

Ontology-Enhanced Knowledge Graph Completion using Large Language Models

Wenbin Guo¹[0009-0006-1649-4131], Xin Wang¹[0000-0001-9651-0651], Jiaoyan Chen²[0000-0003-4643-6750], Zhao Li¹[0000-0003-3996-255X], and Zirui Chen¹[0000-0002-7916-6745]

- ¹ Tianjin University, College of Intelligence and Computing, Tianjin 300350, China
{wenff, wangx, lizh, zrchen}@tju.edu.cn
- ² University of Manchester, Department of Computer Science, Manchester M13 9PL, United Kingdom
jiaoyan.chen@manchester.ac.uk

Abstract. Large Language Models (LLMs) have been extensively adopted in Knowledge Graph Completion (KGC), showcasing significant research advancements. However, as black-box models driven by deep neural architectures, current LLM-based KGC methods rely on implicit knowledge representation with parallel propagation of erroneous knowledge, thereby hindering their ability to produce conclusive and decisive reasoning outcomes. We aim to integrate neural-perceptual structural information with ontological knowledge, leveraging the powerful capabilities of LLMs to achieve a deeper understanding of the intrinsic logic of the knowledge. We propose an **ontology enhanced KGC** method using LLMs — OL-KGC. It first leverages neural perceptual mechanisms to effectively embed structural information into the textual space, and then uses an automated extraction algorithm to retrieve ontological knowledge from the knowledge graphs (KGs) that needs to be completed, which is further transformed into a textual format comprehensible to LLMs for providing logic guidance. We conducted extensive experiments on three widely-used benchmarks — FB15K-237, UMLS and WN18RR. The experimental results demonstrate that OL-KGC significantly outperforms existing mainstream KGC methods across multiple evaluation metrics, achieving state-of-the-art performance. The implementation of the algorithm and related data have been open-sourced³.

Keywords: Knowledge Graph Completion · Ontology · Large Language Model · Neural-symbolic Integration

1 Introduction

Knowledge Graphs (KGs) [14] represent a foundational paradigm for structured knowledge representation, serving as a cornerstone in the progression from perceptual intelligence to cognitive intelligence. They underpin numerous critical

³ <https://github.com/xiumu-gg/OL-KGC>

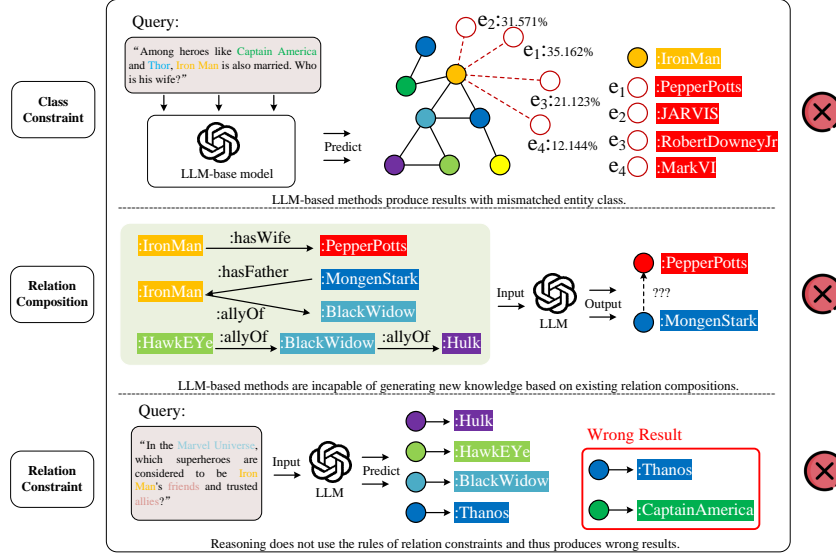


Fig. 1. LLM-based methods that fail to effectively utilize ontological knowledge.

applications across diverse domains, including information retrieval [26], question answering [11], and recommendation [33]. KGs often organize real-world knowledge through triples in form of (head entity, relation, tail entity), enabling the structured representation of relational facts. Their expressive power and reasoning capabilities are further augmented by the incorporation of ontological frameworks and symbolic logic reasoning methods. However, KGs often suffer from incompleteness due to various reasons. For example, algorithms designed to extract structured information from data often struggle to comprehensively capture the full spectrum of information in large datasets. The Knowledge Graph Completion (KGC) is to address this challenge by leveraging existing knowledge and structural patterns to (iteratively) infer missing knowledge [32].

Due to the powerful capabilities of Large Language Models (LLMs), the field of KGC has undergone a significant transformation. Recent LLM-based methods to KGC leverage the advanced contextual and semantic understanding capabilities of LLMs to enhance the precision and completeness of extracted entity and relation information, thereby improving the overall quality and reliability of results[3]. These methods can be broadly categorized into two approaches: 1) leveraging existing paradigms such as zero-shot reasoning [2] and instruction tuning [43] to translate structured KG predictions into LLM-based text predictions; and 2) integrating non-textual structural information from KGs into LLMs through correspondences between KG embeddings and LLM tokens. While these methods are capable of handling complex nonlinear relations, the absence of a mechanism for factual verification does not ensure the veracity and reliability of the generated content [37].

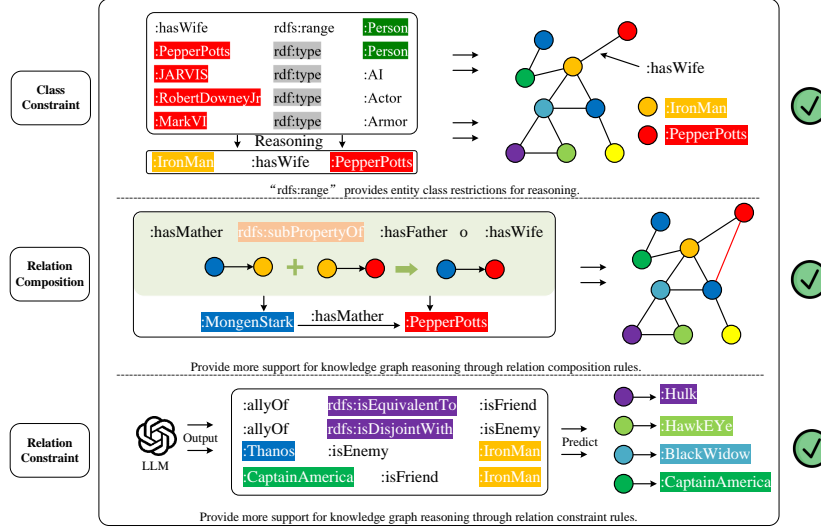


Fig. 2. The role of ontological knowledge in reasoning.

