

# Context-aware Inductive Knowledge Graph Completion with Latent Type Constraints and Subgraph Reasoning

Muzhi Li<sup>1,2\*</sup>, Cehao Yang<sup>3,2\*</sup>, Chengjin Xu<sup>2\*</sup>, Zixing Song<sup>4</sup>, Xuhui Jiang<sup>2</sup>, Jian Guo<sup>2†</sup>,  
Ho-fung Leung<sup>‡</sup>, Irwin King<sup>1†</sup>

<sup>1</sup>Department of Computer Science and Engineering, The Chinese University of Hong Kong

<sup>2</sup>IDEA Research, International Digital Economy Academy

<sup>3</sup>Artificial Intelligence Thrust, Hong Kong University of Science and Technology (Guangzhou)

<sup>4</sup>Department of Engineering, University of Cambridge

{mzli,king}@cse.cuhk.edu.hk, {limuzhi,yangcehao,xuchengjin,guojian}@idea.edu.cn,  
cyang289@connect.hkust-gz.edu.cn, zs456@cam.ac.uk, ho-fung.leung@outlook.com

## Abstract

Inductive knowledge graph completion (KGC) aims to predict missing triples with unseen entities. Recent works focus on modeling reasoning paths between the head and tail entity as direct supporting evidence. However, these methods depend heavily on the existence and quality of reasoning paths, which limits their general applicability in different scenarios. In addition, we observe that latent type constraints and neighboring facts inherent in KGs are also vital in inferring missing triples. To effectively utilize all useful information in KGs, we introduce CATS, a novel context-aware inductive KGC solution. With sufficient guidance from proper prompts and supervised fine-tuning, CATS activates the strong semantic understanding and reasoning capabilities of large language models to assess the existence of query triples, which consist of two modules. First, the type-aware reasoning module evaluates whether the candidate entity matches the latent entity type as required by the query relation. Then, the subgraph reasoning module selects relevant reasoning paths and neighboring facts, and evaluates their correlation to the query triple. Experiment results on three widely used datasets demonstrate that CATS significantly outperforms state-of-the-art methods in 16 out of 18 transductive, inductive, and few-shot settings with an average absolute MRR improvement of 7.2%.

**Code** — <https://github.com/IDEA-FinAI/CATS>

## Introduction

Knowledge Graphs (KGs) are graph-structured knowledge bases that represent facts with triples in the form of (*head entity, relation, tail entity*). KGs become essential in various downstream applications such as question answering (Sun et al. 2024), fact checking (Kim et al. 2023), and recommendation systems (Wang et al. 2019). In practice, most real-world KGs are incomplete, which highlights the significance of the knowledge graph completion (KGC) (or *relation prediction*) task, which aims to predict the missing head or tail entity from the query triples.

\*Equal contribution.

†Corresponding authors.

‡Independent researcher.

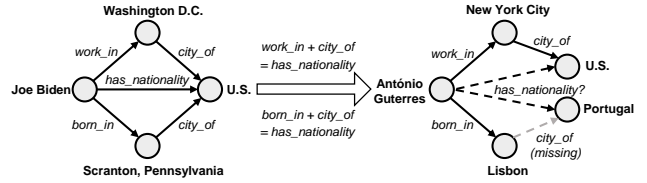


Figure 1: A typical scenario of inductive knowledge graph completion. The model needs to infer the nationality of the unseen entity “António Guterres”.

Existing approaches to the KGC task usually perform well under an “*transductive setting*”, where missing entities can be observed in training triples. Conversely, the “*inductive*” KGC task requires the model to handle newly emerged entities, which is more reflective of real-world scenarios, as KGs are continuously evolving. The inductive setting highlights the importance of three key contexts inherent in KGs, namely entity types, reasoning paths, and neighboring facts.

First, relations within KGs impose *latent type constraints* to head and tail entities being connected, which are crucial in inferring potential missing triples. For instance, the relation *works in* typically connects a *person* (head) and a *location* (tail). Although we have not encountered newly emerged entities during training, a triple can still be considered plausible if the head and tail entities conform to the implicit types as required by the relation.

Second, *reasoning paths* provide direct clues for the existence of missing triple (Zha, Chen, and Yan 2022). However, these paths can be unreliable in certain contexts. For example, in Figure 1, if the training set contains numerous triples such as (*Joe Biden, works in, Washington, D.C.*), (*Washington, D.C., city of, U.S.*), and (*Joe Biden, has nationality, U.S.*), the model may come up with some rules like (*works in + city of = has nationality*). It should be noted that such implicit rules do not invariably apply. A notable counterexample is *António Guterres*, who works in *New York City* as the Secretary-General of the United Nations.

Finally, *neighboring facts* of the head and tail entities also provide valuable clues for triple completion. For instance, in

Figure 1, it is difficult to predict entity “António Guterres” has a Portuguese nationality solely based on the available reasoning path. Fortunately, the presence of specific neighboring facts such as (*António Guterres, born in, Lisbon*) can help disambiguate the proper answer from the distracter.

Despite great efforts, existing methods cannot fully utilize these contexts. Specifically, embedding-based methods (Bordes et al. 2013; Sun et al. 2019) need expensive re-training to embed unseen entities; GNN-based methods (Schlichtkrull et al. 2018; Teru, Denis, and Hamilton 2020) are less robust when few connections between existing and new entities are available; path-based methods (Das et al. 2018; Zha, Chen, and Yan 2022) rely strongly on the existence and reliability of reasoning paths between certain entities; text-enhanced methods disregard entity type properties in KGs.

A direct and promising approach to effectively utilize the three types of contexts is the integration of large language models (LLMs). On the one hand, LLMs, trained on extensive corpora, possess a fundamental understanding of the type of KG entities. On the other hand, the strong semantic understanding and reasoning capabilities enable LLMs to capture crucial information from triples and paths (Liao et al. 2024). *Nevertheless, existing LLM-based KGC methods (Wei et al. 2023; Liu et al. 2024) can only re-rank candidate answers provided by previous KGC approaches.* Consequently, they are inevitably constrained by the limitations of preceding models. *Moreover, these approaches rely on additional triples from the validation set for in-context demonstration (Wei et al. 2023) or supervised fine-tuning (Liu et al. 2024), which can lead to severe information leakage when being applied to inductive scenarios.*

This paper proposes “CATS”, a novel Context-Aware approach for the inductive KGC task based on latent Type constraints and Subgraph reasoning. Considering the semantic gap between natural language sentences and structural KG triples, CATS fine-tunes and guides LLMs to assess the existence of potential missing triples from two perspectives. First, the *Type-Aware Reasoning* (TAR) module evaluates whether the candidate entity conforms to the implicit type constrained by the relation. Since explicit type annotations are not prevalent for entities in non-encyclopedic KGs (e.g. Wordnet), we instead assess whether the candidate head/tail entity and other head/tail entities connected by the same relation belong to the same entity type. Then, the *Subgraph Reasoning* (SR) module proposes a degree-based filtering mechanism to select meaningful paths, and takes relevant neighboring facts of the head and tail entity into consideration. The superior long-context understanding capabilities of LLMs allow the SR module to comprehensively evaluate whether different paths and neighboring facts support the existence of the specific triple. Finally, we ensemble the inference results based on the scorings obtained from the two modules mentioned above.

