Expert Systems With Applications 301 (2026) 130496

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

ReaCo-KGC: a reasoning-enhanced and interaction-corrective framework based on large language models for knowledge graph completion

Tingting Jiang [a,b], Suqing Wu [a], Shuai Yang [a,b], Xiaohui Yuan [a,c], Lichuan Gu [a,*], Xindong Wu [b,*]

[a] School of Information and Artificial Intelligence, Anhui Agricultural University, Hefei, Anhui, China
[b] Key Laboratory of Knowledge Engineering with Big Data, Hefei University of Technology, Hefei, Anhui, China
[c] Department of Computer Science and Engineering, University of North Texas, TX, 76203, USA

ARTICLE INFO

Keywords:
Knowledge graph completion
Large language model
Chain-of-thought
Multi-agent

ABSTRACT

Knowledge Graph Completion (KGC) aims to infer missing entities for incomplete triples. Traditional embedding-based methods rely solely on the graph structural information, making limited use of textual semantics. Although emerging text-based methods, i.e., utilizing large language models (LLMs) to learn semantic information, can solve the aforementioned problems to some extent. They still face limitations: 1) Due to the extensive scale of knowledge graphs, prompt information is often lengthy, making it difficult for LLMs to focus on key information when processing long texts; 2) When dealing with semantically similar entities, LLMs often struggle to capture subtle differences between them, leading to insufficient discriminative capability and resulting in confusion. To address these challenges, we propose a reasoning-enhanced and interaction-corrective framework based on large language models for knowledge graph completion (ReaCo-KGC). First, a two-stage prompting mechanism is designed to enable LLMs to extract key information and reasoning processes from in-context information, i.e., adjacent triples, helping LLMs focus on the core content. In addition, a multi-agent re-ranking component that applies a turn-taking summarization strategy is proposed to refine the results and resolve confusion caused by semantic similarity. Moreover, we reduce the candidate set size effectively by using a lightweight model to eliminate irrelevant entities. Experiments on benchmark datasets FB15k-237 and WN18RR demonstrate the superior performance of the proposed framework, validating its effectiveness.

1. Introduction

Knowledge Graphs (KGs) structure information as triples in the form $(head\ entity, relation, tail\ entity)$, denoted as (e_h, r, e_t). However, due to limitations in knowledge acquisition, constructed knowledge graphs often suffer from incompleteness and missing facts. This significantly hinders their effectiveness in downstream applications, such as question answering (Huang et al., 2019, 2021), recommendation systems (Chen & Xie, 2023; Zhang et al., 2024), and semantic search (Li et al., 2025; Perevalov et al., 2024). To address this issue, Knowledge Graph Completion (KGC) has emerged as a critical task, aiming to predict missing triples and enhance the completeness and utility of KGs.

Existing KGC methods can be broadly categorized into two types: embed-ding-based methods (Bordes et al., 2013; Sun et al., 2019; Yang et al., 2015) and text-based methods (Kim et al., 2020; Wang et al., 2021; Yao et al., 2019). Embedding-based methods rely on the structural information of KG and perform reasoning by implicitly learning relational patterns such as transitivity and symmetry (Wu et al., 2023). These methods often overlook semantic information in text and struggle to represent long-tail entities, leading to factually inconsistent predictions and poor performance. Text-based methods enhance the performance of KGC by leveraging the textual descriptions of entities and relations. Traditional language models such as BERT and RoBERTa encode textual semantics to establish entity associations, capturing richer contextual features. Recently, Large Language Models (LLMs) like GPT and DeepSeek have opened new possibilities for KGC due to their powerful generative and contextual learning capabilities. The mainstream approaches for KGC based on LLMs involve contextual learning, prompt engineering, and model fine-tuning (Li et al., 2024c; Liu et al., 2024; Wei et al., 2023). These approaches achieved considerable results, but still face notable limitations: 1) Due to the vast scale of knowledge graphs, the required prompt information increases accordingly. Having LLMs

* Corresponding author.
E-mail addresses: jiangtt@ahau.edu.cn (T. Jiang), wusuqing@stu.ahau.edu.cn (S. Wu), yangs@ahau.edu.cn (S. Yang), xiaohui.yuan@unt.edu (X. Yuan), glc@ahau.edu.cn (L. Gu), xwu@hfut.edu.cn (X. Wu).

https://doi.org/10.1016/j.eswa.2025.130496
Received 15 July 2025; Received in revised form 13 October 2025; Accepted 16 November 2025
Available online 19 November 2025

process large volumes of text at once can easily lead to a loss of focus, resulting in information dilution. This ultimately affects the accuracy, coherence, and relevance of the output; 2) When dealing with semantically similar entities, LLMs often fail to recognize subtle semantic differences, resulting in indistinct entity representations. This reduces the discriminative accuracy of LLMs and leads to entity confusion. For example, it is difficult to distinguish the differences among Crewe Alexandra F.C., Middlesbrough F.C., and Inverness Caledonian Thistle F.C. based solely on surface semantics; 3) Fine-tuning large language models often requires substantial computational resources and time, involving significant hardware costs and placing higher demands on training efficiency and scalability.

Considering the existing challenges, we propose a reasoning-enhanced and interaction-corrective framework based on large language models for knowledge graph completion (ReaCo-KGC). This framework achieves knowledge graph completion through the coordinated functioning of three core modules. The Knowledge Guide module mitigates the long-tail entity problem by providing the LLM with structured contextual information, including adjacent triples, relevant triples, and entity long text. The module adopts a two-stage prompting strategy. In the first stage, the LLM extracts key information and reasoning processes from the context. In the second stage, the extracted core content is combined with the original question to derive the final answer. This design improves the reasoning ability of LLMs in complex tasks and helps them focus on the key content. The Collaborative Optimizer module introduces a multi-agent debate mechanism, utilizing a novel turn-taking summarization strategy to collaboratively refine candidate results. This effectively resolves confusion caused by semantic similarity. The Candidate Filter module employs a lightweight model to efficiently filter the KG's entity set, reducing the candidate set from tens of thousands to just dozens, thereby significantly lowering the complexity of subsequent processing.

The ReaCo-KGC framework demonstrates strong plug-and-play compatibility and can seamlessly integrate with any base KGC model and mainstream LLMs. Compared to existing embedding-based methods, our framework leverages retrieved contextual information and the LLM's knowledge base to enhance semantic richness. Compared to existing LLM-based methods, we employ a two-stage prompting mechanism to guide the LLM in extracting key information and reasoning processes from redundant context, significantly reducing the problem of attention dispersion. Furthermore, the multi-agent collaborative optimization mechanism effectively reduces confusion among similar entities for LLMs, markedly improving the accuracy and stability of the results. Our main contributions can be summarized as follows:

- A two-stage prompting strategy that explicitly separates reasoning generation from answer extraction is proposed, guiding the LLM to actively extract key information and reasoning processes from the context, helping it focus on the core content and avoiding attention dilution caused by long prompts.
- A multi-agent re-ranking mechanism with turn-taking summarization is proposed, representing a novel approach to applying multi-agent systems to knowledge graph completion. This strategy effectively distinguishes semantically similar entities and significantly improves discrimination accuracy.
- Experiments on FB15k-237 and WN18RR show FGC-KGC achieves strong performance, improving Hits@1 by 15.1 % and 11.1 % respectively over filtering models, demonstrating its effectiveness for the task of KGC.

2. Related work

This section reviews prior work on knowledge graph completion, large language models for KGC, and multi-agent systems, which provides the background and motivation for our proposed framework.

2.1. Knowledge graph completion

Current KGC approaches can be broadly categorized into two main types: embedding-based methods and text-based methods. Embedding-based knowledge graph completion methods learn low-dimensional vector representations of entities and relations by leveraging geometric or mathematical constraints to capture semantic and structural information in knowledge graphs. Classic methods include: TransE (Bordes et al., 2013) interprets relations as "translations" in the vector space. For a valid triple (h, r, t), the embedding vectors should satisfy $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$. DistMult (Yang et al., 2015) models symmetric relations via matrix factorization, making it suitable for scenarios requiring symmetry. RotatE (Sun et al., 2019) employs rotational embeddings in complex space, modeling relations by rotating the head entity's embedding. Text-based knowledge graph completion methods utilize natural language processing techniques to enrich semantic information through textual descriptions and integrate pre-trained language models to improve performance. Examples include: KG-BERT (Yao et al., 2019) applies the BERT model to process triples and their textual descriptions, evaluating triple plausibility via semantic information. StAR (Wang et al., 2021), a structure-aware relational reasoning model, enhances reasoning accuracy and robustness by combining structural information with textual descriptions via contrastive learning. MTL-KGC (Kim et al., 2020) introduces a multi-task learning approach for knowledge graph completion, combining link prediction, relation prediction and relevance ranking to better learn relational patterns and handle lexical similarity interference. Currently, embedding-based methods have obvious advantages in efficiency and scalability, making them suitable for fast completion of large-scale knowledge graphs. However, they have limitations in expressing complex semantics and handling edge entities. In contrast, text-based methods fully leverage rich linguistic semantic information, improving the accuracy and robustness of completion, but they often face high computational costs and challenges in semantic matching.

2.2. LLMs for KGC

Compared to traditional text models that establish entity associations by encoding textual semantics, the use of LLMs to assist KGC has emerged as a promising research direction in text-based methods. For LLMs, a typical approach involves ICL (Brown et al., 2020), where explicit instructions and examples are provided to guide the model's behavior. This method has demonstrated strong performance across various language understanding and generation tasks (Rae et al., 2021; Wei et al., 2022). Leveraging their powerful in-context learning and semantic comprehension capabilities, LLMs offer a novel paradigm for knowledge graph completion (Li et al., 2024a; Zhu et al., 2024). For instance, the KC-GenRe framework (Wang et al., 2024a) employs knowledge constraints to guide LLMs in semantically re-ranking candidate entities, effectively integrating generative capabilities with structured knowledge to accomplish completion tasks. Similarly, KICGPT (Wei et al., 2023) utilizes the few-shot and zero-shot learning abilities of LLMs to achieve outstanding KGC performance without fine-tuning. However, due to the large scale of knowledge graphs, the amount of prompt information required increases accordingly, and having LLMs process a large amount of text at once can easily cause them to lose focus on the key points. The Chain-of-Thought (CoT) technique effectively alleviates this issue by introducing explicit reasoning steps. The CoT method proposed in Wei et al. (2022b) significantly improves performance on complex tasks by generating reasoning chains in the form of "hypothesis-derivation-conclusion." Subsequent improvements, such as Zero-shot CoT (Kojima et al., 2022), eliminate the need for task-specific examples, while Auto-CoT (Zhang et al., 2022) optimizes the reasoning process through automated question clustering and example generation. By incorporating explicit reasoning steps, CoT not only enhances the stability of reasoning but also helps LLMs concentrate on the key content, making the reasoning process more logical and coherent.

