\left(W \cdot [h_i^{l-1} \parallel h_j^{l-1}]\right) \quad (2)$$

其中，$\cdot^T$ 表示转置，$\parallel$ 表示拼接操作。随后，利用与每个节点 i 对应的注意力系数，计算其邻居节点对应特征的线性组合，如下所示：

$$h_i^l = \sigma\left(\sum_{j \in \mathcal{N}_i} \alpha_{ij} W h_j^{l-1}\right) \quad (3)$$

解码器 我们的解码器是一个 Transformer 网络（Vaswani 等人，2017年）。令 $H_t^l = (h_{1t}^l, h_{2t}^l, \ldots, h_{nt}^l)$ 表示图网络最后一层输出的节点表示序列（回想一下，t 在此表示一个交互轮次）。我们的解码器对给定图表示和输入上下文表示的条件下生成 SPARQL 解析树的概率进行建模，即 $p(Y_t \mid H_t^l, Z_t)$。生成 SPARQL 解析树需要生成语法符号（例如 SELECT、WHERE）和知识图谱元素（即实体、类型和关系）。给定第 i^{th} 个生成步骤的解码器状态 s_i，生成 y_i 的概率计算如下：

$$p(y_i \mid y_{<i}, H_t^l, s_i) = p_{\mathcal{G}}(y_i \mid y_{<i}, H_t^l, s_i) + p_{\mathcal{S}}(y_i \mid y_{<i}, s_i) \quad (4)$$

其中，$p_{\mathcal{G}}$ 和 $p_{\mathcal{S}}$ 分别表示生成图节点和语法符号的概率。我们计算 $p_{\mathcal{S}} = \text{softmax}(W_1 s_i)$，使得$W_1 \in \mathbb{R}^{|V_s| \times d}$，其中 $|V_s|$是 SPARQL 语法词汇表的大小。我们使用节点嵌入 H_t^l计算 $p_{\mathcal{G}}$，即 $p_{\mathcal{G}} = \text{softmax}(H_t^l s_i)$。

训练 我们的模型通过优化交叉熵损失进行端到端训练。给定训练实例 $\langle C_t, X_t, Y_t, G_t \rangle$，其中 Y_t 是由标准输出词元 $\langle y_1, y_2, \ldots, y_{|Y_t|} \rangle$组成的序列，我们最小化词元级交叉熵，具体如下：

$$\mathcal{L}(\hat{y}_i) = -log p(y_i \mid X_t, G_t, C_t) \quad (5)$$

其中 $\hat{}y_i$ 表示解码器第 i步预测的输出词元。

5 实验设置

数据集我们在 SPICE(Perez-Beltrachini 等人，2023年近期发布的大规模数据集[4] 上进行了实验。该数据集基于 CSQA 基准构建，用于对话语义解析（Saha 等人，2018年）。 SPICE 该数据集由用户与系统的交互构成，其中自然语言问题与 SPARQL 解析结果配对，系统给出的答案对应于 SPARQL 执行结果（见图1）。表1总结了该数据集的统计信息。如表所示，该数据集包含大量训练实例，对话轮次相对较长（平均轮次长度为 9.5），底层知识图谱规模也很大（包含 1280 万个实体）。 SPICE 该数据集既包含简单的事实性问题，也包含需要对三元组集合进行推理的更复杂问题；此外，它还体现了各种话语相关现象，例如共指和省略。附录中列出了 SPICE C中出现的问题类型。

评估指标遵循先前的研究（Perez-Beltrachini 等人，2023年），我们报告精确匹配准确率和 F1 值（或根据问题类型报告准确率）。精确匹配率是指所占的百分比

[4] https://github.com/EdinburghNLP/SPICE.

# instances	197K
# entities	12.8M
# relations	2,738
# types	3,064
Avg. turn length	9.5
Avg. entities per conversation	7.6
Avg. types per conversations	6.5
Avg. neighborhood per turn	181.4 triples

Table 1: Statistics of the SPICE dataset.

预测的 SPARQL 查询中，与相应标准 SPARQL 查询（gold SPARQL）字符串匹配的查询所占的比例。F1 值（或答案准确率）根据预测查询与标准查询的执行结果计算。对于验证类查询以及涉及定量推理和比较推理的查询，我们计算执行答案准确率。对于其他类型的问题，则将结果视为集合来计算 F1 值。

模型配置 我们的模型使用 PyTorch 实现（Paszke 等人，2019年），并采用 AdamW（Loshchilov 和 Hutter，2019年）优化器进行训练。[5] 模型选择基于验证集上的精确匹配准确率。所有实验均使用两层解码器和两层 GATv2。我们使用 HuggingFace 预训练的 BERT 嵌入（Wolf 等人，2020年），具体采用不区分大小写的基础版本。我们的 GATv2 实现基于 PyTorch Geometric（Fey 和 Lenssen，2019年），并使用两个注意力头。我们采用邻接矩阵堆叠的方法，为 GNN 在不同样本之间创建小批量数据。我们使用 AllenNLP 命名实体识别（NER）系统识别命名实体（Gardner 等人，2018年）。我们的执行结果基于 Perez-Beltrachini 等人 （2023年）提供的维基数据子图。我们的 SPARQL 服务器使用 Blazegraph 部署。[6] 更多实现细节请参见附录B。

如第 4.3 节所述，在每次话语中，我们的模型都会对先前的 t 子图进行编码。更大的上下文虽然包含更多有用信息，但也可能引入噪声。我们将 t 视为超参数，并在开发集上对其取值进行优化。我们报告了 $t = 5$ 的实验结果（详情请参阅附录 A ）。

Comparison Models We compare against the semantic parser of Perez-Beltrachini et al. (2023). Their model is based on BERT (Devlin et al., 2019),

[5]Our code can be downloaded from https://github.com/parajain/dynamic_context.