We conduct extensive experiments on three widely used datasets: WN18RR, FB15k237, and NELL-995. The best variant of CATS significantly outperforms state-of-the-art approaches in 16 out of 18 transductive, inductive, and few-shot settings with an average absolute improvement of 7.2%

in MRR and 10.1% in Hits@1. These results highlight the importance of incorporating the three types of contexts in KGs. Furthermore, ablation studies on various LLMs and configurations show that the effectiveness of the proposed method does not rely on internal knowledge and the scale of LLMs. Our contributions are summarized as follows:

- We propose CATS, the first LLM-based inductive KGC solution capable of handling unseen entities without any external knowledge or prior inference results.
- We devise two novel triple evaluation mechanisms based on latent type constraints, as well as the reasoning paths and neighboring facts within the local subgraph.
- We conduct extensive experiments to evaluate the effectiveness of CATS in different settings, and discuss the contribution of each component and the LLM in detail.

## Related Works

**Embedding-based methods.** The majority of KGC methods rely on KG embeddings, such as TransE (Bordes et al. 2013), RotatE (Sun et al. 2019), and GIE (Cao et al. 2022). These methods learn a set of low-dimensional embeddings for each entity and relation within the KG with certain geometric assumptions, which are inherently transductive. However, they require costly retraining to handle unseen entities (Zha, Chen, and Yan 2022), limiting their adaptability to inductive scenarios.

**Graph neural network (GNN)-based methods.** Graph neural networks (GNNs) (Song, Zhang, and King 2023a,b) are popular in natural language processing for modeling relationships between entities (Ma et al. 2023). GNN-based methods such as CompGCN (Vashishth et al. 2020), RGCN (Schlichtkrull et al. 2018), and WGCN (Zhao et al. 2021) embed entities in a KG by iteratively aggregating features from the local neighborhood. Nevertheless, these approaches struggle to produce meaningful embeddings for newly emerged entities with few links to existing ones, and are not applicable to entirely new graphs (Zha, Chen, and Yan 2022). To conform to the inductive setting, GraIL (Teru, Denis, and Hamilton 2020), and TACT (Chen et al. 2021) embed entities in the local subgraph with their distances to the head and tail entities of the query triple. However, such an embedding approach fails to distinguish different entities that share the same relative position. Consequently, it cannot perform well when the subgraph of query triple is large. RED-GNN (Zhang and Yao 2022) and Adaprop (Zhang et al. 2023) enhance the message-passing mechanisms through progressive and adaptive propagation. Still, these methods fail to address the suboptimal performance of GNNs on sparse graph structures.

**Path-based methods.** Path-based methods aim to mine rules from the co-existences of certain reasoning paths connecting the head and tail entities in a triple and the relation between them (Meilicke et al. 2018). In particular, DeepPath (Xiong, Hoang, and Wang 2017) and MINERVA (Das et al. 2018) exploit random-walk with reinforcement learning to generate reasoning paths, while BERTRL (Zha, Chen,

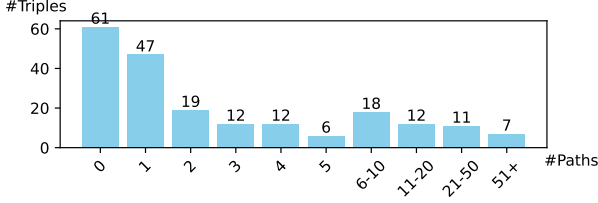


Figure 2: The statistics on the existence of reasoning paths for the query triples in the FB15k-237 (inductive) dataset.

and Yan 2022) and KRST (Su et al. 2023) leverage breadth-first search. However, the existence and quality of reasoning paths between unseen entities is not ensured (Su et al. 2024), which inevitably limits their generality.

**Text-enhanced methods.** In addition to the graph structure, the textual information provided in the KG also entails valuable semantic knowledge (Li et al. 2024). Recently, several KGC methods such as KG-BERT (Yao, Mao, and Luo 2019), BERTRL (Zha, Chen, and Yan 2022), and KRST (Su et al. 2023) employ PLMs to embed entities, relations, and reasoning paths with textual labels and descriptions. APST (Su et al. 2024) further introduces incomplete anchor paths for unseen entities that are not connected with any reasoning paths, achieving state-of-the-art performance. As the authors stated in (Zha, Chen, and Yan 2022), combining multiple reasoning paths encourages knowledge interactions. Nevertheless, their BERT-based backbone PLMs (Devlin et al. 2019) can only each reasoning path independently, leaving significant room for improvements.

## Preliminaries

**Problem specification.** A knowledge graph (or “KG” denoted as  $\mathcal{G} = \{\mathcal{E}, \mathcal{R}, \mathcal{T}\}$ ) consists of a set of triples  $\mathcal{T} = \{(h, r, t)\}$  where head and tail entities  $h, t \in \mathcal{E}$ , relation  $r \in \mathcal{R}$ .  $\mathcal{E}$  and  $\mathcal{R}$  denote the set of entities and relations, respectively. Following the convention of (Teru, Denis, and Hamilton 2020; Su et al. 2023, 2024), the task of inductive knowledge graph completion is defined as follows:

**Definition 1 (Inductive KGC):** Given a training graph  $\mathcal{G}_{train} = \{\mathcal{E}_{train}, \mathcal{R}_{train}, \mathcal{T}_{train}\}$  and a test graph  $\mathcal{G}_{test} = \{\mathcal{E}_{test}, \mathcal{R}_{test}, \mathcal{T}_{test}\}$ , the inductive KGC task aims to complete the missing head or tail entity from a set of query triples  $Q = \{h_q, r_q, t_q\}_{q=1}^{|Q|}$  such that  $\mathcal{E}_{train} \cap \mathcal{E}_{test} = \emptyset$ ,  $\mathcal{R}_{test} \subseteq \mathcal{R}_{train}$ ,  $\forall q, h_q, t_q \in \mathcal{E}_{test}, r_q \in \mathcal{R}_{test}$ .

The settings of the inductive KGC task ensure that entities in the training and test KGs form two disjoint sets. Only the triples in the training graph can be used in model training, while triples in the test graph are provided as evidence for query triple completion. Handling unseen entities requires the model to have inductive reasoning capabilities.

**Type properties of entities.** Apart from triples, entities within a KG are often annotated with entity types (or categories) in an ontological taxonomy (Hao et al. 2019). For instance, in Freebase (Bollacker et al. 2008), entity “Albert

Einstein” belongs to the “/scientist/physicist” type. In general, entity types provide a high-level summary of the salient properties of their instance entities, which play a crucial role in judging whether a specific entity is a plausible head or tail of a specific query relation. Nonetheless, explicit type annotations are typically scarce for entities in non-encyclopedic KGs like Wordnet.