2.3. Multi-agent systems

In recent years, the development of communicative agents has attracted widespread attention. These agents are typically powered by LLMs and aim to facilitate more efficient and productive interactions and collaborations. Different agents can autonomously communicate and negotiate to collectively solve more complex tasks. The multi-agent communication paradigm shows potential synergy in KGC tasks: current KGC research primarily focuses on using a single LLM to generate results through one-time forward reasoning. When handling entities with highly similar semantics, it is often difficult to capture subtle differences, leading to insufficient discrimination ability and resulting in confusion. In contrast, the dynamic communication mechanism among multiple agents can effectively capture fine-grained semantic differences between similar entities through multi-perspective collaborative verification and distributed consensus decision-making, thereby improving discrimination accuracy and the robustness of completion results. Camel (Li et al., 2023) proposed a role-playing cooperative agent framework that enables agents to autonomously collaborate on solving complex tasks. MAD (Liang et al., 2024) employed multi-agent debate frameworks in other scenarios (e.g., translation and arithmetic problems) and achieved improved results. SPP (Wang et al., 2024b) proposed an alternative approach called self-collaboration, where a single LLM prompted with multiple personality descriptions enables inter-agent communication. RoCo (Mandi et al., 2024) introduced a novel framework aimed at enhancing multi-robot collaboration by leveraging multiple LLMs to improve coordination and strategic planning among robots. However, there remains a lack of mature research applying multi-agent systems to knowledge graph completion tasks. This paper proposes a multi-agent system where internal agents are also based on LLMs but are further endowed with distinct role descriptions. Through a turn-taking summarization strategy for interaction, our system demonstrates preliminary potential in KGC tasks.

3. Methodology

This section presents the design of our proposed ReaCo-KGC framework, including its three main components: Candidate Filter, Knowledge Guidance, and Collaborative Optimizer.

3.1. Concept and task definition

A knowledge graph can be represented as a set of triples $G = \{(h, r, t)\}$, where E and R denote the set of entities and the set of relations in the graph G, respectively. Here, $h \in E$ represents the head entity, $t \in E$ represents the tail entity, and $r \in R$ represents the relation between them. Given an incomplete triple $(h, r, ?)$ or $(?, r, t)$ as a query, the goal of knowledge graph completion is to predict the missing entity (denoted by ?). KGC models typically need to score all possible entities as candidates for the missing entity and rank all entities in descending order based on their scores, so as to select the most likely missing entity.

3.2. Overview

The ReaCo-KGC framework is illustrated in Fig. 1. This framework consists of three core modules: Candidate Filter, Knowledge Guidance, and Collaborative Optimizer. Taking the tail entity prediction task as an example, the framework operates as follows: Given an input query triple $(e_h, r, ?)$, where e_h denotes the head entity, r represents the relation, and ? indicates the tail entity to be predicted. First, the lightweight model in the Candidate Filter module computes confidence scores for all candidate triples (e_h, r, e_i) corresponding to each entity $e_i \in E$ in the KG. Based on these scores, it generates a preliminary ordered candidate entity set: $CandidateEntities = [e_1, e_2, \ldots, e_{n-1}, e_n]$. Next, the Knowledge Guidance module retrieves structured prompts, including relevant triples, adjacent triples, and entity long text descriptions. These structured prompts are converted into natural language through semantic mediator in the prompt engineering component to facilitate LLM comprehension. Leveraging ICL capabilities, the LLM extracts rich semantic information from the prompts while employing CoT reasoning to output both a ranking rationale and potential new entities that may have been overlooked by the Candidate Filter. The LLM then performs a combined re-ranking of the candidate entities and new entities, producing an optimized ordered set: $Reordered\ Candidate = [e'_1, e'_2, \ldots, e'_{n-1}, e'_n]$. Finally, the Collaborative Optimizer employs a turn-taking summarization strategy to orchestrate multi-agent discussions. Through iterative deliberation and positional refinement, it further optimizes the reordered candidate, ultimately outputting the final ranked list: $Optimized\ Candidate = [e''_1, e''_2, \ldots, e''_{n-1}, e''_m]$.

3.3. Candidate filter

In KGC tasks, common evaluation methods rely on ranking mechanisms, requiring the model to assess the plausibility of each entity in the knowledge graph as a potential substitute for the missing entity in the query triple. However, due to the extremely large number of entities in the knowledge graph, directly using LLMs to score and rank each entity is both computationally expensive and practically infeasible.

Inspired by Li et al. (2024b), Wei et al. (2023), we employ a base KGC model to initialize the scoring and ranking of entities in the knowledge graph. Formally, we represent the ranked entity list scores as $Candidate\ Score$ as follows:

$$Candidate\ Score = [S_1, S_2, \ldots, S_k],$$
$$where\ S_i = \max f_r(e_h, e_i) \text{ or } f_r(e_i, e_t), \quad i \in \{1, \ldots, k\} \tag{1}$$

Where f_r represents the scoring function of the KGC model under relation, e_h and e_t denote the head and tail entities in a triple, e_i is a candidate entity from the knowledge graph.

3.4. Knowledge guidance

This section presents Knowledge Guidance, an optimized strategy for KGC tasks that leverages large language models' ICL and CoT capabilities. The framework comprises four key components: Relevant and Adjance Triples Retrieval, Entity Long Text Retrieval, Semantic Mediator, and Preliminary Reasoning. Specifically, the module retrieves relevant and adjacent triples to provide contextualized relational information, while long-text entity descriptions are incorporated to supply more detailed semantic cues. These retrieved contexts and descriptions are then transformed into natural language prompts and encoded as inputs to the LLM. Finally, the model is guided to analyze the contextual information and long-text descriptions of different similar entities through Preliminary Reasoning, capturing subtle semantic differences among them. This integrated methodology demonstrably enhances the performance of LLMs in KGC applications.

3.4.1. Relevant and adjacent triples retrieval

In KGs, entity attributes are represented in the form of structured triples. Entities sharing the same relation in triples typically possess similar attributes, while triples containing the same entity can provide additional relevant information. For example, in the triples (Barack Obama, born_in, Honolulu) and (Barack Obama, spouse, Michelle Obama), although the relations differ, both center on the entity "Barack Obama". The first triple provides birthplace information, and the second offers family relationship details. For each query $(e_h, r, ?)$, we construct two types of triples from the KG: relevant triples and adjacent triples. The definition of $Relevant\ Triples$ is as follows:

$$Relevant\ Triples = \left\{(e'_h, r, e'_t) \in D_{train} \cup D_{valid} \mid e'_h, e'_t \in E\right\} \tag{2}$$

Where D_{train} and D_{valid} represent the sets of knowledge graph triples in the training set and validation set, respectively, while E denotes the set

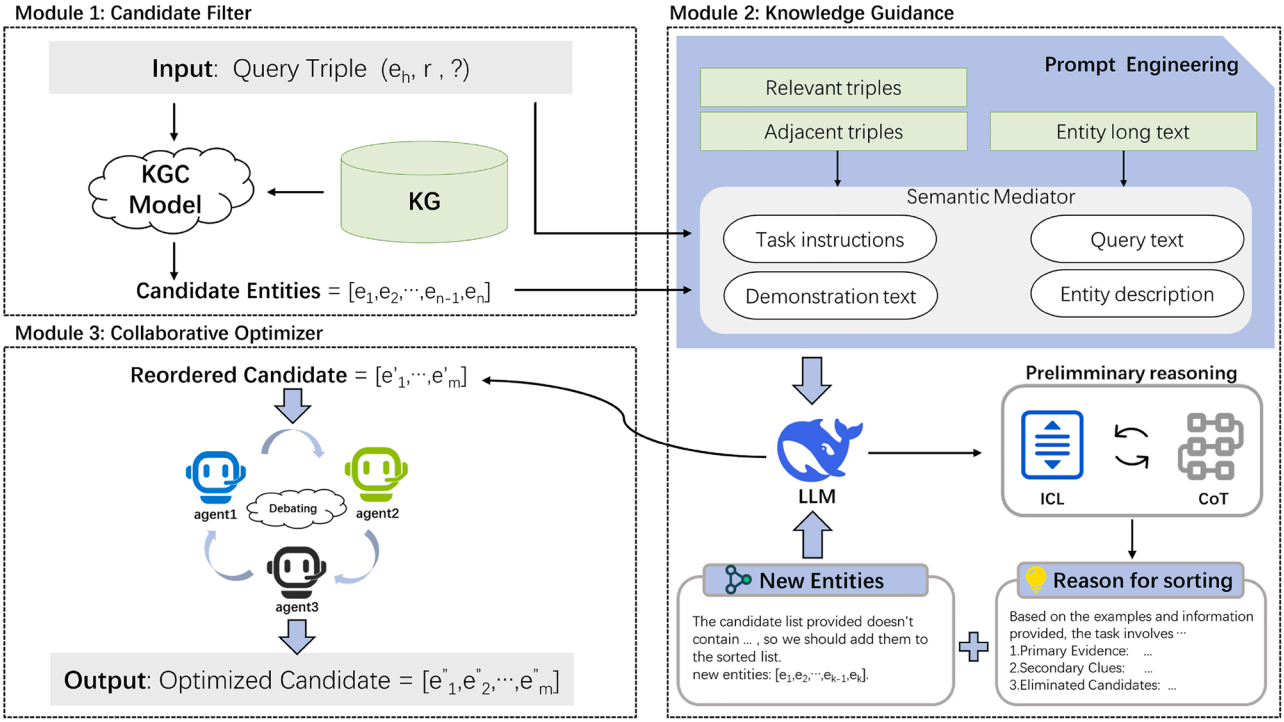


Fig. 1. An illustration of the ReaCo-KGC framework.

of entities in the knowledge graph. By providing triples that share the same relation r as the query, we enable the LLM to better comprehend the semantic meaning of the query. The definition of $Adjacent\ Triples$ is as follows:

$$Adjacent\ Triples = \{(e'_h, r, e'_t) \in D_{train} \cup D_{valid} \mid e'_h, e'_t \in E\} \cup \{(e'_h, r', e_h) \in D_{train} \cup D_{valid} \mid r' \in R, e'_h \in E\} \quad (3)$$

Where R represents the set of relations in the knowledge graph. By providing additional triples associated with the query's head entity e_h, we supply the LLM with richer contextual information.