As depicted in Figure 1, LLM-based methods often fail to effectively leverage ontological information when verifying the accuracy of test triples, primarily in the following three aspects. First, these methods exhibit a lack of sensitivity to entity classes, and are unable to fully utilize the constraints imposed by relation range and domain to filter out the interference from erroneous triples. Second, they are incapable of performing reasoning based on existing relational paths within the KGs, thereby failing to accurately identify newly generated triples through relation compositions. Third, these methods cannot apply complex ontological knowledge, such as equivalence and disjointness relations, and are prone to deriving conclusions that are completely contrary to the facts during the reasoning process. The underlying cause of this phenomenon is that LLMs do not incorporate an explicit symbolic reasoning mechanism during the prediction process. Instead, they rely excessively on the semantic correlation between entities, usually lacking fundamental logical reasoning capabilities. Although previous studies [40] have attempted to address this issue by injecting rules into KG embeddings and constraining the quality of rule embeddings through loss functions, the results have been unsatisfactory. Therefore, how to utilize the extensive knowledge domain covered by LLMs and their ability to capture complex data patterns to achieve a deep integration of vector space and symbolic space is the core issue of this research.

Given that a KG for completion is often short of ontological knowledge, this study first proposes an LLM-based automatic extraction algorithm to derive ontological knowledge from the KG that needs to be completed. As shown in Figure 2, these extracted ontological knowledge cover the following key information:

- The range and domain of each relation, which can effectively validate triples according to the classes of their entities.
- The relation compositions based on existing relational paths, which can enabling the inference of new knowledge.
- Complex relation constraints beyond relation domain and range, such as relation equivalence and disjointness.

We rewrite the extracted ontological knowledge, which is originally represented as RDF triples, into text so as to ensure that the LLM can fully comprehend their semantics. Meanwhile, we align the neural KG embeddings in the vector space with the text space tokens in the LLM, and incorporate the symbolic KG structural information and ontological knowledge into the LLM based on its tuning, thereby achieving effective neural-symbolic integration. To the best of our knowledge, this work is the first to systematically integrate explicit ontological knowledge — encompassing class constraints and relation compositions — into LLMs through the joint exploitation of textual inference and structured symbolic logic. Experimental results demonstrate that OL-KGC achieves SOTA performance in the field of KGC. In addition, we conducted ablation studies to individually assess the contributions of KG structural information and ontological knowledge. Furthermore, we evaluated the impact of different types of ontological knowledge on the model’s reasoning performance.

The main contributions of this paper can be summarized as follows:

- We propose a novel method for integrating neural KG embeddings, and symbolic KG structural information and ontological knowledge based on LLMs and their tuning, achieving effective neural-symbolic integration and significantly enhancing LLM reasoning for KGC.
- We design an LLM-based automatic ontology extraction algorithm that can automate the ontology framework design for general KGs. This approach improves the semantic richness and the consistency and coherence of the relations and overall organization within KGs.
- Through extensive experiments, we demonstrate the performance superiority of OL-KGC in the KGC task. Additionally, we release all related code and data to provide a reproducible benchmark for future research.

2 Related Works

2.1 Traditional Embedding-based KGC Models

Traditional KGC models often adopt geometric embedding, assuming relational consistency, where relations imply a close alignment of entity vectors through specific operations [17]. Models enhance embedding quality using various operator spaces [6], such as TransE [1], which treats relations as translations between entity vectors. Subsequent studies have leveraged advanced natural language processing techniques to model complex relational patterns, including hyperplanes [34], Riemannian spheres [22], relational mapping spaces [15], multi-mapping spaces [9], rotational spaces [29], and convolutional spaces [8,23,18].

The introduction of attention mechanisms has enhanced the flexibility and efficiency of KGC models in deep learning [5]. RelAtt refines embeddings using a relation-aware attention mechanism [28], HRAN aggregates features along relational paths [21], and KGT5 and HittER leverage Transformers for scalable and context-aware representation learning [27,4]. Despite their success, embedding-based models still struggle with distinguishing semantically similar but class-inconsistent entities [10].

2.2 PLM-based and LLM-based KGC Models

PLM-based KGC models leverage the implicit knowledge within Pre-trained Language Models (PLMs) which are usually encoder-based and the structured knowledge in KGs to predict missing triples [36]. KG-BERT is a pioneering work in this domain [38], which takes the textual information of triples as input and predicts the plausibility of triples through fine-tuning the BERT model. However, integrating the implicit knowledge of PLMs with KGs has always been a key challenge for such methods.

In contrast to leveraging contextual embeddings from PLMs to represent entities and relations in KGs for predicting missing triples, LLM-based KGC models exploit the more powerful and flexible text-generation capabilities of LLMs to directly generate missing triples or provide more comprehensive reasoning over KGs by generating textual explanations or intermediate steps. Recently, LLM-based KGC methods have been primarily categorized into two types [24]:

The first approach leverages paradigms like zero-shot reasoning and instruction tuning to translate structured KG information into text-based predictions via LLMs [37]. For example, MuKDC uses LLMs to generate subgraph-based questions for KGC [19], while KLR-KGC enhances LLM reasoning by retrieving analogical and subgraph knowledge from KGs [16].

The second approach incorporates structured KG information into LLMs by aligning KG embeddings with LLM tokens, improving their understanding and representation of KGs. TEA-GLM employs LLMs as zero-shot learners for graph machine learning tasks [31]. KGEitor converts paths into Chain-of-Thought prompts for multi-hop link prediction [6]. KoPA, the current state-of-the-art, introduces a knowledge prefix adapter to align structural embeddings with LLM tokens, effectively integrating KG information into LLMs [42].

2.3 Neuro-Symbolic Integration for KGC

Besides embedding and language model-based methods, there are some neuro-symbolic methods for KGC that focus on the integration of symbolic reasoning and neural networks for enhancing reasoning capabilities and interpretability, often with mutual feedback between symbolic and neural components.

Neuro-symbolic approaches leverage logical rules to expand reliable data and enhance embedding quality [20]. NeoEA aligns entity embeddings with ontological knowledge to reduce distribution discrepancies [12], while SymRITa employs symbolic rule graphs and a logical transformer to address predicate non-

commutativity [25]. Other methods enforce logical constraints via loss functions, such as IterE, which integrates rules into embeddings [41]. GCR uses graph convolution networks with rule-enhanced embeddings [7], and KALE integrates embeddings and logical rules through a loss function [13].

However, these methods compromise the precision and interpretability of symbolic reasoning, facing challenges in integrating vector and symbolic spaces. The emergence of LLMs, which encapsulate extensive knowledge and complex patterns, presents new opportunities for this integration [35].