**Reasoning paths.** Since entities in the test graph are not encountered during training, recent state-of-the-art methods (Su et al. 2023, 2024) leverage reasoning paths to make predictions.

**Definition 2 (Reasoning path):** Given a query triple  $(h_q, r_q, t_q)$ , a reasoning path is a sequence of triples that connects head entity  $h_q$  and tail entity  $t_q$ . Formally, we have:

$$p(h_q, t_q) = h_q \xrightarrow{r_0} e_1, \xrightarrow{r_1} e_2, \xrightarrow{r_2} \dots, \xrightarrow{r_{n-1}} t_q, \quad (1)$$

which satisfies  $\forall i, (e_i, r_i, e_{i+1}) \in \mathcal{T}_x$  and  $\forall j, r_j \neq r_q$ .  $\mathcal{T}_x$  denotes  $\mathcal{T}_{train}$  during training and  $\mathcal{T}_{test}$  for model evaluation. However, the existence and quality of reasoning paths are not guaranteed, especially in few-shot scenarios. For instance, our statistics in Figure 2 show that reasoning paths are unavailable for 61 of the 205 query triples in the test split of the FB15k-237 (inductive) dataset.

## Methodology

Figure 3 shows the end-to-end architecture of the proposed CATS framework. To get rid of the reliance on explicit type annotations, we devise a more generalized approach to exploit latent type constraints w.r.t. relations in the Type-Aware Reasoning (TAR) module. In addition, we incorporate relevant neighboring facts and reasoning paths to support triple assessment in the Subgraph Reasoning (SR) module. Furthermore, we discuss our LLM supervised fine-tuning (SFT) strategy, and aggregate our final predictions for query triples.

### Type-aware Reasoning (TAR)

In KGs, entities connected by the same relation often possess similar attributes and characteristics, thereby belonging to the same entity type. Therefore, when determining whether an unknown entity is the head or tail entity of a triple, we need to confirm that the entity conforms to the type dictated by the relation. In practice, type properties are not explicitly available for all entities in the KG. A direct solution is to employ LLMs to annotate type information for each candidate entity, and to summarize common type for entities connected by the particular relation. However, an entity may belong to multiple types with different granularities in different domains. For instance, in Freebase, entity *Nick Mason* corresponds to type *person*, *film actor*, and *book author*. Without clear guidance from explicit type annotations, the type output from the LLM will be unstable.

Instead of explicitly conducting entity typing, we guide the LLM to evaluate the plausibility of a triple by implicitly considering the type relevance between the candidate head/tail entity and other head/tail entities connected by the same relation. Figure 3 (top-right) shows the detailed prompts. For each query triple  $(h_q, r_q, t_q)$ , we first sample a set of  $k$  triples

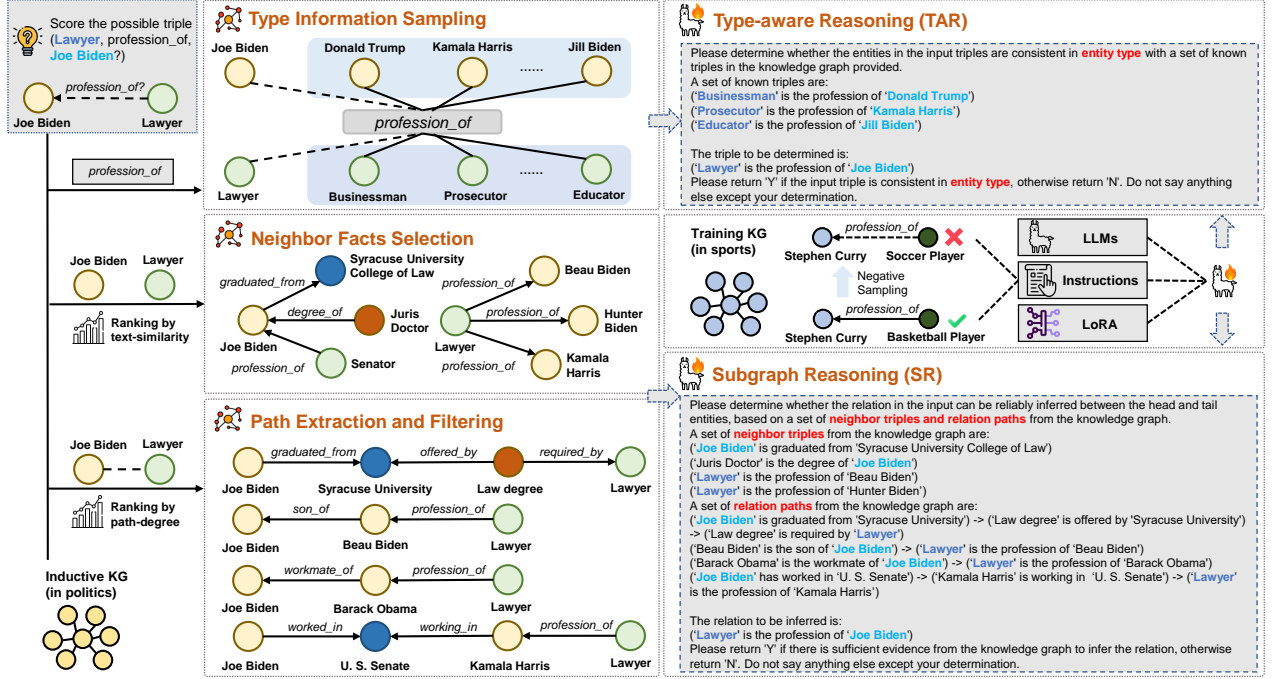


Figure 3: The end-to-end pipeline of the proposed CATS framework.

$\mathcal{S}_r$  with the same relation  $r_q$  as demonstrations. Formally, we have  $\mathcal{S}_r = \{(h, r_q, t) | h, t \in \mathcal{E} \setminus \{h_q, t_q\}\}$ . Then, we linearize these structural triples with the textual labels of entities and relations, which allows the LLM to summarize a latent type triple e.g. (*art work*, *nominated for*, *award*) between the set of head and tail entities. Finally, we provide the linearized query triple to the LLM, and ask the LLM to output “Y” if the query triple is consistent with the same pattern in terms of entity types and “N” otherwise.

We fine-tune the LLM with a contrastive learning strategy. For each triple  $(h, r, t) \in \mathcal{T}_{\text{train}}$ , we construct negative samples by replacing the head or tail entity with a random entity from the training graph  $\mathcal{G}_{\text{train}}$ . Then, we utilize the following loss function for supervised fine-tuning:

$$\mathcal{L}_{\text{TAR}} = - \sum_{(h, r, t) \in \mathcal{T}_{\text{train}}} \log p(\hat{y} = \text{'Y'} | \mathcal{S}_r; \Theta) - \sum_{(h, r, t) \notin \mathcal{T}_{\text{train}}} \log p(\hat{y} = \text{'N'} | \mathcal{S}_r; \Theta), \quad (2)$$

where  $\hat{y}$  denotes the first output token generated by the LLM,  $\Theta$  denotes the model parameters,  $p(\hat{y} = \text{'Y'})$  is the estimated probability that triple  $(h, r, t)$  holds.