3.4.2. Entity long text retrieval

In mainstream knowledge graphs, entities are typically represented as numerical or textual IDs. For instance, the entity "Jeremy Irons" is uniquely identified by the ID "/m/016ywr". Such structured formats pose challenges for the effective processing by LLMs. To fully leverage the semantic understanding capabilities of LLMs, this study adopts a text-based KGC entity representation approach by establishing a one-to-one correspondence between entity textual IDs and their long-text descriptions. The relevant descriptions can be retrieved based on the unique textual IDs of entities. This method transforms abstract IDs in knowledge graphs into semantically rich textual information, enhancing the model's comprehension of entity meanings while providing additional contextual information, thereby improving the model's performance in entity prediction tasks. For entities in the FB15K-237 and WN18RR datasets, the study employs the same textual IDs and long text descriptions as used in KGBERT (Yao et al., 2019).

3.4.3. Semantic mediator

Entities and relations in knowledge graphs are organized as triples, while large language models require natural language input. To convert structured triples into natural language, semantic mediator incorporates a unified prompt template that transforms both relevant and adjacent triples into demonstration text following standardized formatting. Concurrently, entity long text was reformatted into structured entity descriptions.

In addition to demonstration text, task instructions were incorporated into the semantic mediator to explicitly inform the large language model that its objective is to rank candidate answers based on plausibility. The model's feedback was then verified to ensure proper understanding of task requirements. Finally, the query triple $(e_h, r, ?)$ was converted into query text using the same format as demonstration text through the prompt template. These textual components were systematically organized and sequentially fed into the large language model for interaction while continuously monitoring its feedback, with implementation details illustrated in Fig. 2.

3.4.4. Preliminary reasoning

The chain-of-thought reasoning strategy was first proposed in Wei et al. (2022b), with its core idea being to guide the model to simulate the human step-by-step thinking process of "question →reasoning steps →answer," thereby improving the reasoning ability for multi-step complex tasks. Inspired by this, we introduce a key module called Preliminary Reasoning in the proposed framework. The overall workflow of this module is shown in Fig. 3. This module adopts a hybrid reasoning paradigm that integrates the advantages of CoT and ICL. Unlike the original chain-of-thought prompting method, it does not require providing step-by-step few-shot reasoning chain examples; and unlike traditional zero-shot reasoning methods, this module leverages contextual information from the Relevant triples, Adjacent triples, Candidate entities, and Entity description, significantly enhancing the logical coherence of the reasoning chain.

The module adopts a two-stage prompting strategy to extract reasoning and the final answer separately. In the first stage, the input question x is first modified into a prompted format x', using the following template: Q: [X]. A: [T]. Where [X] is the placeholder for the original question, and [T] is the placeholder for the manually written trigger sentence, used to guide the model to generate the reasoning chain. For example, if we use "provide the reason for your ranking." as the trigger sentence, the prompt becomes: Q: [question content]. A: [provide the reason for your ranking]. Then, the constructed prompt is input into the language model to generate the subsequent reasoning sentence z.

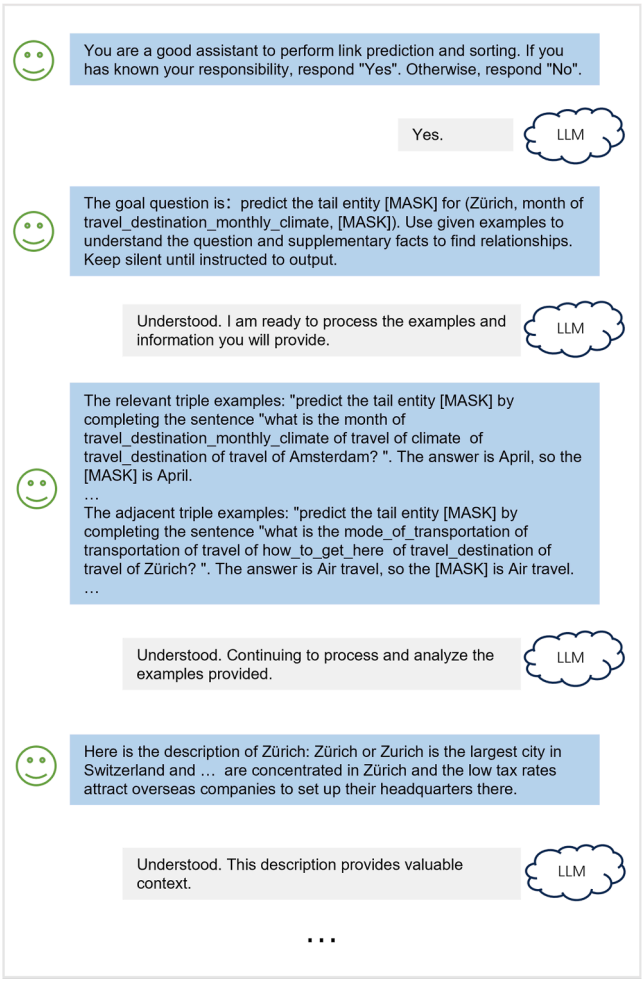


Fig. 2. Illustration of a multi-round interaction with the LLM. Send the task instruction to the LLM to clarify the primary objective; send the query text, which is the triplet to be completed; and separately send the demonstration text and entity description to provide contextual information.

In the second stage, the reasoning sentence z generated in the first step and the original prompt x are used together to extract the final answer from the language model. Specifically, the following three components are concatenated into a new prompt: `[X] [Z] [A]`. Where `[X]` is the question from the first prompt, `[Z]` is the reasoning sentence generated in the first stage, and `[A]` is a trigger sentence used to guide the language model to output the final answer. For example, use "Output the sorted order of candidate answers using the format [most possible answer | second possible answer | …| least possible answer]." as the prompt sentence. Through prompt sentence, we constrain the format of the final result output by the LLMs, facilitating subsequent processing.

Due to the context length limitations of LLMs in reasoning tasks, we cannot directly use all tens of thousands of entities in the knowledge graph as candidates for ranking. To address this, we first introduce a Candidate Filter to select the top-n most relevant entities from the complete set E, forming an initial candidate set: $Candidate\ Entities = [e_1, e_2, \ldots, e_{n-1}, e_n]$. While this strategy effectively alleviates the context window pressure on the LLM, it introduces a new challenge: the correct answer entity may not be included in the Candidate Entities, thereby impacting the final ranking accuracy. To overcome this limitation, the framework incorporates a dynamic generation module. While the LLM performs ranking inference, it is encouraged to leverage contextual information and its rich internal knowledge base to generate an additional set of potential entities: $New\ Entities = [entity_1, entity_2, \ldots, entity_k]$. By integrating this module, we dynamically generate contextually relevant candidate entities that extend beyond the scope of the base model's filtering. The generated new entities are then merged with the candidate entities set to form a dynamically expanded candidate pool, defined as follows:

$$\begin{aligned}\varepsilon_{\text{dynamic}} &= \left[e_1^{(l)}, e_2^{(l)}, \ldots, e_m^{(l)}\right] \\ &= f_{\text{llm}}\left(q, c_e^{(q)}, d(q)\right) \cup \text{Candidate Entities}[0:n] \end{aligned} \tag{4}$$

Where $q, c_e^{(q)}$, and $d(q)$ denote the question, entity description, and demonstration text.

3.5. Collaborative optimizer

In our work, we observe that LLMs may encounter several issues when performing candidate entities re-ranking tasks. Since the training data of LLMs contains inherent bias information such as historical texts and social media content, this can lead to biased outputs during the re-ranking process. In addition, when dealing with entities that have highly similar semantics, LLMs often struggle to capture subtle differences between them, resulting in insufficient discrimination ability and leading to confusion. To mitigate the bias and similar entity confusion of a single large model, The Collaborative Optimizer is proposed for refining candidate entity rankings. Specifically, the Top-M high confidence $CandidateEntities = [e_1^{'}, e_2^{'}, \ldots e_{n-1}^{'}, e_m^{'}]$ are extracted from the LLM's initial ranking results and combine them with the original query context to form a task input for the multi-agent module. The multi-agent team discusses the task, evaluates the ranking positions of candidate entities within the task, and advances reasonably justified candidates to higher positions. After multiple rounds of deliberation among the agents, the system produces: $Optimized\ Candidate = [e_1'', e_2'', \ldots, e_{n-1}'', e_m'']$.

3.5.1. LLM-based agent

Agents serve as one of the most critical components in our multi-agent evaluation module. Here, each individual LLM is treated as an autonomous agent, tasked with generating its own response based on the provided prompts. The responses from other agents are embedded into the prompt template as chat history. Once the agents are configured, multi-agent debate is initiated: Each agent autonomously receives responses from other agents. Agents sequentially provide their own reasoned replies. Notably, the entire process operates without human intervention.