3 Notations and Preliminaries

A KG is represented as $\mathcal{G} = \{\mathcal{E}, \mathcal{R}, \mathcal{T}, \mathcal{H}, \mathcal{D}\}$, where \mathcal{E} and \mathcal{R} denote the sets of entities and relations, respectively, and $\mathcal{T} = \{(h, r, t) \mid h, t \in \mathcal{E}, r \in \mathcal{R}\}$ represents the set of triples. \mathcal{H} refers to the KG’s ontological information, represented as a set of RDF triples. The ontological information we consider includes the following types:

- **Relation Domain and Range Constraints:** For a given entity $e \in \mathcal{E}$, the entity class constraint stipulates that e belongs to a specific class a . Furthermore, if e also belongs to a superclass class of a_s , it is denoted as $a_s \subseteq a$, indicating that a inherits the attributes of a_s . The properties `:range` and `:domain` can be utilized to restrict the classes of the head entity h and the tail entity t of a specific relation r . For instance, for the relation `:hasWife`, the `:range` and `:domain` property can restrict the head entity h to the class `:Person` and the tail entity t to the class `:Person`. Given that the entity `:PepperPotts` belongs to the class `:Women`, which is a subclass of `:Person`, the entity `:PepperPotts` complies with the `:range` restriction of the relation `:hasWife`, thereby validating the `:range` constraint.
- **Relation Compositions:** Relation compositions enable the inference of new relations within a \mathcal{G} . For instance, given two RDF triples `(:MongenStark :hasFather :IronMan)` and `(:IronMan :hasWife :PepperPotts)`, a new relation can be inferred through the relation composition `(:hasMother rdfs:subPropertyOf :hasFather o :hasWife)`. This results in the new inference `(:MongenStark :hasMother :PepperPotts)`, thereby deducing the previously unknown fact that PepperPotts is the mother of MongenStark based on the existing knowledge in the \mathcal{G} .
- **Complex Relation Constraints:** Complex relations, such as equivalence and disjointness relation, are employed to constrain the results of inference. For equivalence relations, the \mathcal{G} is expanded. For example, given that `:allyOf` and `:isFriend` are equivalent relations, the triple `(:IronMan :allyOf :CaptainAmerica)` can be expanded to yield the new triple `(:IronMan :isFriend :CaptainAmerica)`. For disjointness relations, the inference results are validated. For instance, given that `:isEnemy` and `:isFriend` are disjointness relation, if the triple `(:IronMan :isEnemy :Thanos)` is true, then the triple `(:IronMan :isFriend :Thanos)` must be false.

\mathcal{D} denotes the descriptive information of each entity and relation, where each entity and relation is uniquely identified in the real data. Specifically, we use $\mathcal{D}(e)$ and $\mathcal{D}(r)$ to represent the textual descriptions of each entity and relation, respectively. For example, in the widely used dataset FB15K-237, we have $\mathcal{D}(/m/0dzlbn) = \text{"Iron Man"}$.

We introduce an LLM \mathcal{M} as the text decoder for KGC. The input sequence \mathcal{S} consists of three components: the instruction prompt \mathcal{I} , the triple prompt \mathcal{X} , and the ontology prompt \mathcal{O} . \mathcal{I} provides manual guidelines for completing the task, such as the instruction "Please help determine whether the triple (h,r,t) is a valid triple." Meanwhile, \mathcal{X} contains textual descriptions of the triple symbols, represented as $\mathcal{X}(h, r, t) = \mathcal{D}(h) \oplus \mathcal{D}(r) \oplus \mathcal{D}(t)$, where (h, r, t) is the triple, and \oplus denotes the concatenation. \mathcal{O} represents the textual form of the ontological information, which is derived by transforming a relevant part of \mathcal{H} into corresponding text.

We focus on triple classification which is a fundamental task of KGC. Following most LLM-based methods that reformulate tasks as text generation problems, we expect our LLM model \mathcal{M} , given a text sequence $\mathcal{S} = \mathcal{I} \oplus \mathcal{X} \oplus \mathcal{O}$ of an input triple, outputs a response of either "True" or "False" for validating this triple. Unlike text classification, KG's complex semantics cannot be captured by a single description. Without KG structural information, LLMs may primarily encapsulate isolated term-level knowledge, which restricts their capacity for comprehensive understanding. Despite their vast general knowledge, LLMs are prone to hallucinations due to insensitivity to fine-grained facts. Thus, effective KGC with LLMs requires vector-based methods to provide structured knowledge, enabling structure-aware reasoning.

4 Methodology

This chapter elucidates how OL-KGC integrates LLMs with vectorized structural and ontological knowledge. Initially, ontological knowledge is extracted in RDF format and transformed into text to serve as part of the input for the LLM. Subsequently, traditional knowledge graph embedding (KGE) algorithms are employed to extract structural information from the KG, which is then provided to the LLM in prefix form to enhance its understanding of the complex structure of the KG. Finally, the LLM integrates the vectorized structural and ontological knowledge, and the completion performance is optimized through fine-tuning using the LoRA method. The framework of OL-KGC is shown in Figure 3.

4.1 Ontological Knowledge Integration

KGs typically store knowledge in symbolic form. For instance, in the FB15K-237 dataset, entities are represented by symbols such as $/m/0dzlbn$, which are not directly suitable for extracting ontological information. To address this, we used an automated script to map these symbols to the Freebase database, generating

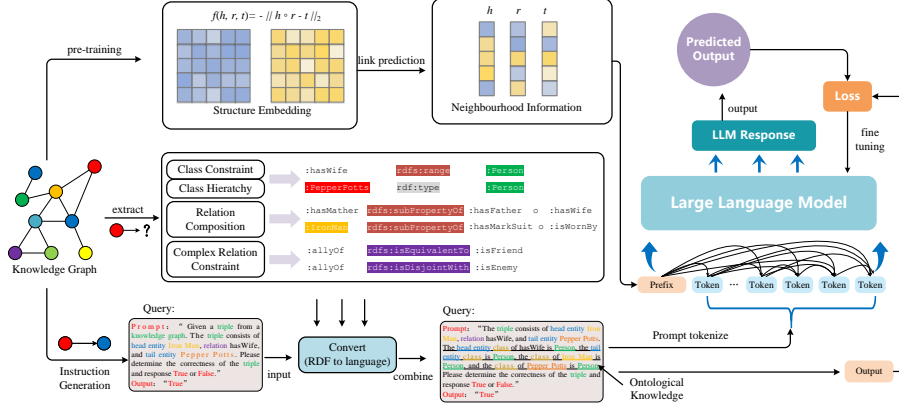


Fig. 3. The overall framework of OL-KGC.

textual descriptions for each entity. For example, the entity `/m/0dzlbbx` is mapped to $\mathcal{D}(/m/0dzlbbx) = \text{"Iron Man"}$.