### Subgraph Reasoning (SR)

Within KGs, the knowledge about an entity is manifested in its local subgraph (Li et al. 2024). Simply considering the type of an entity is insufficient to assert that the candidate entity should be connected to certain entities with a specific relation. Recent studies (Zha, Chen, and Yan 2022; Su et al.

2023) have shown that reasoning paths provide direct evidence for the existence of a particular relation between two entities. Nevertheless, limited by the power of BERT-like pre-trained language models, each reasoning path has to be independently encoded and considered, which may lead to unreliable relation prediction results. Inspired by the powerful reasoning capabilities of LLMs, we can model the interactions between multiple reasoning paths and neighboring facts of the two entities in a query triple.

**Path extraction and filtering.** Following the convention of (Su et al. 2023, 2024), we leverage breadth-first search (BFS) to extract reasoning paths connecting the two entities of the query triple. We only retain reasoning paths with a length less than or equal to  $n$ , as extraordinarily long paths contribute less to relation prediction. Moreover, we find that some high-frequency relations, such as “*has gender*” and “*has color*”, are meaningless for assessing the existence of other relations (Su et al. 2023). On the contrary, infrequent fine-grained relations such as “*appear in film*” usually offer more precise evidence. Hence, we devise a degree-based filtering mechanism to find out meaningful paths. Specifically, given a reasoning path  $p(h_q, t_q)$ , we count the occurrences  $o_r$  for each relation  $r \in p(h_q, t_q)$  in the training triple set  $\mathcal{T}_{\text{train}}$ . Then, we compute the degree of the reasoning path  $d_{p(h_q, t_q)}$  by summing up the occurrences of all relations within the path. Formally, we have:

$$d_{p(h_q, t_q)} = \sum_{r \in p(h_q, t_q)} o_r = \sum_{r \in p(h_q, t_q)} \sum_{(h, r', t') \in \mathcal{T}_{\text{train}}} \mathbb{1}(r = r'), \quad (3)$$

where  $\mathbb{1}(\cdot)$  denotes the identifier function. Finally, for each query triple  $(h_q, r_q, t_q)$ , we select  $\beta$  reasoning paths  $\mathcal{P}_{(h_q, t_q)}$  with the lowest degrees  $d_{p(h_q, t_q)}$  for assessment, while the others are filtered out.

**Neighboring facts selection.** Since the existence of reasoning paths is not guaranteed, we further adopt neighboring facts of the head and tail entities of the query triple as supplementary contexts. Specifically, for each query triple  $(h_q, r_q, t_q)$ , we first collect supporting triples containing the head entity  $h_q$  or the tail entity  $t_q$  from the training graph.\* Then, we embed the query triple and each supporting triple with “bge-small-en v1.5” (Chen et al. 2024) sentence transformer. To safeguard the accuracy of our assessment against irrelevant neighboring information, we select top  $\sigma$  supporting triples which the embeddings have the highest cosine similarities to the query triple. Formally, we have

$$\mathcal{T}_{h_q} = \arg \max_{(h_q, r, t) \in \mathcal{T}_{\text{train}}} \cos(f_{\text{bge}}(h_q, r, t), f_{\text{bge}}(h_q, r_q, t_q)), \quad (4)$$

$$\mathcal{T}_{t_q} = \arg \max_{(h, r, t_q) \in \mathcal{T}_{\text{train}}} \cos(f_{\text{bge}}(h, r, t_q), f_{\text{bge}}(h_q, r_q, t_q)), \quad (5)$$

where  $\cos(a, b) = \frac{a \cdot b}{\|a\|_2 \cdot \|b\|_2}$ ,  $\mathcal{T}_{h_q}$  and  $\mathcal{T}_{t_q}$  are selected supporting triples for  $h_q$  and  $t_q$ . Similarly, we design appropriate prompts to instruct the LLM to output “Y” if the given query triple can be supported by the aforementioned reasoning paths and neighboring triples and “N” otherwise (Figure 3 bottom). Then, we use the following loss function to fine-tune the LLM:

$$\begin{aligned} \mathcal{L}_{\text{SR}} = & - \sum_{(h, r, t) \in \mathcal{T}_{\text{train}}} \log p(\hat{y} = \text{‘Y’} | \mathcal{P}_{(h_q, t_q)}, \mathcal{T}_{h_q}, \mathcal{T}_{t_q}; \Theta) \\ & - \sum_{(h, r, t) \notin \mathcal{T}_{\text{train}}} \log p(\hat{y} = \text{‘N’} | \mathcal{P}_{(h_q, t_q)}, \mathcal{T}_{h_q}, \mathcal{T}_{t_q}; \Theta). \end{aligned} \quad (6)$$

### Triple Scoring

During the inference stage, the commonly adopted evaluation metrics require the KGC model to rank (or score) each masked triple in the form of  $(h, r, ?)$  or  $(?, r, t)$  with a set of candidate entities. Motivated by the complementary relationship between type properties and structural context, we compose the final scoring of a triple  $s(h, r, t)$  by ensembling the probabilities that the LLM outputs “Y” based on the two kinds of prompts. Formally, we have

$$\begin{aligned} s(h, r, t) = & \frac{1}{2} (p(\hat{y} = \text{‘Y’} | \mathcal{S}_r; \Theta) \\ & + p(\hat{y} = \text{‘Y’} | \mathcal{P}_{(h, t)}, \mathcal{T}_h, \mathcal{T}_t; \Theta)). \end{aligned} \quad (7)$$

## Experiments

### Datasets and Evaluation Metrics

We evaluate our proposed method on three widely adopted benchmark KGs WN18RR (Miller 1995), FB15k-237 (Bollacker et al. 2008), and NELL-995 (Carlson et al. 2010) with

\*In inductive scenarios, supporting triples are selected from the test graph during the evaluation phase.

Dataset	Data splits	$ \mathcal{R}_G $	$ \mathcal{E}_G $	$ \mathcal{T}_G $
WN18RR	train	9	2746	6670
	train-2000	9	1970	2002
	train-1000	9	1362	1001
	test-transductive	7	962	638
	test-inductive	8	922	1991
FB15k-237	train	180	1594	5223
	train-2000	180	1280	2008
	train-1000	180	923	1027
	test-transductive	102	550	492
	test-inductive	142	1093	2404
NELL-995	train	88	2564	10063
	train-2000	88	1346	2011
	train-1000	88	893	1020
	test-transductive	60	1936	968
	test-inductive	79	2086	6621

Table 1: Statistics of datasets.

their transductive and inductive subsets. Following the convention of (Zha, Chen, and Yan 2022; Su et al. 2023, 2024), we evaluate each query triple with 1 ground truth entity and 49 negative candidate entities. For a fair comparison, we use the same dataset splits and negative triples provided by (Zha, Chen, and Yan 2022). The detailed statistics of datasets are available in Table 1. We score the plausibility of the query triple with each candidate entity and rank them in descending order. The performance of our model is evaluated based on two metrics: Mean Reciprocal Rank (MRR) and Hits@1.