[6]https://blazegraph.com/

它依赖 AllenNLP（Gardner 等人，2018年）进行命名实体识别，并使用 ElasticSearch[7] 倒排索引进行实体链接。如前所述，它们并未显式执行命名实体消歧（而是将$K = 5$ 最佳匹配实体及其邻域图视为词汇表的一部分），并使用全局查找进行类型链接。由于线性化子图的大小通常超过 BERT 的最大输入词元数量（即 512），因此它们采用了一种变通方法，将图划分为多个子序列，并分别进行编码。我们将其语义解析器称为 BertSP$_{\mathcal{GL}}$，其中 $\mathcal{GL}$ 是全局查找的简称。

除了我们完整的动态上下文图模型（该模型执行上下文相关的类型链接（DCG$_{\mathcal{CL}}$））之外，我们还构建了一个更简单的变体（DCG），该变体仅依赖实体邻域子图来获取类型信息。此外，我们还创建了模型的两个变体：一个能够消解实体歧义，另一个则不能（类似于 Perez-Beltrachini 等人 2023年）。

6 Results

在本节中，我们评估语义解析器在 SPICE 测试集上的性能。我们报告了各个问题类型以及总体的结果。此外，我们还分析了系统处理省略和共指等不同话语现象，以及处理不同长度交互的能力。

The Effect of Dynamic Context Table 2 summarizes our results. We first concentrate on model variants *without entity disambiguation* for a fair comparison with Perez-Beltrachini et al. (2023).

我们将 DCG$_{\mathcal{GL}}$（采用全局查找来进行类型匹配的模型版本）与 BertSP$_{\mathcal{GL}}$ 进行了比较，该版本仅在上下文信息的编码方式上有所不同。如我们所见，基于图的模型表现更好，F1 值达到 72.3%，而 BertSP$_{\mathcal{GL}}$仅为 59%，其性能受到上下文信息编码方式的限制。回顾一下，BertSP$_{\mathcal{GL}}$ 会将子图上下文线性化，将其拆分为多个子序列，然后逐块输入模型。我们的模型通过利用图神经网络高效编码知识图谱信息，同时保留其结构中捕获的依赖关系，缓解了这一问题。因此，DCG$_{\mathcal{GL}}$ 在大多数问题类型上的表现都更好，包括简单问题和推理类问题。

[7]https://www.elastic.co/

	Without Disambiguation						With Disambiguation			
	$\text{DCG}_{\mathcal{CL}}$		$\text{DCG}_{\mathcal{GL}}$		$\text{BertSP}_{\mathcal{GL}}$		$\text{DCG}_{\mathcal{CL}}$		DCG	
Question Type	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM
Clarification	78.60	73.63	80.42	69.47	**83.91**	76.58	82.01	74.82	82.03	72.10
Logical Reasoning	64.12	49.51	51.14	31.54	22.74	28.61	**93.95**	79.52	93.33	78.19
Quantitative Reasoning	55.66	26.29	**93.25**	76.88	76.20	59.01	59.83	31.17	56.66	28.66
Comparative Reasoning	76.06	35.59	80.59	47.28	69.56	39.37	**90.91**	62.46	90.09	61.11
Simple Question (Coref)	86.36	72.03	84.92	67.15	76.51	58.83	**88.49**	79.90	87.41	79.18
Simple Question (Direct)	87.29	71.10	83.24	65.89	71.43	58.71	**88.27**	62.25	85.60	61.44
Simple Question (Ellipsis)	65.22	56.08	52.84	47.48	58.14	50.90	79.08	83.87	**84.35**	82.45
	AC	EM	AC	EM	AC	EM	AC	EM	AC	EM
Verification (Boolean)	78.30	36.97	69.07	21.00	37.16	24.90	**87.41**	66.32	86.75	63.66
Quantitative Reasoning (Count)	61.10	56.94	66.59	62.70	50.86	48.44	**75.20**	70.84	72.96	69.02
Comparative Reasoning (Count)	42.79	30.40	60.91	44.68	43.48	40.67	**67.70**	57.34	66.76	56.60
Overall	69.55	50.85	72.30	53.41	59.00	48.60	**81.28**	66.85	80.59	65.24