3.5.2. Role playing

Each agent is endowed with a unique persona. This meticulous design ensures that every agent focuses on a distinct perspective or brings specific expertise. In this way, collective evaluation benefits from a more comprehensive viewpoint, capturing nuances and subtleties that might be overlooked by a single perspective. We primarily draw this idea from the insight that "there are a thousand Hamlets in a thousand people's eyes," meaning each individual has their own unique interpretation or perspective, especially in the evaluation of complex reasoning tasks. Therefore, diverse role definitions are also essential for the multi-agent evaluation module. Although all agents share a common prompt template, we replace the role descriptions with diversified role prompts, assigning different role personalities to different agents.

3.5.3. Communication strategy

The communication strategy determines how to maintain and manipulate the chat history among agents, which is crucial for effective interaction in a multi-agent system. Different communication strategies can be viewed as distinct approaches to managing and processing chat records, thereby influencing information exchange and interaction outcomes among agents. Inspired by Chan et al. (2023), we adopt a turn-taking summarization strategy, as illustrated in Fig. 4. In each round t of debate with k agents, when agent j takes its turn, it generates response:

$$h_t^j = \text{agent}_j(H_{t-1} \oplus [h_t^1, \ldots, h_t^{j-1}]) \tag{5}$$

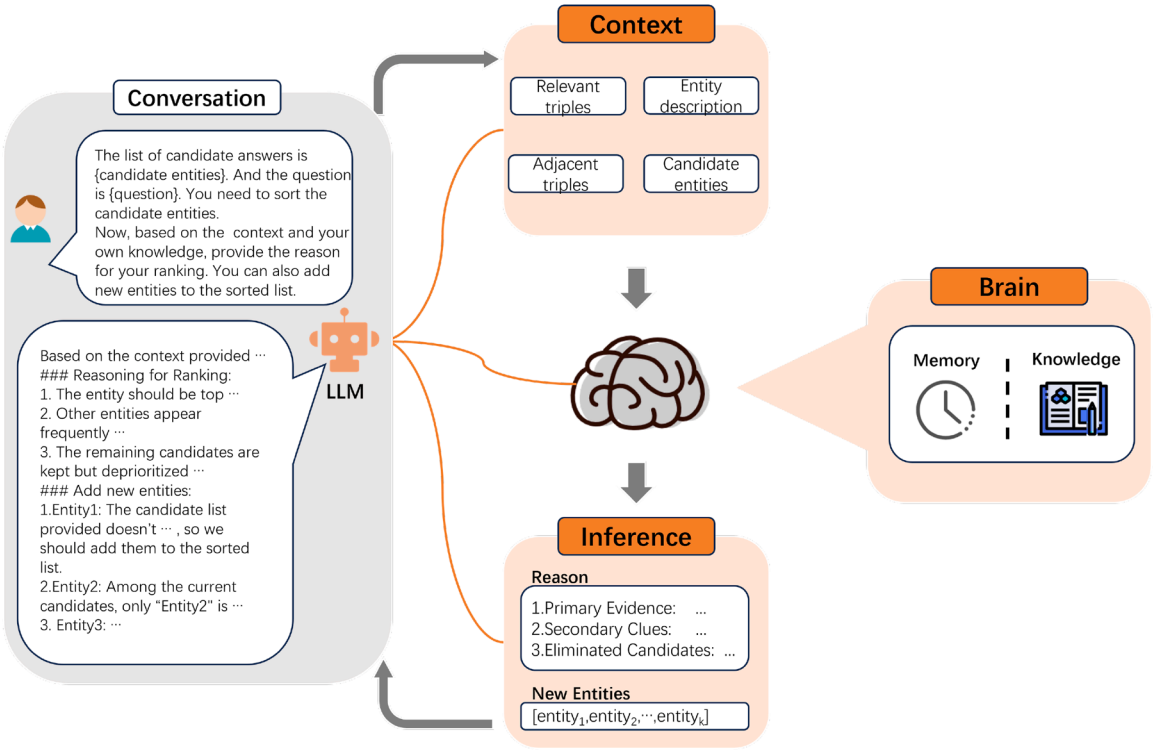


Fig. 3. Preliminary Reasoning flowchart.

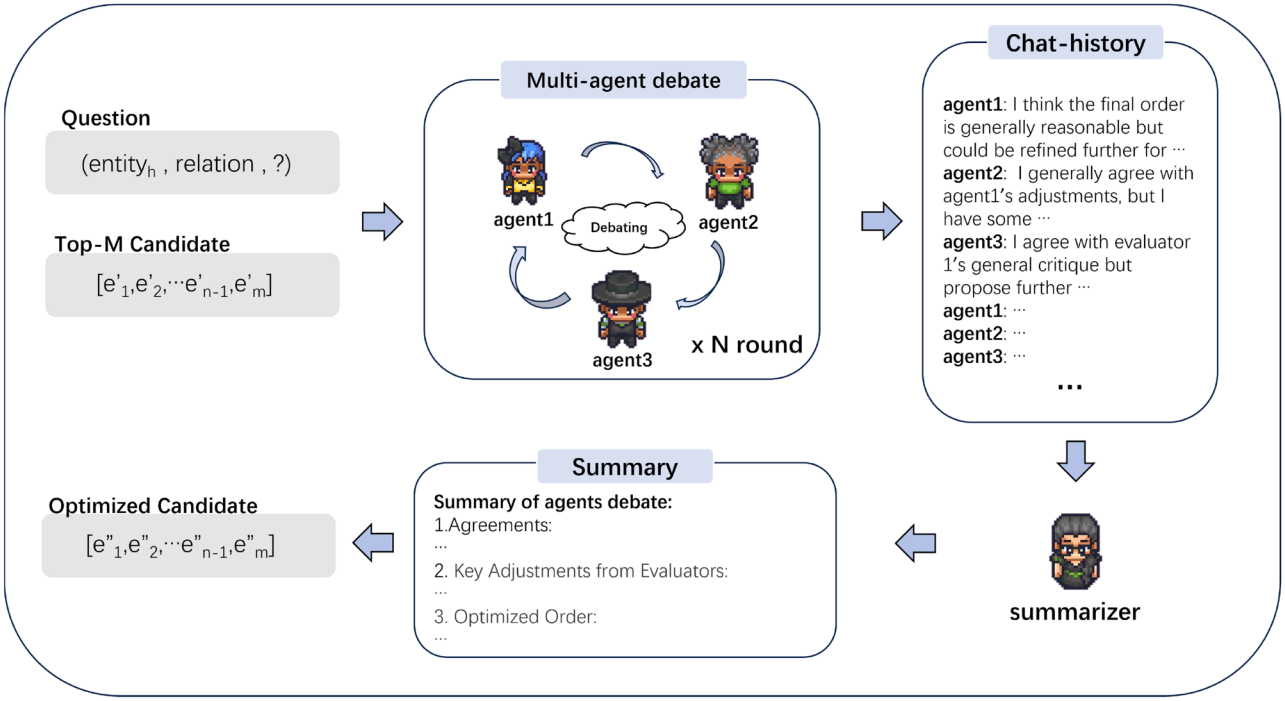


Fig. 4. Communication strategy of speaking and summarizing in sequence.

Where H_{t-1} denotes the accumulated history from all previous rounds, and $[h_t^1, \ldots, h_t^{j-1}]$ denotes the outputs of agents $1, \ldots, j-1$ in the current round t. The operator $\oplus$ represents the concatenation of history and predecessors' outputs. The debating agents take turns generating responses based on their current observations in a predetermined order. When it is a debating agent's turn to speak, we directly concatenate the previous remarks of other agents into its chat history. The agent may then supplement the discussion by referencing others' arguments, raise questions, or express its own viewpoints. After multiple rounds of discussion, all agents' remarks are collectively submitted to the summarizer, which evaluates and synthesizes them to produce the $Optimized\ Candidate = [e_1'', e_2'', \ldots, e_{n-1}'', e_m'']$.

4. Experiment

This section describes the experimental setup, datasets, baselines, and evaluation metrics, followed by experimental results and ablation studies to validate the effectiveness of our framework.

4.1. Datasets

We systematically evaluate the proposed method using two widely adopted KGC benchmark datasets FB15k-237 and WN18RR. FB15k-237 is an optimized subset of the Freebase knowledge graph (Bollacker et al., 2008), a structured, encyclopedia-like database covering multi-dimensional entity relationships across domains such as celebrities, organizations, films, and sports. To enhance model reasoning capabilities, the original dataset underwent specialized processing where inverse relations that could introduce prediction bias were removed. Inverse relations are pairs of relations in which the head and tail entities are swapped. For instance, the inverse relation pair (Academy Award for Best Director, award/award_nominee, Christopher Nolan) and (Christopher Nolan, award_nominee/award, Academy Award for Best Director) was reduced to a single relation, keeping only one direction. The detailed process for handling inverse relations can be referred to reference (Toutanova & Chen, 2015), and the processed dataset is adopted in this manuscript, ensuring only the most challenging 237 relation types were retained in the final dataset. WN18RR, an improved version of the WordNet lexical semantic network (Miller, 1995), focuses on morphological relationships in English vocabulary (e.g., synonymy, antonymy, hypernymy). By eliminating test-set leakage issues present in the WN18 dataset, it provides a more rigorous evaluation benchmark. Detailed statistics for these datasets, including entity scale, relation diversity, and triple distribution, are presented in Table 1.

4.2. Metrics

We employ widely used evaluation metrics: Hits@k (k = 1, 3, 10) and MRR (Mean Reciprocal Rank) to assess the performance of our proposed method. Hits@k measures the proportion of query triples where the ground-truth entity is ranked among the top k positions. MRR calculates the average reciprocal rank of the ground-truth entities across all queries. Higher values for these metrics indicate better performance.