### Baselines and Experiment Settings

We compare CATS with embedding-based methods RuleN and Tucker, GNN-based method GraIL, Red-GNN and Adaprop, text-based methods KG-BERT, and path-based methods MINERVA, BERTRL, KRST and the state-of-the-art method APST. We select Qwen2-7B-Instruct (Yang et al. 2024) as our backbone LLM. The experimental results on Llama-3-8B (AI@Meta 2024) and Qwen2-1.5B are also included in our ablation studies. We employ LoRA (Hu et al. 2021) for parameter-efficient fine-tuning, setting the rank to 16 and  $\alpha$ -value to 32. We use the AdamW optimizer (Loshchilov and Hutter 2017) to fine-tune the LLM with a learning rate of  $1e-4$ , a per-device batch size of 2, and a gradient accumulation step of 4 iterations for 1 epoch only. For each query triple, we sample  $k = 3$  supporting triples with the same relation,  $\delta = 6$  reasoning paths and  $\sigma = 6$  neighboring facts. We construct the instruction set by generating 12 negative samples for each positive triple in  $\mathcal{T}_{\text{train}}$ . All experiments are conducted on a server with 2 Intel Xeon Platinum 8358 processors and 8 NVIDIA A100 40G GPUs.<sup>†</sup> On average, CATS takes 2.4 hours for SFT and 1.43s to evaluate and rank a single test sample.

### Main Results

We evaluate the proposed CATS framework on the three datasets in transductive and inductive settings. The “TAR”

<sup>†</sup>Only 2 GPUs are used in our experiments.

Metric	Method	Transductive			Inductive		
		WN18RR	FB15k-237	NELL-995	WN18RR	FB15k-237	NELL-995
MRR	RuleN	0.669	0.674	0.736	0.780	0.462	0.710
	GRAIL	0.676	0.597	0.727	0.799	0.469	0.675
	RED-GNN	0.758	0.737	0.838	0.837	0.852	0.787
	Adaprop	0.790	0.632	0.807	0.795	0.563	0.791
	MINERVA	0.656	0.572	0.592	-	-	-
	TuckER	0.646	0.682	0.800	-	-	-
	KG-BERT	-	-	-	0.547	0.500	0.419
	BERTRL	0.683	0.695	0.781	0.792	0.605	0.808
	KRST	0.899	0.720	0.800	0.890	0.716	0.769
	APST	0.902	0.774	0.801	0.908	0.764	0.769
	CATS (TAR)	0.956	0.812	0.836	0.937	0.834	0.750
	CATS (SR)	0.972	0.829	0.869	<b>0.992</b>	0.875	<b>0.906</b>
	CATS (full)	<b>0.978</b>	<b>0.843</b>	<b>0.885</b>	0.982	<b>0.882</b>	0.861
Hits@1	RuleN	0.646	0.603	0.636	0.745	0.415	0.638
	GRAIL	0.644	0.494	0.615	0.769	0.390	0.554
	RED-GNN	0.712	0.663	0.771	0.798	0.451	0.702
	Adaprop	0.735	0.534	0.725	0.755	0.483	0.678
	MINERVA	0.632	0.534	0.553	-	-	-
	TuckER	0.600	0.615	0.729	-	-	-
	KG-BERT	-	-	-	0.436	0.341	0.244
	BERTRL	0.655	0.620	0.686	0.755	0.541	0.715
	KRST	0.835	0.639	0.694	0.809	0.600	0.649
	APST	0.839	0.694	0.698	0.837	0.643	0.663
	CATS (TAR)	0.922	0.726	0.745	0.888	0.744	0.624
	CATS (SR)	0.951	0.752	0.792	<b>0.984</b>	0.804	<b>0.849</b>
	CATS (full)	<b>0.962</b>	<b>0.776</b>	<b>0.820</b>	0.965	<b>0.805</b>	0.783

Table 2: Transductive and inductive KGC results on WN18RR, FB15k-237 and NELL-995

Metric	Method	Transductive						Inductive					
		WN18RR		FB15k-237		NELL-995		WN18RR		FB15k-237		NELL-995	
		1000	2000	1000	2000	1000	2000	1000	2000	1000	2000	1000	2000
MRR	RuleN	0.567	0.625	0.434	0.577	0.453	0.609	0.681	0.773	0.236	0.383	0.334	0.495
	GRAIL	0.588	0.673	0.375	0.453	0.292	0.436	0.652	0.799	0.380	0.432	0.458	0.462
	RED-GNN	0.144	0.301	0.250	0.519	0.296	0.469	0.818	0.826	0.482	0.503	0.692	0.737
	Adaprop	0.143	0.299	0.259	0.451	0.292	0.478	0.786	0.794	0.527	0.546	0.702	0.739
	MINERVA	0.125	0.268	0.198	0.364	0.182	0.322	-	-	-	-	-	-
	TuckER	0.258	0.448	0.457	0.601	0.436	0.577	-	-	-	-	-	-
	KG-BERT	-	-	-	-	-	-	0.471	0.525	0.431	0.460	0.406	0.406
	BERTRL	0.662	0.673	0.618	0.667	0.648	0.693	0.765	0.777	0.526	0.565	0.736	0.744
	KRST	0.871	0.882	0.696	0.701	0.743	<b>0.781</b>	0.886	0.878	0.679	0.680	0.745	0.738
	APST	0.874	0.880	0.724	0.753	<b>0.745</b>	0.767	0.894	0.879	0.697	0.747	0.765	0.747
	CATS (TAR)	0.925	0.936	0.774	0.800	0.737	0.751	0.869	0.889	0.796	0.813	0.697	0.712
	CATS (SR)	0.879	0.926	0.734	0.780	0.680	0.679	0.898	<b>0.956</b>	0.847	0.876	<b>0.854</b>	<b>0.871</b>
	CATS (full)	<b>0.932</b>	<b>0.952</b>	<b>0.787</b>	<b>0.824</b>	0.741	0.762	<b>0.922</b>	0.953	<b>0.862</b>	<b>0.877</b>	0.808	0.829
Hits@1	RuleN	0.548	0.605	0.374	0.508	0.365	0.501	0.649	0.737	0.207	0.344	0.282	0.418
	GRAIL	0.489	0.633	0.267	0.352	0.198	0.342	0.516	0.769	0.273	0.351	0.295	0.298
	RED-GNN	0.108	0.267	0.196	0.449	0.214	0.379	0.777	0.785	0.380	0.422	0.549	0.605
	Adaprop	0.107	0.260	0.188	0.366	0.218	0.386	0.741	0.749	0.425	0.451	0.580	0.630
	MINERVA	0.106	0.248	0.170	0.324	0.152	0.284	-	-	-	-	-	-
	TuckER	0.230	0.415	0.407	0.529	0.392	0.520	-	-	-	-	-	-
	KG-BERT	-	-	-	-	-	-	0.364	0.404	0.288	0.317	0.236	0.236
	BERTRL	0.621	0.637	0.517	0.583	0.526	0.582	0.713	0.731	0.441	0.493	0.622	0.628
	KRST	0.790	0.810	0.611	0.602	0.628	<b>0.678</b>	0.811	0.793	0.537	0.524	0.637	0.629
	APST	0.798	0.813	0.632	0.665	0.640	0.663	0.822	0.798	0.561	0.627	0.654	0.637
	CATS (TAR)	0.871	0.888	0.678	0.714	0.628	0.636	0.774	0.811	0.680	0.707	0.584	0.596
	CATS (SR)	0.802	0.874	0.631	0.681	0.574	0.563	0.824	0.915	0.756	0.800	<b>0.767</b>	<b>0.790</b>
	CATS (full)	<b>0.887</b>	<b>0.918</b>	<b>0.702</b>	<b>0.750</b>	<b>0.648</b>	0.664	<b>0.864</b>	<b>0.923</b>	<b>0.776</b>	<b>0.802</b>	0.713	0.746