表2：在 SPICE 数据集（测试集）上的结果。$\text{BertSP}_{\mathcal{GL}}$ (Perez-Beltrachini 等人，2023年）使用基于 AllenNLP 的命名实体识别（NER）和全局查找（下标 $\mathcal{GL}$）进行类型链接。$\text{DCG}_{\mathcal{CL}}$ 使用上下文相关的类型链接（下标 $\mathcal{CL}$），同时也使用 AllenNLP。DCG 不进行类型链接。我们采用 F1 值、准确率（AC）和精确匹配率（EM）作为评估指标。

问题。我们进一步将 $\text{DCG}_{\mathcal{GL}}$ 与另一种采用上下文相关的类型链接的变体（$\text{DCG}_{\mathcal{CL}}$）进行了比较。我们发现，上下文相关的类型链接略逊于全局查找，这一结果符合预期，因为它无法访问完整的知识图谱类型列表。

总体而言，我们观察到，精确匹配（EM）的结果低于 F1 值或准确率。EM 是一种更为严格的评估指标，不允许预测结果与标准 SPARQL 查询存在任何偏差。然而，两个查询可能具有不同的语法形式，但含义等价；此外，部分格式正确的查询也可能得到部分准确的结果。与 EM 相比，F1 值和准确率允许给予部分得分，因此得分更高。

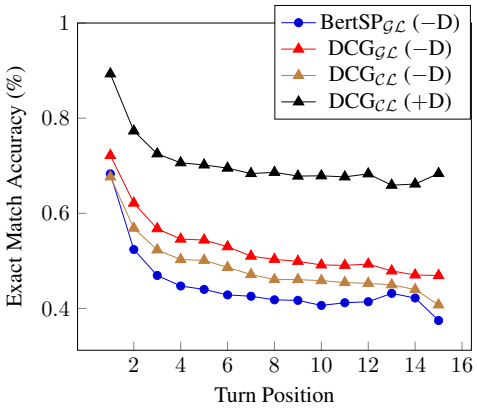

图 3：不同轮次位置上各类问题的精确匹配准确率平均值。 +/−D 表示是否进行实体消歧。

实体消歧的影响 现在，我们展示采用模型变体所得到的结果，该模型在经过消歧的实体上运行（比较表 2中第二组和第五组数据块，标题为 $\text{DCG}_{\mathcal{CL}}$）。我们观察到，消歧对模型性能有显著影响，使 F1 值提高了 11% 以上。我们进一步通过比较 $\text{DCG}_{\mathcal{CL}}$ 与一种无法访问类型图的模型变体，评估依赖上下文的链接的实用性；该变体在训练和评估期间均无法访问类型图 G_t^{type} （参见列 DCG，With Disambiguation）。这种缺少类型信息的模型在 F1 值和 EM 两项指标上的总体表现都较差，但仍优于 $\text{BertSP}_{\mathcal{GL}}$，尽管后者能够通过 top-$K$ 实体查找和全局类型链接获取更多信息。这表明，有针对性地编码上下文比采用蛮力方式更为重要。在附录 A中，我们讨论上下文长度对类型信息的可用性。

对话长度的影响接下来，我们考察动态建模上下文的益处。

理想情况下，无论对话长度如何，模型都应生成准确的语义解析结果。图3 展示了不同轮次位置下的精确匹配准确率（按问题类型取平均值）。总体而言，我们观察到，在对话中较晚出现的话语更难解析。随着对话的推进，对于预期能够利用迄今为止建立的共同基础的模型来说，后续轮次更具挑战性。这不仅涉及根据对话历史维护子图上下文，还需要处理共指和省略等语言现象。总体而言，$\text{DCG}_{\mathcal{CL}}$（无论是否进行实体消歧）均优于 $\text{BertSP}_{\mathcal{GL}}$，并且

	Without Disambiguation			With Disambiguation	
	$DCG_{\mathcal{CL}}$	$DCG_{\mathcal{GL}}$	$BertSP_{\mathcal{GL}}$	$DCG_{\mathcal{CL}}$	DCG
Coref=-1	58.25	51.10	49.39	**74.23**	72.22
Coref< -1	23.52	12.42	00.00	**33.64**	32.40
Ellipsis	39.30	39.90	26.39	**62.26**	61.10
MEntities	43.71	53.46	41.64	**61.59**	58.90

表3：平均精确匹配准确率。Coref=-1 表示指代表达式已在前一轮话语中得到解析的话语。Coref< -1表示指代表达式已在更广泛的话语上下文中（超出前一轮话语范围）得到解析的话语。MEntities 是多个实体（multiple entities）的缩写，指包含复数提及的话语。

gap between the two models is more pronounced for later turns.

话语现象建模 省略和共指等话语现象在对话中十分普遍。省略是指话语中省去的语法成分，这些成分可以根据上下文恢复。在以下交互中：

(Q1) What does Andrei Neagoe do for a living?
(Q2) And how about Wilhelm Dietrichson?