We further leverage the powerful generative capabilities of LLMs to automatically derive corresponding classes for each entity, generate equivalence, disjointness, and composition rules for each relation, and create class inheritance relationships for each class. Based on the output of the LLM, we designed an algorithm to automatically extract ontological knowledge represented in RDF format, denoted as \mathcal{H} , from its generated results. To ensure the accuracy of the extracted ontological knowledge \mathcal{H} , we introduced a mechanism for manual expert verification.

In addition, we constructed an RDF language mapping dictionary. For a given test triple (h, r, t) , we retrieve relevant RDF triples from \mathcal{H} and transform them into natural language text using predefined templates. For example, the RDF triple $(\text{:IronMan} \text{rdf:type} \text{:Person})$ is mapped to "The class of entity `:IronMan` is `:Person`." This mapping method helps the LLM understand the logical structure of the KG. The process can be represented as follows:

$$\mathcal{O} = \mathcal{B} \left(\sum_{\mathcal{H}_i \in \mathcal{H}} \mathcal{S}(\mathcal{H}_i, (h, r, t)) \right) \quad (1)$$

where $\mathcal{S}(\cdot)$ is an indicator function that determines the relevance between \mathcal{H}_i and the test triple by checking whether there exist entity, relation, or corresponding entity class related to the test triple within \mathcal{H}_i . When \mathcal{H}_i is relevant to the test triple (h, r, t) , $\mathcal{S}(\cdot)$ returns \mathcal{H}_i ; otherwise, it returns nothing. \mathcal{B} maps RDF triples to natural language text via a predefined dictionary. \mathcal{O} is the textual description of ontological knowledge of the KG and will serve as part of the input to the LLM.

4.2 Structural Information Integration

\mathcal{G} contains rich structural information, but its complexity makes it difficult to represent through explicit reasoning processes. To capture the intricate structural information within the KG, we utilize self-supervised embedding pretraining. For each entity $e \in \mathcal{E}$ and each relation $r \in \mathcal{R}$, we learn embeddings $\mathbf{e} \in \mathbb{R}^{d_e}$ and $\mathbf{r} \in \mathbb{R}^{d_r}$, where d_e and d_r represent the dimensions of the entity and relation embeddings, respectively, to represent neighborhood structure information. We evaluated existing KGC models experimentally, and the results show that using a rotational operator most effectively extracts the structural features of the KG. The scoring function is defined as:

$$f(h, r, t) = - \|\mathbf{h} \circ \mathbf{r} - \mathbf{t}\|_2 \quad (2)$$

where \circ denotes the Hadamard product between the head entity embedding and the relation embedding, and $\|\cdot\|_2$ represents the l_2 -norm of the complex vector. By improving the quality of embeddings \mathbf{e} and \mathbf{r} , we are able to extract the complex structural information of the KG. As the score of the positive triples increases, the entity embeddings \mathbf{e} and relation embeddings \mathbf{r} are optimized, thus capturing subgraph structures and relation patterns within the KG. This approach has been demonstrated to be effective in several embedding-based KGC methods [29], which can capture classical structural information such as relation patterns and distributed entity representations.

As shown in Figure 3, for the test triple (h, r, t) , we extract its corresponding vector representation through structural embedding. The dimensionality of this vector is consistent with that of the token vectors in the LLM. To enhance the model’s perception of the structural information in KG, we innovatively concatenate the vectors corresponding to (h, r, t) and transform the structural embedding from the vector space to the LLM token text space via a linear adapter. This transformed representation is then used as a prefix for the input to the LLM.

$$\mathbf{v}_{(h,r,t)} = \mathbf{W}_{\text{linear}}(\mathbf{v}_h \oplus \mathbf{v}_r \oplus \mathbf{v}_t) + \mathbf{b}_{\text{linear}} \quad (3)$$

there \oplus denotes the concatenation operation, $\mathbf{W}_{\text{linear}}$ is the weight matrix of the linear layer, and $\mathbf{b}_{\text{linear}}$ is the bias vector. The resulting vector $\mathbf{v}_{(h,r,t)}$ serves as the prefix for the input to the LLM. This design enables the structural information to directly influence the generation process of each subsequent token, thereby providing richer semantic and structural context for the KGC. During training, the pre-trained structural embeddings are frozen, while the linear adapter is optimized to learn the mapping from structural knowledge to textual representation. This mapping can then be generalized to new triples during the inference stage.

4.3 LLM Fine Tuning

The powerful reasoning capabilities of LLMs provide a more effective tool for integrating vectorized KG structural information and ontological information.

The input to the LLM includes the instruction, the textual interpretation of the triple, and the ontological information, which can be represented as follows:

$$\mathcal{S} = \mathcal{I} \oplus \mathcal{X} \oplus \mathcal{O} \quad (4)$$

there \mathcal{X} denotes the descriptive text of the triple, and \mathcal{O} represents the ontological information that describes the logic within the KG. We concatenate these two types of textual information with the instruction \mathcal{I} to form the input to the LLM. Additionally, the structural embedding is transformed from the vector space to the LLM token text space and serves as a prefix during the LLM’s generation phase, providing structural information of the KG for each token generated in this phase.

$$\mathcal{A}_{\text{token}} = \mathbf{v}_{(h,r,t)} \oplus \text{tokenize}(\mathcal{S}) \quad (5)$$

there $\mathcal{A}_{\text{token}}$ represents the complete token sequence input to the LLM, and the $\text{tokenize}(\cdot)$ function denotes the process of tokenizing the text. Through uni-directional attention decoding, the LLM can perceive the structural information of the KG while generating each token, which is beneficial for prediction.

To train the large model to better predict the triples, thereby enhancing its performance in KGC, we fine-tune the model using LoRA by minimizing the following loss function:

$$\mathcal{L} = \gamma \cdot \mathcal{D}(\mathcal{M}(\mathcal{A}_{\text{token}}), \text{Output}) \quad (6)$$

where γ is a weighting factor used to adjust the intensity of the loss; \mathcal{D} is a discrepancy measurement function that quantifies the difference between the model’s output and the expected result. The model parameters are optimized through backpropagation based on this discrepancy, which helps the LLM perform more accurate KGC.

5 Experiment

We conducted experiments to verify the superiority of OL-KGC. The experiments were guided by answering the following four questions: **Q1:** What are the advantages of this method compared with various traditional KGC methods? **Q2:** What roles do the vectorized structural information and ontological knowledge play respectively? **Q3:** Are the ontological knowledge effectively incorporated into the LLM, and what is the impact of the ontological knowledge?

5.1 Experimental Settings

Datasets. We conducted experiments using three public datasets: UMLS, FB15K-237, and MN18RR. Based on the fundamental datasets, we introduced ontological information in the form of RDF triples to each dataset through the LLM, which encompassed rules such as entity classes, relation composition, equivalence

Table 1. Dataset Statistics.