Table 3: Few-shot KGC results on WN18RR, FB15k-237 and NELL-995



and “SR” variants of CATS severally utilize the probabilities generated by corresponding modules to score and rank each candidate entity. Experimental Results in Table 2 demonstrate that the CATS (full) significantly and consistently outperforms all baseline methods. Most notably, CATS (full) achieves *absolute Hits@1 improvements* of 12.8%, 16.2%, and 6.8% on the WN18RR, FB15k-237, and NELL-995 datasets under an *inductive setting*. Correspondingly, the improvements in *transductive* scenarios, namely 12.3%, 8.2%, and 9.1%, are also remarkable.

Among all CATS’ variants, the majority of best results (4 out of 6 cases) are achieved by the “full” variant, which exhibits the importance of considering latent type constraints and subgraph contexts. In comparison, the improvements observed with the TAR variant are less pronounced, indicating that while matching entity types may help filter out irrelevant entities, it is insufficient for delivering accurate relation predictions. Moreover, the TAR variant demonstrates improved effectiveness in transductive settings. The better performance can be credited to SFT, which enhances the LLM’s understanding of the type properties of known entities. However, the training graph does not contain any contexts for unseen entities. Therefore, the LLM may not be able to precisely infer types for some of them, thereby compromising the performance of TAR in inductive scenarios. Conversely, the SR variant benefits from neighboring facts and reasoning paths sampled from the test graph, providing relevant contextual information for unseen entities and allowing it to achieve state-of-the-art performance on WN18RR and NELL-995 datasets in inductive settings.

## Ablation Studies

We examine the effectiveness of each component of the proposed CATS framework in different settings by answering the following research questions (RQs).

**RQ1: Can CATS generate plausible inference results in few-shot scenarios?** Table 3 shows the experiment results on the three datasets with 1000 and 2000 triples in the training graph. In general, CATS achieves significant improvements for 10 out of 12 cases in terms of MRR compared to state-of-the-art methods. For the remaining cases, the performance gap to the best baseline method is marginal. Considering the three variants of CATS, TAR outperforms SR in transductive settings, while SR performs better in inductive scenarios. The disparity in performance can be attributed to the following reasons: Reducing the number of triples in the training graph significantly decreases the average degree of each entity. Consequently, it becomes challenging to sample a sufficient number of neighboring facts and reasoning paths for entities in the query triple, which diminishes the effectiveness of the SR variant. However, in most KGs, the number of relations is considerably smaller than the number of entities ( $|\mathcal{R}| \ll |\mathcal{E}|$ ). Hence, we are still able to retrieve sufficient triples with the same relation from the training graph, which sustains the desirable performance of the TAR variant, and ensures the robustness of the the proposed framework. In inductive scenarios, neighboring facts and reasoning paths are sampled from the supplementary test graph.

Hence, reducing training triples has subtle negative impacts on the effectiveness of the SR variant.

One may notice that CATS does not achieve stat-of-the-art performance on the NELL-995 dataset in transductive settings. This is attributed to the higher number of triples in the NELL-995 training graph. Selecting the subset of 1000 or 2000 triples results in an extremely sparse graph structure. Without external knowledge, the training graph may fail to provide sufficient support for the inference of certain triples, thereby lowering the performance ceiling. For the same reason, GNN-based methods such as RED-GNN (Zhang and Yao 2022) and Adaprop (Zhang et al. 2023) exhibit significant performance drops in few-shot scenarios. Moreover, KRST (Su et al. 2023) and APST (Su et al. 2024) take advantage of extra knowledge from entity descriptions, allowing them to catch up with CATS.

**RQ2: Do reasoning paths and neighboring facts improve the inference performance?** From the experimental results in Table 4, we observe that both reasoning paths and neighboring facts play a crucial role in enhancing inference performance. However, the contribution of neighboring triples is more pronounced. This reconfirms the key shortcoming of path-based methods, which struggle to assess query triples without suitable paths. In comparison, neighboring triples guarantee CATS’s robustness and generality. Moreover, the performance decline resulting from the removal of the path filtering step emphasizes the effectiveness of the proposed degree-based filtering mechanism, further reaffirming that irrelevant reasoning paths may misdirect the evaluation of query triples.

**RQ3: Can we simply attribute the performance improvement to extra knowledge inherent in the LLM?** We conduct additional experiments by presenting the query triple and corresponding entities to the LLM, prompting the model to make judgments with its internal knowledge. However, experimental results in Table 5 indicate that the LLM fails to make reliable assessments on KG triples in such a zero-shot setting (see Qwen2-7B w/o. all), showing that CATS does not benefit from the LLM’s internal knowledge. Furthermore, the pre-trained LLM cannot adequately comprehend the contextual information (e.g., paths and triples) outlined in our prompts without proper guidance (see Qwen2-7B w/o SFT). The unsatisfactory performance underscores the substantial semantic gap between natural language sentences and KG triples, emphasizing the importance of SFT.

**RQ4: Can we simply attribute the performance improvements to the power of LLMs?** We conduct an extra experiment by directly fine-tuning the LLM to evaluate the plausibility of query triples based solely on the triple itself (see Table 5 w/o. TAR & SR). The significant performance decline indicates the following: despite possessing enhanced semantic understanding capabilities, the LLM does not inherently know the appropriate method to evaluate a triple. On the one hand, the backbone LLM in this variant cannot recognize the importance of type relevance between the target entity and entities connected by the same relation. On the other hand, the process of SFT is insufficient to inject knowl-

Metric	Configuration	Transductive			Inductive		
		WN18RR	RB15k-237	NELL-995	WN18RR	RB15k-237	NELL-995
MRR	CATS (SR)	<b>0.972</b>	<b>0.829</b>	<b>0.869</b>	<b>0.992</b>	<b>0.875</b>	<b>0.906</b>
	- w/ NF only	0.962	0.824	0.861	0.971	0.873	0.895
	- w/ RP (filt.) only	0.960	0.800	0.773	0.968	0.851	0.829
	- w/ RP only	0.954	0.801	0.769	0.965	0.844	0.821
Hits@1	CATS (SR)	<b>0.951</b>	<b>0.752</b>	<b>0.792</b>	<b>0.984</b>	<b>0.804</b>	<b>0.849</b>
	- w/ NF only	0.932	0.747	0.776	0.944	0.793	0.830
	- w/ RP (filt.) only	0.932	0.714	0.666	0.941	0.768	0.729
	- w/ RP only	0.929	0.714	0.660	0.937	0.756	0.724

Table 4: Transductive and Inductive KGC performance (in MRR) on the SR variant with different structural contexts. Here NF denotes neighboring facts, RP denotes reasoning paths, and (filt.) refers to the application of path filtering.