短语 谋生 在（Q2）中被省略，但可以根据（Q1）的上下文理解。另一方面，共指关系发生在指代同一实体的话语之间。例如，图1中话语2和话语3之间就存在这种关系。

在表3中，我们采用精确匹配（Exact Match）指标，评估不同模型在处理各类问题中的省略和共指现象时的表现。Coref=-1 指的是能够在即时上下文中消解共指的情况，即前一个话轮。Coref< -1 则涉及需要访问更广泛对话上下文（超出前一个话轮范围）的话语。在不对实体进行消歧的设置下，我们观察到，与 $BertSP_{\mathcal{GL}}$相比，能够利用话语上下文的模型（$DCG_{\mathcal{GL}}$ 和 $DCG_{\mathcal{CL}}$的变体）在解析共指表达和省略表达方面表现更好。我们还发现，实体消歧非常有帮助，使 $DCG_{\mathcal{CL}}$ 在话语相关现象方面取得了显著提升。

与 Perez-Beltrachini 等人 (2023年) 类似，我们还评估了模型在包含复数提及的话语上的表现；这些话语通常与多个实体相关联，语义解析器必须枚举这些实体才能构建正确的解析结果（表 3中的 MEntities）。$DCG_{\mathcal{CL}}$ 在进行消歧时总体表现最佳，而 DCG（不含类型链接）的表现较差。这并不令人意外，因为包含多个实体的话语通常具有复杂的解析结构，其中 multiple sub-queries and entity types, which DCG does not have access to.

解析错误的性质 总体而言，我们发现模型能够预测语法有效的 SPARQL 查询。错误主要源于：在给定图上下文和先前问题的情况下，误解问题的意图，或缺少必要信息。我们的模型在解析澄清问题和定量推理问题时也存在困难。对于澄清问题，模型无法在澄清后选择正确的实体。例如，在以下对话中：

答案：Peter G. Van Winkle、Arthur I. Boreman、William E. Stevenson 话语：那个人会写哪种语言？澄清：你指的是 Arthur I. Boreman 吗？ 答案：不，我指的是 Peter G. Van Winkle。你能告诉我这个问题的答案吗？

它选择了 *Arthur I. Boreman* (Q709961)，而不是 *Peter G. VanWinkle* (Q1404201)，从而导致 SQL 解析结果错误。在这种情况下，更广泛的上下文覆盖了紧接着的上一轮对话中有用的信息。然而，根据具体的问题意图来确定相关上下文将有所帮助，不过我们将这一问题留待未来工作。

类型链接失败是导致定量推理问题出错的主要原因，因为这类问题通常没有上下文或上下文非常有限（例如：“哪些铁路车站恰好由 1 个社会群体管理？”）。然而，我们的模型依赖于实体邻域中类型的可用性，因为它以依赖上下文的方式执行类型链接。我们观察到，当能够访问所有知识图谱类型时，模型在解析此类问题方面的表现会更好（见表2，$DCG_{\mathcal{GL}}$ 与 $DCG_{\mathcal{CL}}$）。

7 Conclusions

本文提出了一种用于知识库问答的语义解析器，该解析器能够在先前话语的上下文中，以交互方式将用户话语映射为可执行的逻辑形式。我们的模型将有关话语及其上下文的信息表示为知识图谱子图，这些子图动态生成，并通过图神经网络进行编码。此外，我们还提出了一种高效且可扩展的上下文相关类型链接方法。

我们的实验表明，更好地建模上下文信息能够提升实体和类型链接、指代消解与省略消解方面的性能，并有助于持续跟踪其中的.....

随着对话的不断推进，交互历史也随之变化。未来的研究方向多种多样。在实验中，我们使用现成的命名实体识别（NER）系统，然而，联合学习语义解析器和实体链接器将使二者相互受益，并避免错误传播。鉴于对整个知识图谱（KG）进行编码的代价过高，我们采用动态方式对相关子图进行编码。我们还可以进一步显式建模知识图谱实体与问题词元之间的关系，先前的研究表明，这一方向具有良好的应用前景（Wang 等人，2020）。最后，如何调整我们的模型，使其能够处理非独立同分布（non-i.i.d.）的泛化场景，也将是一个值得研究的方向。

8 Limitations

我们的模型依赖一个预训练的命名实体识别（NER）模块来实现实体链接。由于该模块是在特定数据集上训练和评估的，因此其性能可能无法泛化到维基数据中未见过的领域。此外，我们没有明确考虑关系。我们假设所需的正确信息始终可用，但实际情况并非总是如此。我们专注于编码知识图谱的结构信息，并将知识图谱与语言话语之间交互关系的学习交由解码器完成。正如先前的研究所示（Zhang 等人，2022），在问答任务的上下文中，将知识图谱信息与语言模型有效结合能够实现互利共赢。然而，要确定特定任务的参数化方式，仍需开展一项广泛而深入的研究（Wang 等人，2022）。

Acknowledgements

本研究部分得到了华为以及英国自然语言处理博士培训中心（UKRI Centre for Doctoral Training in Natural Language Processing）的资助（项目编号：EP/S022481/1）。Lapata 衷心感谢英国工程与自然科学研究理事会（Engineering and Physical Sciences Research Council）的支持（项目编号：EP/W002876/1）。

参考文献

阿尔弗雷德·V·阿霍和玛格丽特·J·科拉西克。1975年。高效字符串匹配：一种辅助文献检索的方法。《*ACM* 通讯》，18(6)：333–340。

Nikita Bhutani、Xinyi Zheng 和 H V Jagadish。2019年。通过查询组合学习使用知识库回答复杂问题。载于 第*28*届 *ACM* 国际信息与知识管理会议论文集，CIKM '19，第739–748页，美国纽约州纽约市。计算机协会。