Dataset	Rel	Ent	Tri	Ont	Ont-cla	Ont-rel	Ont-com
UMLS-O	46	135	6,529	165	106	13	46
FB15K-237O	237	14,541	310,116	18,763	16,073	93	2597
MN18RR-O	11	40,943	94,003	4,710	4,683	9	18

relations, and disjointness relations. This resulted in the creation of the UMLS-O, FB15K-237O, and MN18RR-O datasets. The fundamental datasets are composed in the form of symbols. For the FB15K-237 and WN18RR datasets, we extracted relevant entity description text information from Freebase and WordNet respectively. As for the UMLS dataset, we introduced the Metathesaurus ontology⁴ to obtain relevant information. The enriched datasets will be publicly available⁵. Detailed information is provided in Table 1. The symbols Rel, Ent, Tri, Ont represent the quantities of relations, entities, triples, and ontological information, respectively. The ontological information is further subdivided into three distinct categories: Ont-cla, which pertains to entity classes; Ont-rel, which encompasses relation composition; and Ont-com, which represents constraints on complex relations.

For the data leakage scenario where the LLM may have directly seen or learned from the data used during testing, we have checked that our datasets FB15K-237 and WN18RR, as well as the original KGs they come from, are not used in any training stage of the GLM model which used in our experiments. LLM may have seen text with similar semantics as the structured triples, but such memorization is a key feature of LLM itself. We intentionally selected UMLS — a domain-specific biomedical dataset — to reduce the overlap with the pretraining corpora.

Baseline Methods. In the experimental section, we perform a comparative analysis across four categories of KGC models: traditional KGE models, neuro-symbolic integrated KGC models, PLM-based KGC models, and LLM-based KGC models. For traditional KGE models, we selected four baseline models: TransE [1], ComplEx [30], RotatE [29], and ConvE [8], which represent different approaches to capturing semantic relations in KGs via various scoring functions. The neuro-symbolic models IterE [41] and NeoEA [12] were also included for comparison. For PLM-based KGC models, we chose KG-BERT [38] and KnowC [36], which utilize the semantic power of PLM-based models to integrate graph structure and semantic information. Additionally, we selected high-performance LLM-based models such as KG-LLaMA [39], KoPA [42], MuKDC [19], and KGEitor [6] as baselines.

Evaluation Metrics. Unlike conventional link prediction tasks, LLMs are not inherently suited for predicting the global ranking of specific entities. Consequently, we selected triple classification as the evaluation task for KGC models, which serves as a crucial subtask within the KGC framework. We assessed model

⁴ https://www.nlm.nih.gov/research/umls/knowledge_sources/metathesaurus

⁵ <https://github.com/xiumu-gg/OL-KGC/data>

Table 2. The experimental results for triple classification are evaluated using accuracy (Acc), precision (P), and F1-score (F1). The best results are marked in **bold**, and the second-best results are underlined. The performance metrics for the models are primarily sourced from [42]. For models without reported results in [42], we conducted local experiments to obtain the corresponding values. In cases where certain results could not be reproduced, a '-' symbol is used to indicate missing data.

Model	UMLS-O			FB15K-237O			WN18RR-O		
	Acc	P	F1	Acc	P	F1	Acc	P	F1
TransE	84.46	86.13	83.82	69.85	70.80	68.75	62.55	51.23	52.87
ComplEx	90.76	89.90	90.85	65.68	66.45	64.85	77.43	71.83	69.91
RotatE	91.95	90.14	92.15	68.82	69.43	67.10	78.61	74.82	70.06
ConvE	90.08	88.34	90.26	68.59	69.91	66.88	77.05	71.67	69.50
IterE	80.37	83.29	81.89	65.73	66.62	65.10	71.85	67.81	65.06
NeoEA	85.62	84.62	83.47	66.92	67.29	66.38	72.61	69.42	67.68
KG-BERT	77.25	70.84	80.24	55.83	53.42	67.71	58.80	57.28	64.38
KnowC	80.84	86.97	84.89	59.72	60.12	66.83	68.16	64.80	66.57
KG-LLaMA	85.68	87.77	85.30	74.77	67.40	79.27	<u>80.85</u>	76.83	80.54
KoPA	92.58	<u>90.85</u>	<u>92.70</u>	<u>77.65</u>	70.81	<u>80.81</u>	-	-	-
MuKDC	89.13	90.07	86.85	72.70	<u>75.92</u>	74.63	77.55	77.83	73.82
KGEditor	<u>92.85</u>	86.96	90.56	63.47	68.73	70.03	80.63	<u>81.22</u>	77.81
OL-KGC	93.10	92.32	93.16	80.41	76.26	84.66	81.45	83.50	<u>78.65</u>

performance with the standard suite of metrics—accuracy, precision, recall, and F1-score—defined as follows: Accuracy is defined as the ratio of correctly classified triples (both positive and negative) to the total number of evaluated triples. Precision is the ratio of correctly predicted positive triples to the total number of triples predicted as positive. Recall is the ratio of correctly predicted positive triples to the total number of ground-truth positive triples. F1 is the harmonic mean of precision and recall.

5.2 Link Prediction Results

The experimental results for triple classification are shown in Table 2. Given that the core objective of this study is to evaluate the prediction accuracy and precision of the model, and considering the limitations of page space, we did not present the results of recall in the link prediction experiment section. We believe that precision and the F1 score can more accurately reflect the value of the model. The complete experimental results report has been detailed in the supplementary material for reference by the readers.

OL-KGC achieved state-of-the-art performance on average across the three datasets, with respective performance improvements of 2.238%, 1.361%, and 2.246% on the UMLS-O, FB15K-237O, and WN18RR-O datasets. This indicates that the KGC method combining vectorized structural information and ontological information using LLM can achieve more accurate predictions. It is worth noting that the model reached state-of-the-art levels in terms of accuracy across different datasets, demonstrating its capability to accurately complete

missing information. However, the model exhibited poor performance in terms of the F1 metric on the WN18RR-O dataset. In KGs, certain classes contain a significantly larger number of entities compared to others. The ontological knowledge we designed is unable to effectively constrain the prediction process, which can easily lead to the neglect of minority positive triples, thereby resulting in a relatively lower F1 metric compared to other metrics.

Compared to traditional KGE methods, LLM-based approaches achieve superior results. While PLM-based models possess complex structures, they fail to capture the structural information of triples, leading to poor performance on triple classification tasks. Traditional KGE methods are more suitable for reasoning on small datasets, and their reasoning results are more balanced across four different metrics, without the drawback of a significantly lower F1 score. LLM-based methods, on the other hand, are overall better suited for triple classification tasks, but exhibit significant disparities across different metrics.