4.3. Baselines

We compare our framework with four categories of baseline methods: (1) Embedding-based methods, including TransE (Bordes et al., 2013), DistMult (Yang et al., 2015), RotatE (Sun et al., 2019), ComplEx (Trouillon et al., 2016), TuckER (Balazevic et al., 2019), CompGCN (Vashishth et al., 2019), HAKE (Zhang et al., 2020) and HittER (Chen et al., 2021); (2) Text-based methods, such as KG-BERT (Yao et al., 2019), StAR (Wang et al., 2021), MTL-KGC (Kim et al., 2020) and CoLE (Liu et al., 2022); (3) Fine-tuning methods, including KGR (Li et al., 2024b), GS-KGC (Yang et al., 2025), CSProm-KG-CD (Li et al., 2024a) and FtG (Liu et al., 2025); (4) Training-free methods, including DeepSeek (Zhu et al., 2024), KICGPT (Wei et al., 2023). Note that LLM-based methods are categorized into two types: Fine-tuning methods and Training-free methods.

Table 1
Statistical information of the dataset.

Dataset	#Entities	#Relations	#Train	#Valid	#Test
FB15k-237	14,541	237	272,115	17,535	20,466
WN18RR	40,943	11	86,835	3034	3134

4.4. Implementation details

In our framework, we select RotatE as the filtering KGC model and DeepSeek (DeepSeek-v3; DeepSeek-v2.5) along with Qwen (Qwen2.5-72B-Instruct) as the LLMs. Although the proposed method is compatible with various embedding models, we prioritize lightweight base models for efficiency. The hyperparameters of RotatE align with those in Sun et al. (2019). For the LLMs used in this study, we employ third-party APIs with default parameter settings, including: temperature, top-p, presence penalty and frequency penalty.

4.5. Experimental results

The experimental results are shown in Table 2. As can be seen, the proposed ReaCo-KGC framework achieves state-of-the-art performance on most metrics for the FB15k-237 and WN18RR datasets.

On the FB15k-237 dataset, our model achieves the best overall performance to date, with an MRR of 0.466, significantly outperforming all baseline methods. Although the Hits@1 score (0.392) is slightly lower than that of KGR (0.400), this is because the KGR method primarily relies on a fine-tuned large language model to select the entity with the highest confidence. As a result, KGR demonstrates particular advantages in the Hits@1 metric. In contrast, our proposed Collaborative Optimizer module performs global optimization on the candidate entity list and adjusts the overall ranking. Consequently, our model achieves scores of 0.466 for MRR, 0.496 for Hits@3, and 0.630 for Hits@10, all exceeding KGR's corresponding scores of 0.456, 0.476, and 0.569.

On the WN18RR dataset, our model also achieves outstanding performance, reaching state-of-the-art levels across all evaluation metrics: the MRR is 0.607, while Hits@1, Hits@3, and Hits@10 reach 0.539, 0.646, and 0.751, respectively. Compared with the strongest competitor, KICGPT (Hits@3 = 0.620, Hits@10 = 0.716), existing approaches tend to suffer from information dilution when LLMs process large volumes of context at once. In contrast, our Knowledge Guide module extracts key information and generates reasoning processes from the context, thereby enhancing the LLM's reasoning ability in complex tasks and helping it focus on the most relevant content. As a result, our model achieves significant improvements across all metrics, with Hits@3 and Hits@10 increasing by more than 2.6 % and 3.5 %, respectively.

Overall, the experimental results show that our proposed framework delivers highly consistent and stable performance across different datasets. On one hand, the relatively small performance gaps between Hits@1, Hits@3, and Hits@10 demonstrate the robustness and adaptability of our ranking strategy. On the other hand, unlike methods such as KGR that focus solely on selecting the best entity, our approach introduces a multi-agent collaboration mechanism to globally optimize the ranking of candidate entities. Additionally, through the use of Knowledge Guide for semantic-aware design, we fully exploit the LLM's ability to understand entity context, enabling more precise relation modeling and semantic reasoning. Furthermore, the Preliminary Reasoning module encourages the inclusion of novel candidate entities, overcoming the limitations of initial rankings from base models and significantly expanding the candidate space. Together, these design innovations contribute to our model's superior performance across multiple datasets, demonstrating its broad applicability and high value in real-world knowledge graph completion tasks.

4.6. Ablation studies

In this experiment, we conducted ablation studies on the proposed framework using the FB15k-237 and WN18RR datasets to validate the effectiveness of each component. The experimental results are presented in Table 3.

To validate the effectiveness of Preliminary Reasoning module, we removed it in the ablation study (w/o Preliminary Reasoning). The results show a performance decline, demonstrating the importance of the

Table 2
Comparison results on FB15k-237 and WN18RR. Best results are in bold, and the second best are underlined. The same large language model (DeepSeek-V3) is employed for the training-free methods.

Models	FB15k-237				WN18RR			
	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10
Embedding-based methods								
TransE (NeurIPS2013 (Bordes et al., 2013))	0.279	0.198	0.376	0.441	0.243	0.043	0.441	0.532
DistMult (ICLR2015 (Yang et al., 2015))	0.241	0.155	0.263	0.419	0.430	0.390	0.440	0.490
ComplEx (ICML2016 (Trouillon et al., 2016))	0.247	0.158	0.275	0.428	0.440	0.410	0.460	0.510
RotatE (ICLR2019 (Sun et al., 2019))	0.338	0.241	0.375	0.533	0.476	0.428	0.492	0.571
TuckER (EMNLP2019 (Balazevic et al., 2019))	0.358	0.266	0.394	0.544	0.470	0.443	0.482	0.526
CompGCN (ICLR2019 (Vashishth et al., 2019))	0.355	0.264	0.390	0.535	0.479	0.443	0.494	0.546
HAKE (AAAI2020 (Zhang et al., 2020))	0.346	0.250	0.381	0.542	0.497	0.452	0.516	0.582
HittER (EMNLP2021 (Chen et al., 2021))	0.344	0.246	0.380	0.535	0.496	0.449	0.514	0.586
Text-based methods								
KG-BERT (arXiv2019 (Yao et al., 2019))	–	–	–	0.420	0.216	0.041	0.302	0.524
StAR (WWW2021 (Wang et al., 2021))	0.296	0.205	0.322	0.482	0.401	0.243	0.491	0.709
MTL-KGC (COLING2020(Kim et al., 2020))	0.267	0.172	0.298	0.458	0.331	0.203	0.383	0.597
CoLE (CIKM2022(Liu et al., 2022))	0.387	0.293	0.426	0.570	0.585	0.532	0.607	0.689
LLM-based methods								
Training-free methods								
DeepSeek-V3$_{zero-shot}$ (WWW2024 (Zhu et al., 2024))	–	0.107	–	–	–	0.103	–	–
DeepSeek-V3$_{one-shot}$ (WWW2024 (Zhu et al., 2024))	–	0.189	–	–	–	0.128	–	–
KICGPT (EMNLP2023 (Wei et al., 2023))	0.430	0.347	0.471	0.597	0.544	0.443	0.620	0.716
Fine-tuning methods								
GS-KGC (INFFUS2025 (Yang et al., 2025))	–	0.280	0.426	–	–	0.346	0.516	–
CSProm-KG-CD (EACL2024 (Li et al., 2024a))	0.372	0.288	0.410	0.530	0.559	0.508	0.578	0.660
FtG (COLING2025 (Liu et al., 2025))	0.392	0.321	0.413	0.542	–	–	–	–
KGR (arXiv2024 (Li et al., 2024b))	0.456	**0.400**	0.476	0.569	0.520	0.495	0.520	0.550
ReaCo-KGC	**0.466**	0.392	**0.496**	**0.630**	**0.607**	**0.539**	**0.646**	**0.751**

Table 3
Experimental results for ablation studies.

Ablation	FB15k-237				WN18RR			
	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10
FGC-KGC	0.464	0.391	0.493	0.629	0.607	0.539	0.646	0.751
w/o Preliminary Reasoning	0.430	0.347	0.471	0.597	0.544	0.443	0.620	0.716
w/o Knowledge Guide	0.395	0.308	0.435	0.578	0.402	0.299	0.458	0.609
w/o Collaborative Optimizer	0.450	0.360	0.497	0.629	0.597	0.512	0.654	0.752

preliminary reasoning strategy proposed in Section 3.4.4. By incorporating structured reasoning, the LLM is guided to perform context summarization and logical structuring, enabling more accurate prediction of missing triples.

To validate the effectiveness of Knowledge Guide module, we removed it in the ablation study (w/o Knowledge Guide). Without providing any contextual information, the LLM was directly tasked with re-ranking candidate entities. The experimental results exhibited a significant drop, confirming the critical role of the knowledge guide introduced in Section 3.4. Providing rich contextual information enables the LLM to better understand entities and relations, thereby reducing hallucination.

To validate the effectiveness of Collaborative Optimizer module, we removed it in the ablation study (w/o Collaborative Optimizer). The results revealed a substantial decrease in Hits@1, highlighting the effectiveness of leveraging multi-agent interactions to optimize entity ranking. The collaboration among multiple agents enables more precise positioning of top-ranked candidate answers.

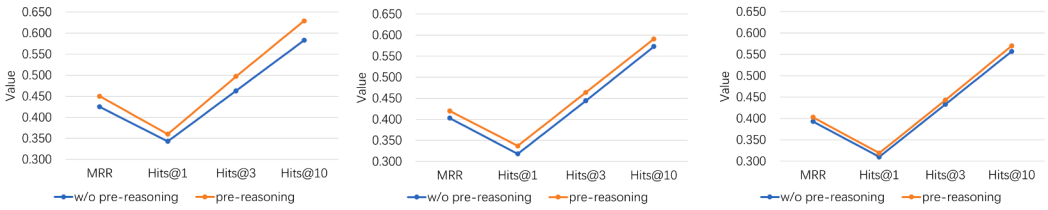
4.7. Analysis on preliminary reasoning

To further validate the effectiveness and generalizability of the Preliminary Reasoning module in our framework, we conducted systematic experiments on LLMs of varying scales, including Deepseek-V3 (671B), Deepseek-V2.5 (236B), and Qwen2.5 (72B). As shown in Fig. 5, the x-axis represents the four evaluation metrics: MRR, Hits@1, Hits@3, and Hits@10, while the y-axis indicates the corresponding metric values. Higher values represent better performance. after introducing the Preliminary Reasoning module, all three models achieved performance improvements to varying degrees across all evaluation metrics on the FB15k-237 datasets, clearly demonstrating the general positive impact of the preliminary reasoning strategy on ranking tasks.