Ben Bogin、Matt Gardner 和 Jonathan Berant。2019年。面向文本到 SQL 解析的数据库结构全局推理。载于 *2019*年自然语言处理实证方法会议和第九届自然语言处理国际联合会议（*EMNLP-IJCNLP*）论文集，第 3659–3664 页，中国香港。计算语言学协会。

Shaked Brody、Uri Alon 和 Eran Yahav。2022。图注意力网络的注意力程度如何？载于第十届国际学习表征会议（*ICLR 2022*），线上会议，*2022*年*4*月*25*日至*29*日。OpenReview.net。

Yitao Cai 和 Xiaojun Wan。2020。IGSQL：一种基于数据库模式交互图的上下文相关文本到 SQL 生成神经模型。载于*2020* 年自然语言处理实证方法会议（*EMNLP*）论文集，第 6903–6912 页，线上。计算语言学协会。

Deborah A. Dahl、Madeleine Bates、Michael Brown、William Fisher、Kate Hunicke-Smith、David Pallett、Christine Pao、Alexander Rudnicky 和 Elizabeth Shriberg。1994年。扩展 ATIS任务的范围：ATIS-3语料库。载于人类语言技术：*1994*年*3*月*8*日至*11*日在新泽西州普莱恩斯伯勒举行的研讨会论文集。

Jacob Devlin、Ming-Wei Chang、Kenton Lee 和 Kristina Toutanova。2019年。BERT：用于语言理解的深度双向变换器预训练。载于计算语言学协会北美分会*2019*年年会论文集：人类语言技术，第*1*卷（长篇与短篇论文），第4171–4186页，明尼阿波利斯，明尼苏达州。计算语言学协会。

Jiwei Ding、Wei Hu、Qixin Xu 和 Yuzhong Qu。2019年。利用高频查询子结构生成用于复杂问答的形式化查询。载于*2019*年自然语言处理实证方法会议暨第九届自然语言处理国际联合会议（*EMNLP-IJCNLP*）论文集，第2614–2622页，中国香港。计算语言学协会。

Li Dong 和 Mirella Lapata。2018年。用于神经语义解析的由粗到细解码。载于计算语言学协会第*56*届年会论文集（第*1*卷：长篇论文），第731–742页，墨尔本，澳大利亚。计算语言学协会。

Matthias Fey and Jan E. Lenssen. 2019. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*.

Matt Gardner、Joel Grus、Mark Neumann、Oyvind Tafjord、Pradeep Dasigi、Nelson F. Liu、Matthew Peters、Michael Schmitz 和 Luke Zettlemoyer。2018年。

AllenNLP：一个深度语义自然语言处理平台。在自然语言处理开源软件研讨会（*NLP-OSS*）论文集，第 1–6 页，澳大利亚墨尔本。计算语言学协会。

M. Gori、G. Monfardini 和 F. Scarselli。2005年。一种用于图领域学习的新模型。载于*2005*年*IEEE*国际神经网络联合会议论文集。，第2卷，第729–734页，第2卷。

Yu Gu, Vardaan Pahuja, Gong Cheng, and Yu Su. 2022. Knowledge base question answering: A semantic parsing perspective.

Xixin Hu、Yiheng Shu、Xiang Huang 和 Yuzhong Qu。2021年。基于边的问题分解：面向知识库的复杂问答。载于语义网 – *ISWC 2021*，第128–145页，Cham。施普林格国际出版。

Binyuan Hui、Ruiying Geng、Qiyu Ren、Binhua Li、Yongbin Li、Jian Sun、Fei Huang、Luo Si、Pengfei Zhu 和 Xiaodan Zhu。2021年。用于跨领域上下文依赖语义解析的动态混合关系探索网络。*AAAI* 人工智能会议论文集，35(14):13116–13124。

Binyuan Hui、Ruiying Geng、Lihan Wang、Bowen Qin、Yanyang Li、Bowen Li、Jian Sun 和 Yongbin Li。2022 年。S^2SQL：为文本转SQL解析器的问题-模式交互图编码器注入语法。载于计算语言学协会论文集：*ACL 2022*，第 1254–1262 页，都柏林，爱尔兰。计算语言学协会。

Parag Jain and Mirella Lapata. 2021. Memory-Based Semantic Parsing. *Transactions of the Association for Computational Linguistics*, 9:1197–1212.