On the small UMLS-O dataset, LLM-based methods do not exhibit overwhelming advantages, as smaller parameterized KGE models are capable of capturing sufficient structural and semantic information. However, on larger datasets, LLM-based approaches demonstrate more significant advantages. Specifically, the FB15K-237O dataset contains more relations, and OL-KGC is able to perform more accurate reasoning in KGs with a higher number of relations. In contrast, on the relatively simpler WN18RR-O dataset characterized by fewer relational patterns, OL-KGC lacks the capacity to infer richer relation-based ontological knowledge, leading to performance comparable to other current state-of-the-art LLM-based KGC methods (answering **RQ1**).

Table 3. The results of the ablation study on the FB15K-237O dataset are presented, with the best performance highlighted in **bold**.

	Acc	P	R	F1
w/o ontology	67.81	61.41	69.08	69.40
w/o structure	74.35	70.81	79.38	77.44
w/o structure & ontology	62.18	57.53	66.20	64.83
OL-KGC	80.41	76.26	86.44	84.66

5.3 Ablation Study

Structural Information and Ontological Information. The model achieved state-of-the-art performance by integrating vectorized structural information and ontological knowledge through the LLM. However, the specific contributions of the vectorized structural information and ontological knowledge to the final prediction results remain unknown. To investigate this, we conducted experiments by removing the ontological information from the input $\mathcal{S} = \mathcal{I} \oplus \mathcal{X} \oplus \mathcal{O} \oplus \mathcal{U}$ and eliminating the relevant text in the instructions to assess the impact of removing ontological knowledge (w/o ontology). We also randomized the prefixes

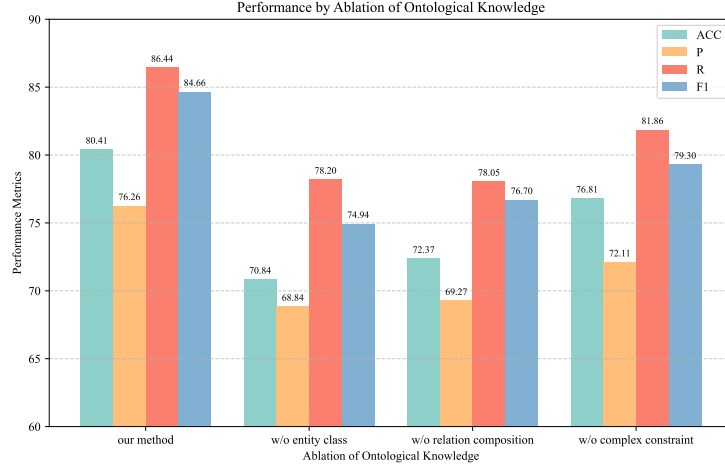


Fig. 4. The results of the ontology rule ablation experiments.

of the neural-perceived structural information to cancel the implicit structural information influence (w/o structure). Additionally, we removed both structural information and ontological knowledge (w/o structure & ontology) to evaluate the performance of a standard LLM on the triple classification task. The results on the FB15K-237O dataset are shown in Table 3.

The experimental results indicate that each module of OL-KGC plays a role in the model’s final performance. The impact of structural information on the final outcome is relatively minor, ranging from 7.05% to 8.50%. The performance of a standard LLM on the triple classification task is significantly inferior, with a gap of 22.77% to 24.48% compared to OL-KGC. The influence of ontological knowledge on the model lies between 15.77% and 20.08%. Given that LLMs are inherently based on a black-box architectural design, the implicit knowledge provided by vectorized structural information serves as a supplement to LLMs. Ontological knowledge, on the other hand, offers logical connections for LLM reasoning. Therefore, it is reasonable that ontological knowledge has a more substantial impact on the results.

Concrete Ontological Information. To investigate the impact of different classes of ontological information, we selectively removed distinct rules and assessed the effects of entity classes (w/o entity class), relation compositions (w/o relation composition), and complex constraints (w/o complex constraint) on the model. Specifically, for entity classes and class inheritance relations, we eliminated the RDF triples corresponding to these rules and the RDF triples that restrict the class of triples. For relation compositions, we removed all RDF rules related to relation compositions. Similarly, for complex relations, we deleted rules concerning equivalence, disjointness, and other complex relations. The results on the FB15K-237O dataset are illustrated in Figure 4.

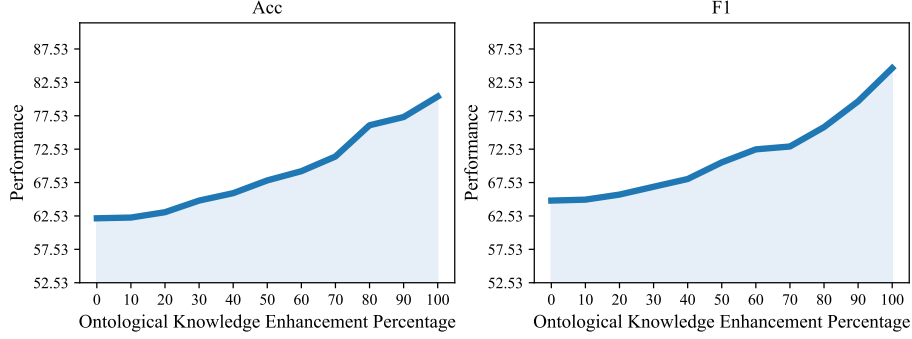


Fig. 5. The results of the ontological knowledge analysis experiment.

The experimental results indicate that OL-KGC mitigates hallucinations by injecting logical constraints derived from the underlying ontology, thereby enhancing inter-run consistency and improving inferential accuracy. Specifically, the impact of entity class rules is relatively substantial, ranging from 9.53% to 12.01%. In contrast, the influence of complex relations on the model is relatively minor, falling between 4.60% and 6.30%. We attribute these differences to the fact that specialized complex category relations are not extensively present in the KG data; they primarily serve as supplementary information to aid the LLM in reasoning. Conversely, entity classes can impose constraints on every triple (answering **RQ2**).

5.4 Ontological Knowledge Analysis Experiment

We focus specifically on the impact of ontological knowledge on model performance. By gradually increasing the application of ontology rules, we compare the model’s ACC and F1 under constrained and unconstrained conditions. This comparison visually demonstrates the positive impact of explicit rules on model performance, thereby validating the model’s interpretability in KGC. The results on the FB15K-237O dataset are shown in Figure 5.

The dark-colored lines indicate the performance of the model after incorporating the corresponding percentage of ontological knowledge, while the shaded areas represent the performance gains attributable to the use of such knowledge—that is, correct reasoning outcomes derived from ontological analysis. Across all evaluation metrics, model performance consistently improves as the amount of ontological knowledge increases. This trend suggests that the incremental incorporation of ontological knowledge provides the model with progressively richer semantic constraints.