In terms of specific metrics, the Preliminary Reasoning module significantly enhanced Hits@10 in particular. For instance, Deepseek-V3's Hits@10 improved from approximately 0.58 to 0.63, indicating the model's stronger ability to distinguish among candidate entities in overall ranking. Moreover, the improvements in MRR and Hits@3 were also notable, further showing that the automatically generated CoT can effectively guide the model toward more structured semantic reasoning, thereby improving the overall ranking quality.

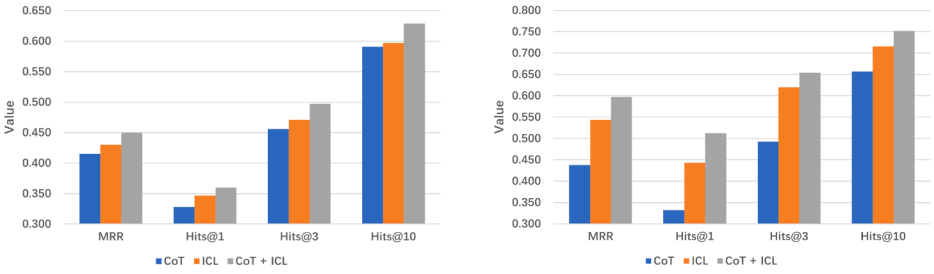
It is worth noting that the performance gains show a clear positive correlation with model scale. Deepseek-V3 (671B), being the largest model, exhibited the most substantial improvements across all metrics. While Qwen2.5 (72B) also benefited from the Preliminary Reasoning module, the gains were relatively limited, especially in Hits@1. This suggests that smaller-scale models still face challenges in generating logically coherent and semantically consistent chains of thought, aligning with prior research (Wei et al., 2022b), which posits that Chain-of-Thought reasoning is an emergent capability associated with larger model sizes.

The Preliminary Reasoning module effectively enhances LLMs understanding of complex semantic relations, and demonstrates stronger reasoning and ranking performance, especially in large models, making it highly valuable for high-quality knowledge graph completion tasks.



(a) Deepseek-V3 (671B) (b) Deepseek-V2.5 (236B) (c) Qwen2.5 (72B)

Fig. 5. Experimental results of models of different sizes after incorporating preliminary reasoning.



(a) FB15k-237 datasets (b) WN18RR datasets

Fig. 6. Experimental results on chain-of-thought combined with in-context Learning.

4.8. Analysis on knowledge guidance

In this experiment, we preliminarily investigated the integration of large language models' ICL capability with the CoT mechanism. Specifically, we systematically ablated key components of the Knowledge Guidance module by: (1) removing prompt engineering while retaining the two-stage prompt, and (2) eliminating the two-stage prompt while preserving prompt engineering. As illustrated in Fig. 6, we conducted a comparative evaluation of different methodologies (CoT-only, ICL-only, and their combination CoT + ICL) on the FB15k-237 and WN18RR datasets. The x-axis represents the four evaluation metrics: MRR, Hits@1, Hits@3, and Hits@10, while the y-axis indicates the corresponding metric values. Higher values represent better performance. As shown, the combined approach consistently outperforms the individual methods across all evaluation metrics, highlighting the complementary strengths of chain-of-thought reasoning and in-context learning.

On the FB15k-237 dataset, the combined method improves the Hits@1 metric by 3.2 % compared to ICL, increasing from approximately 0.34 to 0.37. This indicates that the hybrid approach still brings some benefits in fact-based reasoning scenarios. However, since this dataset primarily relies on factual relationships with weaker logical structures, improvements in other metrics such as MRR and Hits@10 are relatively limited–for example, Hits@10 only rises from about 0.59 to 0.63–showing that gains are constrained by the nature of the dataset.

The advantages of the combined approach are much more prominent on the WN18RR dataset, which is centered around lexical semantic and hierarchical relations and is thus more suited to logical reasoning. The results show that Hits@1 increases by approximately 18 %, from 0.28 to 0.33, and MRR jumps from 0.46 to 0.60, indicating substantial performance gains. Additionally, Hits@10 reaches a peak of around 0.76, further confirming the effectiveness of the hybrid method in complex reasoning tasks.

In summary, the experimental results demonstrate that integrating chain-of-thought reasoning with in-context learning significantly outperforms using either method alone. This combination enhances the model's generalization ability and stability across different reasoning tasks, especially in datasets with strong logical structures.

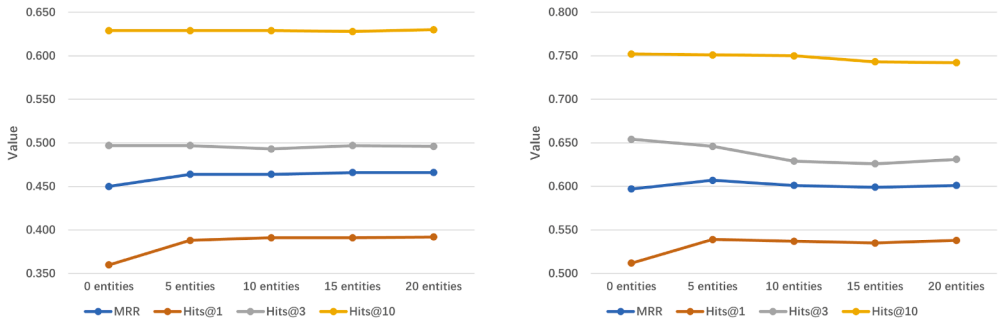
4.9. Analysis on collaborative optimizer

In our study, we investigated whether the optimization of candidate entity rankings by multi-agents is influenced by the number of candidate entities. Fig. 7 illustrates the impact of varying the number of candidate entities on Collaborative Optimizer performance across the FB15k-237 and WN18RR datasets. The x-axis represents different candidate entity settings, including 0, 5, 10, 15, and 20 entities. The y-axis shows the values of four evaluation metrics: MRR, Hits@1, Hits@3, and Hits@10.

On the FB15k-237 dataset, as the number of candidate entities increases from 5 to 20, the Hits@1 and MRR metrics show steady improvements. Specifically, Hits@1 increases from 0.36 to approximately 0.39, while MRR rises from 0.44 to around 0.46, indicating that the multi-agent approach effectively enhances Top-1 ranking accuracy. However, Hits@3 and Hits@10 remain largely unchanged, staying around 0.49 and 0.63, respectively. This suggests that while the multi-agent mechanism improves precision at the top rank, it brings limited benefit to broader ranking performance.

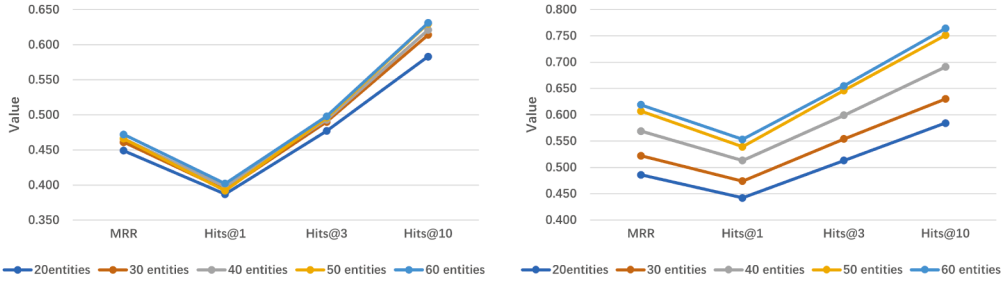
On the WN18RR dataset, the multi-agent approach achieves the best performance when the number of candidate entities is limited to 5. Hits@1 improves to about 0.54, and MRR peaks at around 0.61–both significantly higher than the results without the multi-agent module (Hits@1 at 0.51 and MRR at 0.60). However, as the number of candidate entities increases to 10, 15, and 20, both Hits@3 and Hits@10 show a continuous decline–from around 0.66 and 0.75 down to 0.63 and 0.73, respectively. This trend indicates that an excessive number of entities may introduce irrelevant information, reducing the effectiveness of inter-agent reasoning. Moreover, limitations in the LLM's context window may prevent accurate identification and optimization of the correct target entity, further affecting overall ranking performance.

The multi-agent mechanism significantly enhances Top-1 ranking accuracy, especially when the number of candidate entities is kept between 5 and 10. However, including too many candidates does not lead to further gains and may negatively impact broader ranking metrics such as Hits@3 and Hits@10. Therefore, to balance performance and efficiency, the optimal number of candidate entities should be limited to approximately 5-10.



(a) FB15k-237 datasets (b) WN18RR datasets

Fig. 7. The impact of different entity numbers on multi-agent.



(a) FB15k-237 datasets (b) WN18RR datasets

Fig. 8. The impact of the number of candidate entities on ReaCoKGC.

4.10. Analysis on the number of candidate entities

In our study, candidate entities were screened by the base KGC model (RotatE). Fig. 8 presents the impact of varying the number of candidate entities on the overall performance across the FB15k-237 and WN18RR datasets. The x-axis represents the evaluation metrics (MRR, Hits@1, Hits@3, and Hits@10), while the y-axis denotes the corresponding performance values. Each line corresponds to a specific number of candidate entities, including 20, 30, 40, 50, and 60.

On the FB15k-237 dataset, all evaluation metrics exhibit a gradual improvement as the number of candidate entities increases from 20 to 60. Specifically, MRR rises from 0.44 to 0.47, and Hits@1 improves from 0.38 to 0.40, demonstrating that expanding the candidate pool helps the model more accurately capture the ground-truth entity. Similarly, Hits@3 and Hits@10 increase from 0.47 to 0.49 and from 0.58 to 0.63, respectively, suggesting that broader candidate coverage enhances the model's overall ranking performance. However, the performance gain becomes marginal beyond 50 entities, indicating that the model begins to reach a saturation point where further increases contribute little to prediction accuracy.