Endri Kacupaj、Joan Plepi、Kuldeep Singh、Harsh Thakkar、Jens Lehmann 和 Maria Maleshkova。2021年。基于 Transformer 和图注意力网络的知识图谱对话式问答。载于第*16*届欧洲计算语言学协会分会会议论文集：主卷，第850–862页，线上。计算语言学协会。

Ilya Loshchilov 和 Frank Hutter。2019年。 解耦权重衰减正则化。发表于 第七届国际学习表征会议（*ICLR 2019*），美国路易斯安那州新奥尔良，*2019*年*5*月*6*日至*9*日。OpenRe- view.net。

Pierre Marion、Pawel Nowak 和 Francesco Piccinno。2021年。用于知识图谱对话式问答的结构化上下文与高覆盖率语法。载于*2021*年自然语言处理实证方法会议论文集，第8813–8829页，线上及多米尼加共和国蓬塔卡纳。计算语言学协会。

Adam Paszke、Sam Gross、Francisco Massa、Adam Lerer、James Bradbury、Gregory Chanan、Trevor Killeen、Zeming Lin、Natalia Gimelshein、Luca Antiga、Alban Desmaison、Andreas Kopf、Edward Yang、Zachary DeVito、Martin Raison、Alykhan Te- jani、Sasank Chilamkurthy、Benoit Steiner、Lu Fang、Junjie Bai 和 Soumith Chintala。2019年。PyTorch：一种命令式风格的高性能深度学习库。载于神经信息处理系统进展，第32卷，第8026–8037页。Curran Associates, Inc.

Laura Perez-Beltrachini、Parag Jain、Emilio Monti 和 Mirella Lapata。2023年。基于知识图谱的对话式问答语义解析。载于第*17*届欧洲计算语言学协会分会会议论文集，第2499–2514页，克罗地亚杜布罗夫尼克。计算语言学协会。

Srinivas Ravishankar、June Thai、Ibrahim Abdelaziz、Nandana Mihindukulasooriya、Tahira Naseem、Pa- van Kapanipathi、Gaetano Rossillo 和 Achille Fok- oue。2021年。知识库问答泛化的两阶段方法。*CoRR*，abs/2111.05825。

Amrita Saha、Vardaan Pahuja、Mitesh Khapra、Karthik Sankaranarayanan 和 Sarath Chandar。2018年。复杂序列问答：利用知识图谱，基于链接问答对学习对话。载于第三十二届*AAAI*人工智能会议论文集（*AAAI-18*），第705–713页，美国路易斯安那州新奥尔良。AAAI出版社。

Franco Scarselli、Marco Gori、Ah Chung Tsoi、Markus Hagenbuchner 和 Gabriele Monfardini。2009年。图神经网络模型。*IEEE* 神经网络汇刊，20(1):61–80。

Wei Shen、Jianyong Wang 和 JiaweiHan。2015年。基于知识库的实体链接：问题、技术与解决方案。*IEEE* 知识与数据工程汇刊，27(2)：443–460。

Alane Suhr、Srinivasan Iyer 和 Yoav Artzi。2018年。学习将上下文相关句子映射为可执行的形式化查询。载于 *2018*年计算语言学协会北美分会年会论文集：人类语言技术，第*1*卷（长篇论文），第2238–2249页，新奥尔良，路易斯安那州。计算语言学协会。

Ashish Vaswani、Noam Shazeer、Niki Parmar、Jakob Uszkoreit、Llion Jones、Aidan N. Gomez、Lukasz Kaiser 和 Illia Polosukhin。2017年。注意力就是你所需要的一切。《神经信息处理系统进展》第*30*卷：*2017*年神经信息处理系统年度会议，*2017*年*12*月*4*日至*9*日，美国加利福尼亚州长滩，第5998–6008页。

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio.

<*escape*>2018年。 图注意力网络。载于 第六届国际学习表征会议（*ICLR 2018*），加拿大不列颠哥伦比亚省温哥华，*2018*年*4*月*30*日至*5*月*3*日，会议论文集。OpenReview.net。</*escape*>

DennyVrandeˇci´c 和 Markus Krötzsch。2014 年。Wiki-data：一个免费的协作知识库。*ACM* 通讯，57(10):78–85。

Bailin Wang、Richard Shin、Xiaodong Liu、Oleksandr Polozov 和 Matthew Richardson。2020年。RAT-SQL：面向文本到 SQL 解析器的关系感知模式编码与链接。载于第*58*届计算语言学协会年会论文集，第7567–7578页，线上。计算语言学协会。

Kuan Wang、Yuyu Zhang、Diyi Yang、Le Song 和 Tao Qin。2022年。GNN 是一个计数器吗？重新审视用于问答的 GNN。载于第十届国际学习表征会议（*ICLR 2022*），线上会议，*2022*年*4*月*25*日至*29*日。OpenReview.net。

Thomas Wolf、Lysandre Debut、Victor Sanh、J ulien Chaumond、Clement Delangue、Anthony Moi、Pier- ric Cistac、Tim Rault、Remi Louf、Morgan Funtow- icz、Joe Davison、Sam Shleifer、Patrick von Platen、Clara Ma、Yacine Jernite、Julien Plu、Canwen Xu、Teven Le Scao、Sylvain Gugger、Mariama Drame、Quentin Lhoest 和 Alexander Rush。2020 年。Transformers：最先进的自然语言处理技术。载于*2020*年自然语言处理实证方法会议论文集：系统演示，第 38–45 页，线上。计算语言学协会。

Xiaoyu Yin、Dagmar Gromann 和 Sebastian Rudolph。2021年。从自然语言到 SPARQL 的神经机器翻译。未来一代计算机系统，117:510–519。