Meanwhile, our approach is readily applicable to real-world KGs that embed rich ontological knowledge, with financial KGs serving as a paradigmatic example. Within these graphs, hierarchical concept taxonomies such as `:Enterprise` \sqsupseteq `:SmallEnterprise` \sqsupseteq `:HighRiskSmallEnterprise` enable OL-KGC to (i) in-

fer class inheritance and (ii) verify constraint satisfaction, thereby safeguarding class-level consistency and enhancing the accuracy of downstream risk assessments (answering **RQ3**).

5.5 Case Analysis Experiment

To track the role of ontological knowledge, we conducted a case study, showcasing several correct and incorrect predictions from our method, with their explanation. In our experiments on the FB15K-237 dataset, we observed that the triple `(:LosAngeles, :locatedIn, :Country)` was incorrectly predicted as True by the LLM when no ontological knowledge was provided. This misjudgment arises from strong semantic associations between the entities, despite violating the range constraint of `:locatedIn` relation, which expects the tail entity to belong to `:Country` class (for example, entity `:America` satisfies this constraint), whereas the entity `:Country` itself does not belong to the `:Country` class. When ontological constraints — specifically domain and range information — were injected, OL-KGC correctly predicted this triple as False. Conversely, the valid triple `(:LosAngeles, :locatedIn, :Country)` was consistently predicted as True due to class consistency.

6 Conclusion

This paper introduces OL-KGC, a KGC method that leverages LLMs. By integrating vectorized structural information and ontological knowledge, the performance and reasoning capability of the KGC are effectively enhanced. We introduced various types of ontological knowledge into existing KGs and utilized LLMs to incorporate symbolic ontological knowledge into reasoning. Experimental results demonstrate that OL-KGC achieves state-of-the-art performance in the field of KGC.

Supplemental Material Statement: We have published the algorithms for introducing ontological knowledge to existing KGs using LLM on GitHub⁶. The datasets FB15K-237O, WN18RR-O, and UMLS-O, which were constructed on FB15K-237, WN18RR, and UMLS, are also open-sourced on GitHub⁷. The algorithmic implementations and complete experimental parameters — including per-component computational costs and detailed hardware configurations — are publicly available on GitHub⁸.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (62472311), the Key Research and Development Program of Ningxia Hui Autonomous Region (2023BEG02067) and the EPSRC Project OntoEm (EP/Y017706/1).

⁶ <https://github.com/xiumu-gg/OL-KGC/dataset-process>

⁷ <https://github.com/xiumu-gg/OL-KGC/data>

⁸ <https://github.com/xiumu-gg/OL-KGC>

References

1. Bordes, A., Usunier, N., Garcia-Durán, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: Proceedings of the 26th International Conference on Neural Information Processing Systems. pp. 2787–2795. NeurIPS’13 (2013)
2. Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J.D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al.: Language models are few-shot learners. vol. 33, pp. 1877–1901 (2020)
3. Chen, J., Zhang, K., Gan, A., Tong, S., Shen, S., Liu, Q.: Enhancing knowledge graph completion with entity neighborhood and relation context. arXiv preprint arXiv:2503.23205 (2025)
4. Chen, S., Liu, X., Gao, J., Jiao, J., Zhang, R., Ji, Y.: Hitter: Hierarchical transformers for knowledge graph embeddings. In: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. pp. 10395–10407. EMNLP’21 (2021)
5. Chen, Y., Li, S., Fang, H., Li, Y., Hu, J.: A knowledge graph completion model combining graph convolutional networks with self-attention mechanisms. In: 2024 IEEE 5th International Conference on Pattern Recognition and Machine Learning (PRML). pp. 199–203 (2024). <https://doi.org/10.1109/PRML62565.2024.10779724>
6. Cheng, S., Zhang, N., Tian, B., Chen, X., Liu, Q., Chen, H.: Editing language model-based knowledge graph embeddings. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 38, pp. 17835–17843 (2024)
7. DeLong, L.N., Mir, R.F., Fleuriot, J.D.: Neurosymbolic ai for reasoning over knowledge graphs: A survey. IEEE Transactions on Neural Networks and Learning Systems (2024)
8. Dettmers, T., Minervini, P., Stenetorp, P., Riedel, S.: Convolutional 2d knowledge graph embeddings. In: Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence. pp. 1811–1818. AAAI’18 (2018)
9. Fan, M., Zhou, Q., Chang, E., Zheng, F.: Transition-based knowledge graph embedding with relational mapping properties. In: Proceedings of the 28th Pacific Asia Conference on Language, Information and Computing. pp. 328–337 (2014)
10. Feng, T., Wu, Y., Li, L.: Research on knowledge graph completion based upon knowledge graph embedding. In: 2024 9th International Conference on Computer and Communication Systems (ICCCS). pp. 1335–1342 (2024). <https://doi.org/10.1109/ICCCS61882.2024.10603073>
11. Gu, Y., Kase, S., Vanni, M., Sadler, B., Liang, P., Yan, X., Su, Y.: Beyond iid: three levels of generalization for question answering on knowledge bases. In: Proceedings of the Web Conference 2021. pp. 3477–3488 (2021)
12. Guo, L., Zhang, Q., Sun, Z., Chen, M., Hu, W., Chen, H.: Understanding and improving knowledge graph embedding for entity alignment. In: International Conference on Machine Learning. pp. 8145–8156. PMLR (2022)
13. Guo, S., Wang, Q., Wang, L., Wang, B., Guo, L.: Jointly embedding knowledge graphs and logical rules. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics (2016)
14. Hogan, A., Blomqvist, E., Cochez, M., D’amato, C., Melo, G.D., Gutierrez, C., Kirrane, S., Gayo, J.E.L., Navigli, R., Neumaier, S., Ngomo, A.C.N., Polleres, A., Rashid, S.M., Rula, A., Schmelzeisen, L., Sequeda, J., Staab, S., Zimmermann, A.:

- Knowledge graphs. *ACM Comput. Surv.* **54**(4) (Jul 2021). <https://doi.org/10.1145/3447772>, <https://doi.org/10.1145/3447772>
15. Ji, G., He, S., Xu, L., Liu, K., Zhao, J.: Knowledge graph embedding via dynamic mapping matrix. In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*. pp. 687–696. *ACL’15* (2015)
 16. Ji, S., Liu, L., Xi, J., Zhang, X., Li, X.: Klr-kgc: Knowledge-guided llm reasoning for knowledge graph completion. *Electronics* **13**(24) (2024). <https://doi.org/10.3390/electronics13245037>, <https://www.mdpi.com/2079-9292/13/24/5037>
 17. Jiang, D., Wang, R., Xue, L., Yang, J.: Multisource hierarchical neural network for knowledge graph embedding. *Expert Systems with Applications* **237**, 121446 (2024)
 18. Jiang, X., Wang, Q., Wang, B.: Adaptive convolution for multi-relational learning. In: *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)*. pp. 978–987. *NAACL’20* (2019)
 19. Li, Q., Chen, Z., Ji, C., Jiang, S., Li, J.: Llm-based multi-level knowledge generation for few-shot knowledge graph completion. In: *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*. vol. 271494703 (2024)
 20. Li, Z., Liu, X., Wang, X., Liu, P., Shen, Y.: Transo: a knowledge-driven representation learning method with ontology information constraints. *World Wide Web* **26**(1), 297–319 (2023)
 21. Li, Z., Liu, H., Zhang, Z., Liu, T., Xiong, N.N.: Learning knowledge graph embedding with heterogeneous relation attention networks. *IEEE Transactions on Neural Networks and Learning Systems* **33**(8), 3961–3973 (2021)
 22. Nayyeri, M., Vahdati, S., Aykul, C., Lehmann, J.: 5* knowledge graph embeddings with projective transformations. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. vol. 35, pp. 9064–9072 (2021)
 23. Nguyen, D.Q., Nguyen, T.D., Nguyen, D.Q., Phung, D.: A novel embedding model for knowledge base completion based on convolutional neural network. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. pp. 327–333. *NAACL’18* (2018)
 24. Pan, S., Luo, L., Wang, Y., Chen, C., Wang, J., Wu, X.: Unifying large language models and knowledge graphs: A roadmap. *IEEE* (2024)
 25. Pan, Y., Liu, J., Zhao, T., Zhang, L., Lin, Y., Dong, J.S.: A symbolic rule integration framework with logic transformer for inductive relation prediction. In: *Proceedings of the ACM Web Conference 2024*. p. 2181–2192. *WWW ’24*, Association for Computing Machinery, New York, NY, USA (2024). <https://doi.org/10.1145/3589334.3645594>, <https://doi.org/10.1145/3589334.3645594>
 26. Sanmartin, D.: Kg-rag: Bridging the gap between knowledge and creativity. *arXiv preprint arXiv:2405.12035* (2024)
 27. Saxena, A., Kochsiek, A., Gemulla, R.: Sequence-to-sequence knowledge graph completion and question answering. In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*. pp. 2814–2828. *ACL’22* (2022)
 28. Sheikh, N., Qin, X., Reinwald, B., Miksovic, C., Gschwind, T., Scotton, P.: Knowledge graph embedding using graph convolutional networks with relation-aware attention. *arXiv preprint arXiv:2102.07200* (2021)
 29. Sun, Z., Deng, Z.H., Nie, J.Y., Tang, J.: Rotate: Knowledge graph embedding by relational rotation in complex space. In: *Proceedings of the Seventh International Conference on Learning Representations*. pp. 328–337. *ICLR’19* (2019)

30. Trouillon, T., Welbl, J., Riedel, S., Gaussier, É., Bouchard, G.: Complex embeddings for simple link prediction. In: Proceedings of the 33rd International Conference on International Conference on Machine Learning. pp. 2071–2080. ICML’16 (2016)
31. Wang, D., Zuo, Y., Li, F., Wu, J.: Llms as zero-shot graph learners: Alignment of gnn representations with llm token embeddings. arXiv preprint arXiv:2408.14512 (2024)
32. Wang, Q., Mao, Z., Wang, B., Guo, L.: Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering* **29**(12), 2724–2743 (2017)
33. Wang, X., Huang, T., Wang, D., Yuan, Y., Liu, Z., He, X., Chua, T.S.: Learning intents behind interactions with knowledge graph for recommendation. In: Proceedings of the Web Conference 2021. pp. 878–887 (2021)
34. Wang, Z., Zhang, J., Feng, J., Chen, Z.: Knowledge graph embedding by translating on hyperplanes. In: Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence. pp. 1112–1119. AAAI’14 (2014)
35. Xu, J., Fei, H., Pan, L., Liu, Q., Lee, M.L., Hsu, W.: Faithful logical reasoning via symbolic chain-of-thought (2024), <https://arxiv.org/abs/2405.18357>
36. Yang, G., Liu, Y., Zhang, L., Zhang, L., Xie, H., Mao, Z.: Knowledge context modeling with pre-trained language models for contrastive knowledge graph completion. In: Findings of the Association for Computational Linguistics ACL 2024. pp. 8619–8630 (2024)
37. Yang, L., Chen, H., Li, Z., Ding, X., Wu, X.: Chatgpt is not enough: Enhancing large language models with knowledge graphs for fact-aware language modeling. pp. arXiv–2306 (2023)
38. Yao, L., Mao, C., Luo, Y.: Kg-bert: Bert for knowledge graph completion. arXiv e-prints pp. arXiv–1909 (2019)
39. Yao, L., Peng, J., Mao, C., Luo, Y.: Exploring large language models for knowledge graph completion. arXiv preprint arXiv:2308.13916 (2023)
40. Zhang, W., Chen, J., Li, J., Xu, Z., Pan, J.Z., Chen, H.: Knowledge graph reasoning with logics and embeddings: Survey and perspective. In: 2024 IEEE International Conference on Knowledge Graph (ICKG). pp. 492–499 (2024). <https://doi.org/10.1109/ICKG63256.2024.00069>
41. Zhang, W., Paudel, B., Wang, L., Chen, J., Zhu, H., Zhang, W., Bernstein, A., Chen, H.: Iteratively learning embeddings and rules for knowledge graph reasoning. In: The World Wide Web Conference. p. 2366–2377. WWW ’19, Association for Computing Machinery, New York, NY, USA (2019). <https://doi.org/10.1145/3308558.3313612>, <https://doi.org/10.1145/3308558.3313612>
42. Zhang, Y., Chen, Z., Guo, L., Xu, Y., Zhang, W., Chen, H.: Making large language models perform better in knowledge graph completion. In: Proceedings of the 32nd ACM International Conference on Multimedia. pp. 233–242 (2024)
43. Zhao, W.X., Zhou, K., Li, J., Tang, T., Wang, X., Hou, Y., Min, Y., Zhang, B., Zhang, J., Dong, Z., Du, Y., Yang, C., Chen, Y., Chen, Z., Jiang, J., Ren, R., Li, Y., Tang, X., Liu, Z., Liu, P., Nie, J.Y., Wen, J.R.: A Survey of Large Language Models. p. arXiv:2303.18223 (Mar 2023). <https://doi.org/10.48550/arXiv.2303.18223>