On the WN18RR dataset, a similar trend can be observed. When the number of candidate entities increases from 20 to 50, MRR and Hits@1 steadily improve–from 0.48 and 0.44 to 0.60 and 0.53, respectively–while Hits@3 and Hits@10 rise from 0.51 and 0.58 to 0.64 and 0.75. This indicates that expanding the candidate set improves the model's ability to locate correct entities among difficult relation contexts. Nonetheless, when the candidate number reaches 60, only slight improvements are observed, while computational costs increase notably due to longer input sequences for the LLM.

Overall, increasing the number of candidate entities can effectively improve model performance at first, but excessive expansion leads to diminishing returns and higher inference costs. Considering both predictive accuracy and computational efficiency, we finally adopt 50 candidate entities in all subsequent experiments as an optimal configuration that maintains strong performance while avoiding unnecessary resource overhead.

4.11. The limitations of the proposed approach

Although the proposed ReaCo-KGC framework demonstrates strong performance, it still faces limitations: the current experiments are conducted under a transductive setting, where all entities and relations in the test phase have already appeared in the training data. This limits the framework's ability to handle unseen entities and novel relations. That is, our current work mainly focuses on static knowledge graphs, without considering the temporal dynamics of real-world knowledge. In addition, the complex multi-stage prompting and multi-agent debate mechanisms introduce significant reasoning latency, making it difficult to apply the framework to scenarios with high real-time requirements.

Conclusion

In this paper, we propose a reasoning-enhanced and interaction-corrective framework based on large language models for knowledge graph completion (ReaCo-KGC). The framework consists of three key components: candidate filter module performs rapid screening of candidate entities through lightweight embedding computations; knowledge guide module integrates contextual information and employs a two-stage prompting mechanism to generate chain-of-thought reasoning, guiding the LLM to perform contextual summarization and logical organization; and collaborative optimizer module refines results through multi-agent collaborative interactions, correcting confusion caused by semantic similarity. Experimental results demonstrate that the ReaCo-KGC framework, integrating these three modules, significantly improves knowledge graph completion performance, achieving efficient and reliable completion.

In our future work, we will explore inductive and temporal knowledge graph completion by leveraging the generalization and reasoning capabilities of large language models, enabling the framework to handle unseen entities and evolving facts in dynamic real-world scenarios.

CRediT authorship contribution statement

Tingting Jiang: Conceptualization, Methodology, Formal analysis, Writing – original draft, Writing – review & editing; **Suqing Wu:** Methodology, Investigation, Data curation, Software, Validation, Writing – original draft; **Shuai Yang:** Software, Visualization, Resources, Writing – review & editing; **Xiaohui Yuan:** Resources, Writing – review & editing, Supervision, Funding acquisition; **Lichuan Gu:** Resources, Writing – review & editing, Supervision, Funding acquisition; **Xindong Wu:** Resources, Writing – review & editing, Supervision, Funding acquisition.

Data availability

Data will be made available on request.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was partly supported by the National Natural Science Foundation of China (no 32501770, no 62306008, no 32472007), the Natural Science Research Project of Education Department of Anhui Province of China (no 2023AH051020).

Appendix A.

A.1. Interaction process

Representative samples were selected from the FB15k-237 and WN18RR datasets to demonstrate the complete interaction process, with details provided in Tables 4 and 5.

Table 4
Examples of interaction process on the FB15k-237 dataset.

Steps	Prompt Template	LLM
Responsibility Description	You are a good assistant to perform link prediction and sorting. Given a goal question and a list of candidate answers of this question. Your need to order these candidate answers in the list to let candiate answers which are more possible to be the answer of the question more prior. If you has known your responsibility, respond “Yes”. Otherwise, respond “No”. Do not output anything except “Yes” and “No”.	Yes.
Question and Demonstration Description	The goal question is: predict the tail entity [MASK] from the given (RoboCop, language of film of film of , [MASK]) by completing the sentence “what is the language of film of film of RoboCop? The answer is ”. To sort the candidate answers, typically you would need to refer to some other examples that may be similar to or related to the question. Part of given examples are similar to the goal question, you should analogy them to understand potential meaning of the goal question. Another part of given facts contain supplementary information, keep capturing these extra information and mining potential relationships among them to help the sorting.	Understood. I will analyze the examples, extract useful information, and use it to sort the candidate answers effectively. Please proceed with the examples.
Multiple Demonstrations	Examples used to Analogy: “predict the tail entity [MASK] from the given (Quantum of Solace, language of film of film of , [MASK]) by completing the sentence ”what is the language of film of film of Quantum of Solace? The answer is “. The answer is Spanish Language, so the [MASK] is Spanish Language. Examples give supplement information: ”predict the tail entity [MASK] from the given (RoboCop, genre of film of film of , [MASK]) by completing the sentence “what is the genre of film of film of RoboCop? The answer is ”. The answer is Stop motion, so the [MASK] is Stop motion.	Understood. Continuing to process and analyze the examples provided.
Entity Description	Here is the description of RoboCop: Sergeant Warren Reed is a fictional character from the film series RoboCop, RoboCop 2, RoboCop 3.	Understood. This description provides valuable context.
Preliminary Reasoning	The list of candidate answers is [Russian Language,English Language,Japanese Language,Arabic Language,Spanish Language,Chinese language,Italian Language,…,Tagalog language,Bulgarian Language,Thai Language]. And the question is predict the tail entity [MASK] from the given (RoboCop, language of film of film of , [MASK]) by completing the sentence “what is the language of film of film of RoboCop? The answer is ”. Now, based on the previous examples, entity description, and your own knowledge and thinking, sort the list to let the candidate answers which are more possible to be the true answer of the question more prior. If you believe the most likely answer is not in the candidate list, you may also add new answers to the sorted list. Let’s think step by step.	Based on the analysis of the examples and supplementary information, here’s the step-by-step reasoning for sorting the candidate answers: 1. **Primary Evidence from Examples**: …2. **Pattern from Analogous Examples**: …3. **Supplementary Information**: …4. **Candidate Evaluation**: …5. **Conclusion**: The clear primary answer …
Final Query	Output the sorted order of candidate answers using the format “[most possible answer | second possible answer | …| least possible answer]” and please start your response with “The final order:”. Do not output anything except the final order. Note your output sorted order should contain all the candidates in the list.	The final order: [English Language | Russian Language | Japanese Language | Arabic Language |…| Tagalog language | Bulgarian Language | Thai Language]

Table 5
Examples of interaction process on the WN18RR dataset.

Steps	Prompt Template	LLM
Responsibility Description	You are a good assistant to perform link prediction and sorting. Given a goal question and a list of candidate answers of this question. Your need to order these candidate answers in the list to let candiate answers which are more possible to be the answer of the question more prior. If you has known your responsibility, respond “Yes”. Otherwise, respond “No”. Do not output anything except “Yes” and “No”.	Yes.
Question and Demonstration Description	The goal question is: predict the tail entity [MASK] from the given (telephone, _verb_group of , [MASK]) by completing the sentence “what is the _verb_group of telephone? The answer is ”. To sort the candidate answers, typically you would need to refer to some other examples that may be similar to or related to the question. Part of given examples are similar to the goal question, you should analogy them to understand potential meaning of the goal question. Another part of given facts contain supplementary information, keep capturing these extra information and mining potential relationships among them to help the sorting.	Understood. I will analyze the examples, extract useful information, and use it to sort the candidate answers effectively. Please proceed with the examples.
Multiple Demonstrations	Examples used to Analogy: “predict the tail entity [MASK] from the given (snap, _verb_group of , [MASK]) by completing the sentence ”what is the _verb_group of snap? The answer is “. The answer is take, so the [MASK] is take. Examples give supplement information: ”predict the tail entity [MASK] from the given (telephone, _synset_domain_topic_of of , [MASK]) by completing the sentence “what is the _synset_domain_topic_of of telephone? The answer is ”. The answer is telephony, so the [MASK] is telephony.	Understood. Continuing to process and analyze the examples provided.
Entity Description	Here is the description of telephone: telephone, get or try to get into communication (with someone) by telephone; “I tried to call you all night”; “Take two aspirin and call me in the morning”.	Understood. This description provides valuable context.
Preliminary Reasoning	The list of candidate answers is [telephone,call,war machine,urban center,cell phone,…,dissolve,electronics,ring,spot,take in,work]. And the question is predict the tail entity [MASK] from the given (telephone, _verb_group of , [MASK]) by completing the sentence “what is the _verb_group of telephone? The answer is ”. Now, based on the previous examples, entity description, and your own knowledge and thinking, sort the list to let the candidate answers which are more possible to be the true answer of the question more prior. If you believe the most likely answer is not in the candidate list, you may also add new answers to the sorted list. Let’s think step by step.	Based on the examples and supplementary information provided, here’s the step-by-step reasoning for sorting the candidate answers for the question “what is the _verb_group of telephone?”: 1. **Pattern Recognition from Examples**: …2. **Key Observations**: …3. **Priority Sorting**: - **Top Priority**: Verbs directly interchangeable with “telephone”: …4. **Remaining Candidates**: …5. **Final Sorted List** (most likely first): …This ordering reflects that: …
Final Query	Output the sorted order of candidate answers using the format “[most possible answer | second possible answer | …| least possible answer]” and please start your response with “The final order:”. Do not output anything except the final order. Note your output sorted order should contain all the candidates in the list.	The final order: [call | ring | dial | call in |…| lead | statistics | music | weep | wed | fall down | fall]

References

Balazevic, I., Allen, C., & Hospedales, T. (2019). TuckER: Tensor factorization for knowledge graph completion. In *2019 conference on empirical methods in natural language processing and 9th international joint conference on natural language processing (EMNLP-IJCNLP)* (pp. 5184–5193).

Bollacker, K., Evans, C., Paritosh, P., Sturge, T., & 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)* (pp. 1247–1250).

Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., & Yakhnenko, O. (2013). Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems (neurIPS)* (pp. 2787–2795).

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., & Amodei, D. (2020). Language models are few-shot learners. In *Advances in neural information processing systems (neurIPS)* (pp. 1877–1901) .

Chan, C.-M., Chen, W., Su, Y., Yu, J., Xue, W., Zhang, S., Fu, J., & Liu, Z. (2023). Chateval: Towards better LLM-based evaluators through multi-agent debate. In *International conference on learning representations (ICLR)*.