Tao Yu、Rui Zhang、Heyang Er、Suyi Li、Eric Xue、Bo Pang、Xi Victoria Lin、Yi Chern Tan、Tianze Shi、Zihan Li、Youxuan Jiang、Michihiro Yasunaga、Sungrok Shim、Tao Chen、Alexander Fabbri、Zifan Li、Luyao Chen、Yuwen Zhang、Shreya Dixit、Vin- cent Zhang、Caiming Xiong、Richard Socher、Walter Lasecki 和 Dragomir Radev。2019年a。CoSQL：一项面向跨领域数据库自然语言接口的对话式文本到 SQL 挑战。载于*2019*年自然语言处理实证方法会议暨第九届国际自然语言处理联合会议（*EMNLP-IJCNLP*）论文集，第1962–1979 页，中国香港。计算语言学协会。

陶宇、张睿、Michihiro Yasunaga、Yi Chern Tan、Xi Victoria Lin、李素怡、贺阳、Irene Li、Bo Pang、陈涛、Emily Ji、Shreya Dixit、David Proctor、SungrokShim、Jonathan Kraft、Vin- cent Zhang、Caiming Xiong、Richard Socher 和 Dragomir Radev。2019年b。SParC：上下文中的跨领域语义解析。发表于《*57th Annual Meeting of the Association for Computational Linguistics*, pages 4511–4523, Florence, Italy. Association for Computational Linguistics.

Xikun Zhang、Antoine Bosselut、Michihiro Yasunaga、Hongyu Ren、Percy Liang、Christopher D. Manning 和 Jure Leskovec。2022。GreaseLM：图推理增强型语言模型。载于第十届国际学习表征会议（*ICLR 2022*），线上会议，*2022*年*4*月*25*日至*29*日。OpenReview.net。

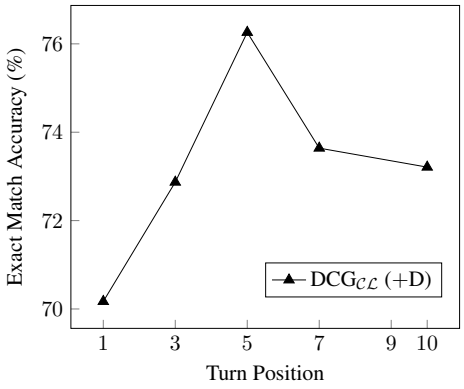

Figure 4: Exact Match Accuracy for different context lengths averaged across question types (validation set; +D: with entity disambiguation).

A The Effect of Context Length

图4 展示了 DCG$_{\mathcal{CL}}$ （消歧设置）在上下文长度逐步增加时的性能表现 $\in [1, 10]$。我们观察到，能够访问更广泛的上下文在一定范围内是有益的。当上下文过长时（超过第 5 轮），性能会有所下降。我们认为这主要有两个原因。首先，较长的交互可能是因为用户询问了多个实体或主题，在这种情况下，局部上下文可能已足以提供答案。其次，较长的交互本身可能确实更容易产生混淆，且标注人员难以创建此类交互，更不用说模型了。

我们进一步评估了上下文长度与类型信息可用性之间的相互作用。表 4展示了在上下文长度为 1 和 5 时，使用和不使用显式类型链接所带来的性能差异。如第 5节所述，DCG 不使用显式类型链接，而 DCG$_{\mathcal{CL}}$ 使用依赖于上下文的链接，但两种模型均采用实体消歧。Δ_{F1_1} 表示上下文长度为 $_{\mathcal{CL}}$ 1 时，DCG与 DCG Δ_{F1_5} 之间 F1 值的绝对差异。同样，Δ_{F1_5} 表示上下文长度为5时的差异。

总体而言，我们发现，与上下文长度相比，上下文长度为 1 时的性能显著下降。

	Context Length 1				Context Length 5				Diff due to type linking			
	$DCG_{\mathcal{CL}}$		DCG		$DCG_{\mathcal{CL}}$		DCG					
Question Type	F1	EM	F1	EM	F1	EM	F1	EM	Δ_{F1_1}	Δ_{EM_1}	Δ_{F1_5}	Δ_{EM_5}
Clarification	75.66	68.61	52.22	53.87	82.01	74.82	82.03	72.1	23.44	14.74	0.02	2.72
Logical Reasoning	92.59	77.07	86.9	67.94	93.95	79.52	93.33	78.19	5.69	9.13	0.62	1.33
Quantitative Reasoning	36.1	13.74	30.91	11.04	59.83	31.17	56.66	28.66	5.19	2.7	3.17	2.51
Comparative Reasoning	76.72	39.75	74.77	37.79	90.91	62.46	90.09	61.11	1.95	1.96	0.82	1.35
Simple Question (Coref)	88.18	79.83	83.04	76.27	88.49	79.9	87.41	79.18	5.14	3.56	1.08	0.72
Simple Question (Direct)	87.56	61.59	79.74	58.21	88.27	62.25	85.6	61.44	7.82	3.38	2.67	0.81
Simple Question (Ellipsis)	80.38	81.75	74.2	76.84	79.08	83.87	84.35	82.45	6.18	4.91	5.27	1.42
	AC	EM	AC	EM	AC	EM	AC	EM	Δ_{AC_1}	Δ_{EM_1}	Δ_{AC_5}	Δ_{EM_5}
Verification (Boolean)	88.02	61.45	82.15	48.24	87.41	66.32	86.75	63.66	5.87	13.21	0.66	2.66
Quantitative Reasoning (Count)	69.41	65.34	65.16	60.58	75.2	70.84	72.96	69.02	4.25	4.76	2.24	1.82
Comparative Reasoning (Count)	42.81	30.5	39.74	28.69	67.7	57.34	66.76	56.6	3.07	1.81	0.94	0.74
Overall	73.74	57.96	66.88	51.95	81.28	66.85	80.59	65.24	6.86	6.01	0.69	1.61