Metric	LLM & Config.	Transductive			Inductive		
		WN18RR	RB15k-237	NELL-995	WN18RR	RB15k-237	NELL-995
MRR	Previous SOTA	0.902	0.774	0.801	0.908	0.764	0.808
	Llama-3-8B (full)	0.965	0.802	0.867	0.956	0.862	0.837
	Qwen2-1.5B (full)	0.966	0.804	0.877	0.948	0.814	0.835
	Qwen2-7B (comb.)	0.967	0.830	0.861	<b>0.985</b>	0.859	<b>0.885</b>
	Qwen2-7B (full)	<b>0.978</b>	<b>0.843</b>	<b>0.885</b>	0.982	<b>0.882</b>	0.861
	- w/o. TAR & SR	0.947	0.800	0.811	0.901	0.814	0.710
	- w/o. SFT	0.225	0.197	0.177	0.207	0.179	0.208
	- w/o. all	0.199	0.140	0.156	0.174	0.144	0.138
Hits@1	Previous SOTA	0.839	0.694	0.698	0.837	0.643	0.715
	Llama-3-8B (full)	0.942	0.718	0.794	0.931	0.783	0.750
	Qwen2-1.5B (full)	0.940	0.720	0.809	0.912	0.722	0.748
	Qwen2-7B (comb.)	0.941	0.750	0.779	<b>0.971</b>	0.778	<b>0.811</b>
	Qwen2-7B (full)	<b>0.962</b>	<b>0.776</b>	<b>0.820</b>	0.965	<b>0.805</b>	0.783
	- w/o. TAR & SR	0.905	0.705	0.704	0.824	0.707	0.572
	- w/o. SFT	0.132	0.103	0.081	0.101	0.093	0.009
	- w/o. all	0.152	0.093	0.103	0.128	0.093	0.087

Table 5: Transductive and Inductive KGC performance (in MRR) with different LLMs and configurations.

edge stored in KGs into the LLM. This reaffirms the significance of providing relevant guidance and structural contexts in the KGC task. Furthermore, we investigate whether combining prompts from the two reasoning modules improves the model performance (see Table 5 (comb.)). Our experiments show that the current adopted separated setting achieves better results in most of the (4 out of 6) cases.

**RQ5: How does the selection of backbone LLM affect the experimental results?** In Table 5, we evaluate the performance of the proposed CATS framework on three LLMs, namely Llama3-8B, Qwen2-1.5B, and Qwen2-7B. The experimental results show that CATS significantly and consistently surpasses the state-of-the-art method with all these LLMs, demonstrating its broad effectiveness across different model scales and architectures. Most notably, CATS can still achieve desirable results with an 1.5B model, which significantly reduces the average inference time from 1.43s to 0.51s, showcasing a perfect balance between performance and efficiency. Furthermore, the performance improvements

observed with the Qwen2-7B model indicate that increasing the model size is likely to yield better results. Since comparison among different LLMs is not the primary focus, this paper does not explore the performance of CATS with larger LLMs due to time and resource constraints.

## Conclusion

In this paper, we propose CATS, a novel context-aware approach for knowledge graph completion. CATS is designed to guide the LLM to assess the plausibility of query triples based on latent type constraints, selected reasoning paths, and relevant neighboring facts. With sufficient guidance from proper prompts and SFT, CATS achieves state-of-the-art performance in transductive, inductive, and few-shot scenarios, showing its robustness and generality. Overall, CATS demonstrates the potential of leveraging LLMs in conducting knowledge-intensive reasoning tasks on structural data. In the future, we aim to incorporate LLMs in more complicated KG-related tasks such as complex query answering.



## Acknowledgment

The work described in this paper was partially supported by the Research Grants Council of the Hong Kong Special Administrative Region, China (CUHK 14222922, RGC GRF 2151185).

## References

- AI@Meta. 2024. Llama 3 Model Card.
- Bollacker, K.; Evans, C.; Paritosh, P.; Sturge, T.; and Taylor, J. 2008. Freebase: A Collaboratively Created Graph Database for Structuring Human Knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, 1247–1250. New York, NY, USA: Association for Computing Machinery. ISBN 9781605581026.
- Bordes, A.; Usunier, N.; Garcia-Durán, A.; Weston, J.; and Yakhnenko, O. 2013. Translating Embeddings for Modeling Multi-relational Data. In Burges, C. J. C.; Bottou, L.; Welling, M.; Ghahramani, Z.; and Weinberger, K. Q., eds., *Advances in Neural Information Processing Systems 26*, volume 26. Curran Associates, Inc.
- Cao, Z.; Xu, Q.; Yang, Z.; Cao, X.; and Huang, Q. 2022. Geometry Interaction Knowledge Graph Embeddings. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(5): 5521–5529.
- Carlson, A.; Betteridge, J.; Kisiel, B.; Settles, B.; Hruschka, E. R.; and Mitchell, T. M. 2010. Toward an architecture for never-ending language learning. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*, AAAI'10, 1306–1313. AAAI Press.
- Chen, J.; He, H.; Wu, F.; and Wang, J. 2021. Topology-Aware Correlations Between Relations for Inductive Link Prediction in Knowledge Graphs. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(7): 6271–6278.
- Chen, J.; Xiao, S.; Zhang, P.; Luo, K.; Lian, D.; and Liu, Z. 2024. BGE M3-Embedding: Multi-Lingual, Multi-Functionality, Multi-Granularity Text Embeddings Through Self-Knowledge Distillation. arXiv:2402.03216.
- Das, R.; Dhuliawala, S.; Zaheer, M.; Vilnis, L.; Durugkar, I.; Krishnamurthy, A.; Smola, A.; and McCallum, A. 2018. Go for a Walk and Arrive at the Answer: Reasoning Over Paths in Knowledge Bases using Reinforcement Learning. In *International Conference on Learning Representations*.
- Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Burstein, J.; Doran, C.; and Solorio, T., eds., *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 4171–4186. Minneapolis, Minnesota: Association for Computational Linguistics.
- Hao, J.; Chen, M.; Yu, W.; Sun, Y.; and Wang, W. 2019. Universal Representation Learning of Knowledge Bases by Jointly Embedding Instances and Ontological Concepts. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, 1709–1719. New York, NY, USA: Association for Computing Machinery. ISBN 9781450362016.
- Hu, E. J.; Shen, Y.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; and Chen, W. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.
- Kim, J.; Park, S.; Kwon, Y.; Jo, Y.; Thorne, J.; and Choi, E. 2023. FactKG: Fact Verification via Reasoning on Knowledge Graphs. In Rogers, A.; Boyd-Graber, J.; and Okazaki, N., eds., *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 16190–16206. Toronto, Canada: Association for Computational Linguistics.
- Li, M.; Hu, M.; King, I.; and Leung, H.-f. 2024. The Integration of Semantic and Structural Knowledge in Knowledge Graph Entity Typing. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, 6625–6638. Mexico City, Mexico: Association for Computational Linguistics.
- Liao, R.; Jia, X.; Li, Y.; Ma, Y.; and Tresp, V. 2024. GenTKG: Generative Forecasting on Temporal Knowledge Graph with Large Language Models. arXiv:2310.07793.
- Liu, Y.; Tian, X.; Sun, Z.; and Hu, W. 2024. Fine-tuning Generative Large Language Models with Discrimination Instructions for Knowledge Graph Completion. arXiv:2407.16127.
- Loshchilov, I.; and Hutter, F. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.
- Ma, Y.; Song, Z.; Hu, X.; Li, J.; Zhang, Y.; and King, I. 2023. Graph Component Contrastive Learning for Concept Relatedness Estimation. In AAAI, 13362–13370. AAAI Press.
- Meilicke, C.; Fink, M.; Wang, Y.; Ruffinelli, D.; Gemulla, R.; and Stuckenschmidt, H. 2018. Fine-Grained Evaluation of Rule- and Embedding-Based Systems for Knowledge Graph Completion. In *The Semantic Web – ISWC 2018: 17th International Semantic Web Conference, Monterey, CA, USA, October 8–12, 2018, Proceedings, Part I*, 3–20. Berlin, Heidelberg: Springer-Verlag. ISBN 978-3-030-00670-9.
- Miller, G. A. 1995. WordNet: a lexical database for English. *Commun. ACM*, 38(11): 39–41.
- Schlichtkrull, M.; Kipf, T.; Bloem, P.; Berg, R.; Titov, I.; and Welling, M. 2018. *Modeling Relational Data with Graph Convolutional Networks*, 593–607. ISBN 978-3-319-93416-7.
- Song, Z.; Zhang, Y.; and King, I. 2023a. Optimal Block-wise Asymmetric Graph Construction for Graph-based Semi-supervised Learning. In *NeurIPS*.
- Song, Z.; Zhang, Y.; and King, I. 2023b. Towards Fair Financial Services for All: A Temporal GNN Approach for Individual Fairness on Transaction Networks. In *CIKM*, 2331–2341. ACM.
- Su, Z.; Wang, D.; Miao, C.; and Cui, L. 2023. Multi-aspect explainable inductive relation prediction by sentence transformer. In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence and Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence and*

- Thirteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'23/IAAI'23/EAAI'23. AAAI Press. ISBN 978-1-57735-880-0.
- Su, Z.; Wang, D.; Miao, C.; and Cui, L. 2024. Anchoring Path for Inductive Relation Prediction in Knowledge Graphs. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(8): 9011–9018.
- Sun, J.; Xu, C.; Tang, L.; Wang, S.; Lin, C.; Gong, Y.; Ni, L. M.; Shum, H.-Y.; and Guo, J. 2024. Think-on-Graph: Deep and Responsible Reasoning of Large Language Model on Knowledge Graph. arXiv:2307.07697.
- Sun, Z.; Deng, Z.-H.; Nie, J.-Y.; and Tang, J. 2019. RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space. In *International Conference on Learning Representations*.
- Teru, K. K.; Denis, E. G.; and Hamilton, W. L. 2020. Inductive relation prediction by subgraph reasoning. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org.
- Vashishth, S.; Sanyal, S.; Nitin, V.; and Talukdar, P. 2020. Composition-based Multi-Relational Graph Convolutional Networks. In *International Conference on Learning Representations*.
- Wang, H.; Zhao, M.; Xie, X.; Li, W.; and Guo, M. 2019. Knowledge Graph Convolutional Networks for Recommender Systems. In *The World Wide Web Conference*, WWW '19, 3307–3313. New York, NY, USA: Association for Computing Machinery. ISBN 9781450366748.
- Wei, Y.; Huang, Q.; Zhang, Y.; and Kwok, J. 2023. KICGPT: Large Language Model with Knowledge in Context for Knowledge Graph Completion. In Bouamor, H.; Pino, J.; and Bali, K., eds., *Findings of the Association for Computational Linguistics: EMNLP 2023*, 8667–8683. Singapore: Association for Computational Linguistics.
- Xiong, W.; Hoang, T.; and Wang, W. Y. 2017. DeepPath: A Reinforcement Learning Method for Knowledge Graph Reasoning. In Palmer, M.; Hwa, R.; and Riedel, S., eds., *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 564–573. Copenhagen, Denmark: Association for Computational Linguistics.
- Yang, A.; Yang, B.; Hui, B.; Zheng, B.; Yu, B.; Zhou, C.; Li, C.; Li, C.; Liu, D.; Huang, F.; Dong, G.; Wei, H.; Lin, H.; Tang, J.; Wang, J.; Yang, J.; Tu, J.; Zhang, J.; Ma, J.; Yang, J.; Xu, J.; Zhou, J.; Bai, J.; He, J.; Lin, J.; Dang, K.; Lu, K.; Chen, K.; Yang, K.; Li, M.; Xue, M.; Ni, N.; Zhang, P.; Wang, P.; Peng, R.; Men, R.; Gao, R.; Lin, R.; Wang, S.; Bai, S.; Tan, S.; Zhu, T.; Li, T.; Liu, T.; Ge, W.; Deng, X.; Zhou, X.; Ren, X.; Zhang, X.; Wei, X.; Ren, X.; Liu, X.; Fan, Y.; Yao, Y.; Zhang, Y.; Wan, Y.; Chu, Y.; Liu, Y.; Cui, Z.; Zhang, Z.; Guo, Z.; and Fan, Z. 2024. Qwen2 Technical Report. arXiv:2407.10671.
- Yao, L.; Mao, C.; and Luo, Y. 2019. KG-BERT: BERT for knowledge graph completion. *arXiv preprint arXiv:1909.03193*.
- Zha, H.; Chen, Z.; and Yan, X. 2022. Inductive Relation Prediction by BERT. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(5): 5923–5931.
- Zhang, Y.; and Yao, Q. 2022. Knowledge Graph Reasoning with Relational Digraph. In *Proceedings of the ACM Web Conference 2022*, WWW '22, 912–924. New York, NY, USA: Association for Computing Machinery. ISBN 9781450390965.
- Zhang, Y.; Zhou, Z.; Yao, Q.; Chu, X.; and Han, B. 2023. AdaProp: Learning Adaptive Propagation for Graph Neural Network based Knowledge Graph Reasoning. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '23, 3446–3457. New York, NY, USA: Association for Computing Machinery. ISBN 9798400701030.
- Zhao, Y.; Qi, J.; Liu, Q.; and Zhang, R. 2021. WGCN: Graph Convolutional Networks with Weighted Structural Features. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, 624–633. New York, NY, USA: Association for Computing Machinery. ISBN 9781450380379.