Chen, G., & Xie, X. (2023). Ml-kgcl: Multi-level knowledge graph contrastive learning for recommendation. In *International conference on database systems for advanced applications (DASFAA)* (pp. 253–268). Springer.

Chen, S., Liu, X., Gao, J., Jiao, J., Zhang, R., & Ji, Y. (2021). HittER: Hierarchical transformers for knowledge graph embeddings. In *Proceedings of the 2021 conference on empirical methods in natural language processing (EMNLP)* (pp. 10395–10407).

Huang, X., Zhang, J., Li, D., & Li, P. (2019). Knowledge graph embedding based question answering. In *Proceedings of the twelfth ACM international conference on web search and data mining (WSDM)* (pp. 105–113).

Huang, X., Zhang, J., Xu, Z., Ou, L., & Tong, J. (2021). A knowledge graph based question answering method for medical domain. *PeerJ Computer Science*, *7*, e667.

Kim, B., Hong, T., Ko, Y., & Seo, J. (2020). Multi-task learning for knowledge graph completion with pre-trained language models. In *Proceedings of the 28th international conference on computational linguistics (COLING)* (pp. 1737–1743).

Kojima, T., Gu, S. S., Reid, M., Matsuo, Y., & Iwasawa, Y. (2022). Large language models are zero-shot reasoners. In *Advances in neural information processing systems (neurIPS)* (pp. 22199–22213).

Li, D., Tan, Z., Chen, T., & Liu, H. (2024a). Contextualization distillation from large language model for knowledge graph completion. In *Findings of the association for computational linguistics: EACL* (pp. 458–477).

Li, G., Hammoud, H., Itani, H., Khizbullin, D., & Ghanem, B. (2023). Camel: Communicative agents for“ mind” exploration of large language model society. In *Advances in neural information processing systems (neurIPS)* (pp. 51991–52008).

Li, J., Peng, H., & Li, L. (2025). Sublinear smart semantic search based on knowledge graph over encrypted database. *Computers & Security*, *151*, 104319.

Li, M., Yang, C., Xu, C., Jiang, X., Qi, Y., Guo, J., Leung, H.-f., & King, I. (2024b). Retrieval, reasoning, re-ranking: A context-enriched framework for knowledge graph completion. arXiv preprint arXiv:2411.08165.

Li, W., Ge, J., Feng, W., Zhang, L., Li, L., & Wu, B. (2024c). Sikgc: Structural information prompt based knowledge graph completion with large language models. In *2024 IEEE international conference on systems, man, and cybernetics (SMC)* (pp. 1116–1123). IEEE.

Liang, T., He, Z., Jiao, W., Wang, X., Wang, Y., Wang, R., Yang, Y., Shi, S., & Tu, Z. (2024). Encouraging divergent thinking in large language models through multi-agent debate. In *Proceedings of the 2024 conference on empirical methods in natural language processing (EMNLP)* (pp. 17889–17904).

Liu, B., Zhang, J., Lin, F., Yang, C., & Peng, M. (2025). Filter-then-generate: Large language models with structure-text adapter for knowledge graph completion. In *Proceedings of the 31st international conference on computational linguistics (COLING)* (pp. 11181–11195).

Liu, Y., Sun, Z., Li, G., & Hu, W. (2022). I know what you do not know: Knowledge graph embedding via co-distillation learning. In *Proceedings of the 31st ACM international conference on information & knowledge management (CIKM)* (pp. 1329–1338).

Liu, Y., Tian, X., Sun, Z., & Hu, W. (2024). Finetuning generative large language models with discrimination instructions for knowledge graph completion. In *International semantic web conference* (pp. 199–217). Springer.

Mandi, Z., Jain, S., & Song, S. (2024). Roco: Dialectic multi-robot collaboration with large language models. In *2024 IEEE international conference on robotics and automation (ICRA)* (pp. 286–299). IEEE.

Miller, G. A. (1995). Wordnet: A lexical database for english. *Communications of the ACM*, *38*(11), 39–41.

Perevalov, A., Chinchghare, J., Krishna, M., Sharma, S., Lal, A. N., Deshwal, A., Both, A., & Ngonga-Ngomo, A.-C. (2024). Class mate: Cross-lingual semantic search for material science driven by knowledge graphs. In *European semantic web conference (ESWC)* (pp. 281–285). Springer.

Rae, J. W., Borgeaud, S., Cai, T., Millican, K., Hoffmann, J., Song, F., Aslanides, J., Henderson, S., Ring, R., Young, S., Rutherford, E., Hennigan, T., Menick, J., Cassirer, A., Powell, R., van den Driessche, G., Hendricks, L. A., Rauh, M., Huang, P.-S., Glaese, A., Welbl, J., Dathathri, S., Huang, S., Uesato, J., Mellor, J., Higgins, I., Creswell, A., McAleese, N., Wu, A., Elsen, E., Jayakumar, S., Buchatskaya, E., Budden, D., Sutherland, E., Simonyan, K., Paganini, M., Sifre, L., Martens, L., Li, X. L., Kuncoro, A., Nematzadeh, A., Gribovskaya, E., Donato, D., Lazaridou, A., Mensch, A., Lespiau, J.-B., Tsimpoukelli, M., Grigorev, N., Fritz, D., Sottiaux, T., Pajarskas, M., Pohlen, T., Gong, Z., Toyama, D., de Masson d'Autume, C., Li, Y., Terzi, T., Mikulik, V., Babuschkin, I., Clark, A., de Las Casas, D., Guy, A., Jones, C., Bradbury, J., Johnson, M., Hechtman, B., Weidinger, L., Gabriel, I., Isaac, W., Lockhart, E., Osindero, S., Rimell, L., Dyer, C., Vinyals, O., Ayoub, K., Stanway, J., Bennett, L., Hassabis, D., Kavukcuoglu, K., & Irving, G. (2021). Scaling language models: Methods, analysis & insights from training gopher. arXiv preprint arXiv:2112.11446 .

Sun, Z., Deng, Z.-H., Nie, J.-Y., & Tang, J. (2019). Rotate: Knowledge graph embedding by relational rotation in complex space. In *International conference on learning representations (ICLR)*.

Toutanova, K., & Chen, D. (2015). Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd workshop on continuous vector space models and their compositionality* (pp. 57–66).

Trouillon, T., Welbl, J., Riedel, S., Gaussier, É., & Bouchard, G. (2016). Complex embeddings for simple link prediction. In *International conference on machine learning (ICML)* (pp. 2071–2080). PMLR.

Vashishth, S., Sanyal, S., Nitin, V., & Talukdar, P. (2019). Composition-based multi-relational graph convolutional networks. In *International conference on learning representations (ICLR)*.

Wang, B., Shen, T., Long, G., Zhou, T., Wang, Y., & Chang, Y. (2021). Structure-augmented text representation learning for efficient knowledge graph completion. In *Proceedings of the web conference (WWW)* (pp. 1737–1748).

Wang, Y., Hu, M., Huang, Z., Li, D., Yang, D., & Lu, X. (2024a). Kc-genre: A knowledge-constrained generative re-ranking method based on large language models for knowledge graph completion. In *Proceedings of the 2024 joint international conference on computational linguistics, language resources and evaluation (COLING-LREC)* (pp. 9668–9680).

Wang, Z., Mao, S., Wu, W., Ge, T., Wei, F., & Ji, H. (2024b). Unleashing the emergent cognitive synergy in large language models: A task-solving agent through multi-persona self-collaboration. In *2024 conference of the North American chapter of the association for computational linguistics: Human language technologies (NAACL)* (pp. 257–279).

Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., Yogatama, D., Bosma, M., Zhou, D., Metzler, D., Chi, E. H., Hashimoto, T., Vinyals, O., Liang, P., Dean, J., & Fedus, W. (2022a). Emergent abilities of large language models. arXiv preprint arXiv:2206.07682.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q. V., & Zhou, D. (2022b). Chain-of-thought prompting elicits reasoning in large language models. In *Advances in neural information processing systems (neurIPS)* (pp. 24824–24837) .

Wei, Y., Huang, Q., Zhang, Y., & Kwok, J. (2023). Kicgpt: Large language model with knowledge in context for knowledge graph completion. In *Findings of the association for computational linguistics: EMNLP* (pp. 8667–8683).

Wu, H., Wang, Z., Wang, K., Omran, P. G., & Li, J. (2023). Rule learning over knowledge graphs: A review. *Transactions on Graph Data and Knowledge*, *1*(1), 7–1.

Yang, B., Yih, S. W.-t., He, X., Gao, J., & Deng, L. (2015). Embedding entities and relations for learning and inference in knowledge bases. In *International conference on learning representations (ICLR)*.

Yang, R., Zhu, J., Man, J., Liu, H., Fang, L., & Zhou, Y. (2025). Gs-kgc: A generative subgraph-based framework for knowledge graph completion with large language models. *Information Fusion*, *117*, 102868.

Yao, L., Mao, C., & Luo, Y. (2019). Kg-bert: Bert for knowledge graph completion. arXiv preprint arXiv:1909.03193.

Zhang, J.-C., Zain, A. M., Zhou, K.-Q., Chen, X., & Zhang, R.-M. (2024). A review of recommender systems based on knowledge graph embedding. *Expert Systems with Applications*, (p. 123876).

Zhang, Z., Cai, J., Zhang, Y., & Wang, J. (2020). Learning hierarchy-aware knowledge graph embeddings for link prediction. In *Proceedings of the AAAI conference on artificial intelligence (AAAI)* (pp. 3065–3072).

Zhang, Z., Zhang, A., Li, M., & Smola, A. (2022). Automatic chain of thought prompting in large language models. In *International conference on learning representations (ICLR)*.

Zhu, Y., Wang, X., Chen, J., Qiao, S., Ou, Y., Yao, Y., Deng, S., Chen, H., & Zhang, N. (2024). Llms for knowledge graph construction and reasoning: recent capabilities and future opportunities. *World Wide Web*, *27*(5), 58.