表 4：上下文长度与类型链接的交互。 Δ_{F1_1} 是 $DCG_{\mathcal{CL}}$ 与 DCG 在上下文长度为 1 时 F1 值的绝对差值。 Δ_{F1_5} 是上下文长度为 5 时 F1 值的绝对差值。

长度为 5。这表明随着上下文长度的增加，可获得更多类型信息。然而，性能会因问题类型而异。具体而言，与似乎需要访问更大 KB 子图的定量推理问题相比，澄清问题的精确匹配差异要大得多。

B Model Details

我们的模型使用 PyTorch（Paszke 等人，2019年）实现，并采用 AdamW（Loshchilov 和 Hutter，2019年）优化器进行训练。模型使用 A100 GPU 进行训练，批量大小为 64，初始学习率为 0.001。AdamW 系数 β_1 和 β_2 （用于计算梯度及其平方的运行平均值）分别设置为 0。9 和 0。999。所有实验的权重衰减系数均设置为 0.01。超参数根据使用手动选择的网格进行的初步实验确定。我们没有调整学习率参数。我们从 $[1, 4]$ 中选择 GATv2 和解码器层的数量，并发现 2 层的效果最佳。我们的 SPARQL 查询服务器使用 Blazegraph 进行部署。[8]该服务器仅使用基于 CPU 的资源，并可访问 100G RAM。

我们使用 GATv2 的两个注意力头。具体而言，令 K 表示按照公式（（Velickovic 等人，，2018年）（3）计算得到的注意力头。每个注意力头的输出按如下方式拼接：

$$h^l = \Big\Vert_{k=1}^{K} \sigma\left(\sum_{j \in \mathcal{N}_i} \alpha_{ij}^k W^k h_j^{l-1}\right)$$

where $\|$ represents concatenation. α_{ij}^k are normalized attention coefficients computed by the k-th attention mechanism as in Equation (1).

<escape>我们的图采用邻接矩阵表示。为了创建一个小批次，我们将邻接矩阵沿对角线堆叠（Fey 和 Lenssen，2019年）。这样便创建了一个组合图，其中包含多个彼此隔离的子图，如下所示：</escape>

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_1 & & \\ & \ddots & \\ & & \mathbf{A}_n \end{bmatrix}$$

where n is the batch-size number of graphs. Node input H and target $\bar{H}$ features are simply concatenated in the node dimension as follows:

$$\mathbf{H} = \begin{bmatrix} \mathbf{H}_1 \\ \vdots \\ \mathbf{H}_n \end{bmatrix}, \qquad \bar{\mathbf{H}} = \begin{bmatrix} \bar{\mathbf{H}}_1 \\ \vdots \\ \bar{\mathbf{H}}_n \end{bmatrix}.$$

[8] https://blazegraph.com/

C The SPICE Dataset: Question Types

Simple Question (Ellipsis)	Utterance: Who created the design for Samus Aran? Answer: Hiroji Kiyotake Utterance: And how about The Dreamland Chronicles: Freedom Ridge? Answer: Julian Gollop
Simple Question (Direct)	Utterance: Who starred in Mathias Kneissl ? Answer: Rainer Werner Fassbinder, Volker Schlöndorff, Hanna Schygulla
Simple Question (Coreferenced)	Utterance: Who was the director of that work of art ? Answer: Reinhard Hauff
Verification (Boolean)	Utterance: Does Dubashi have that person as actor ? Answer: No
Logical Reasoning (All)	Utterance: Which works of art are Rainer Werner Fassbinder or Laura Esquivel a screenwriter of ? Answer: The American Soldier, Lili Marleen, Love Is Colder Than Death...
Clarification	Utterance: How many administrative territories or political territories did that work of art originate ? Did you mean Lili Marleen ? No, I meant Querelle. Could you tell me the answer for that? Answer: 2
Quantitative Reasoning (Count)	Utterance: How many still waters are situated nearby Norway or Austria-Hungary ? Answer: 2
Comparative Reasoning (Count)	Utterance: How many watercourses are more number of landscapes located on than Kafue River ? Answer: 2
Comparative Reasoning (All)	话语：哪些政治领土附近的水道或水体数量较少 于巴雷约? 答案：2

表 5：在 SPICE (Perez-Beltrachini 等人，2023年）中出现的问题类型示例。灰色的话语旨在便于理解。