ENHANCING LARGE LANGUAGE MODELS WITH RELIABLE KNOWLEDGE GRAPHS

QINGGANG ZHANG

PhD

The Hong Kong Polytechnic University

2025

The Hong Kong Polytechnic University
Department of Computing

Enhancing Large Language Models with Reliable Knowledge Graphs

Qinggang Zhang

A thesis submitted in partial fulfillment of the requirements for
the degree of Doctor of Philosophy
March 2025

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgment has been made in the text.

Signature: _____

Name of Student: Qinggang Zhang

Abstract

Large Language Models (LLMs) have demonstrated remarkable capabilities in text generation and understanding, yet their reliance on implicit, unstructured knowledge often leads to factual inaccuracies and limited interpretability. Knowledge Graphs (KGs), with their structured, relational representations, offer a promising solution to ground LLMs in verified knowledge. However, their potential remains constrained by inherent noise, incompleteness, and the complexity of integrating their rigid structure with the flexible reasoning of LLMs. This thesis presents a systematic framework to address these limitations, advancing the reliability of KGs and their synergistic integration with LLMs through five interconnected contributions. This thesis addresses these challenges through a cohesive framework that enhances LLMs by refining and leveraging reliable KGs. First, we introduce contrastive error detection, a structure-based method to identify incorrect facts in KGs. This approach is extended by an attribute-aware framework that unifies structural and semantic signals for error correction. Next, we propose an inductive completion model that further refines KGs by completing the missing relationships in evolving KGs. Building on these refined KGs, KnowGPT integrates structured graph reasoning into LLMs through dynamic prompting, improving factual grounding. These contributions form a systematic pipeline (from error detection to LLM integration), demonstrating that reliable KGs significantly enhance the robustness, interpretability, and adaptability of LLMs.

Publications Arising from the Thesis

1. Qinggang Zhang, Junnan Dong, Keyu Duan, Xiao Huang*, Yezi Liu, Linchuan Xu, "Contrastive Knowledge Graph Error Detection", in *ACM International Conference on Information and Knowledge Management (CIKM)*, 2022.

2. Qinggang Zhang, Junnan Dong, Qiaoyu Tan, Xiao Huang*. "Integrating Entity Attributes for Error-Aware Knowledge Graph Embedding", in *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 2023.

3. Qinggang Zhang, Keyu Duan, Junnan Dong, Pai Zheng, Xiao Huang*, "Logical Reasoning with Relation Network for Inductive Knowledge Graph Completion", in *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 2024.

4. Qinggang Zhang†, Junnan Dong†, Hao Chen, Daochen Zha, Zailiang Yu, Xiao Huang*, "KnowGPT: Knowledge Graph based Prompting for Large Language Models", in *The Thirty-eighth Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2024.

Acknowledgments

I would like to begin by expressing my heartfelt gratitude to my supervisor, Prof. Xiao Huang, for his unwavering support, insightful guidance, and the freedom to explore areas of personal interest. His patience, thoughtful advice on personal and professional development, and valuable insights on career planning have been immensely beneficial to me. Additionally, I greatly admire his approach to tackling complex research problems, which has deeply influenced my own work.

I also wish to extend my thanks to my co-supervisor, Dr. Jiannong Cao, for his constructive advice on various aspects of my research. His support, as well as the resources he provided, were essential in shaping my academic journey. I am also deeply grateful to several senior researchers who have contributed significantly to my research. Dr. Hao Chen has offered invaluable suggestions, particularly in identifying real-world problems and improving my writing skills. My thanks also go to Dr. Qiaoyu Tan, Dr. Daochen Zha, Prof. Feiran Huang and Prof. Pai Zheng for their collaborative efforts and insightful contributions. Besides, I would like to express my warm appreciation to the members of the DEEP (Data Exploring and Extracting @ PolyU) Lab in the Department of Computing for their companionship and continuous support throughout these years.

Lastly, I want to extend my deepest thanks to my family and friends. Their unwavering encouragement and constant support have been crucial in helping me complete this journey. I am truly grateful for everything they have done for me.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Large language models (LLMs), such as the Clude [4] and GPT series [73], have demonstrated exceptional capabilities across diverse tasks, achieving breakthroughs in text comprehension [11], question answering [50], and content generation [21]. Despite their success, LLMs face persistent criticism for their limitations in knowledge-intensive tasks, particularly those requiring domain expertise [115]. Their application in specialized domains remains challenging due to three key factors: ❶ **Knowledge limitations**: LLMs possess broad but superficial knowledge in specialized fields, as their training data primarily consists of general-domain content, leading to gaps in domain-specific depth and alignment with current professional standards. ❷ **Reasoning complexity**: Specialized domains demand precise, multi-step reasoning governed by domain-specific rules and constraints. LLMs often struggle to maintain logical consistency and accuracy throughout extended reasoning chains, especially in technical or highly regulated contexts. ❸ **Context sensitivity**: Professional fields rely on nuanced, context-dependent interpretations where identical terms or concepts may carry different meanings based on specific scenarios. LLMs frequently fail to

grasp these subtleties, resulting in misinterpretations or overly generalized responses.

To adapt LLMs for specific domains, researchers have explored various approaches, which can be broadly classified into two categories:

Fine-tuning LLMs with Domain-specific Data. Fine-tuning pre-trained LLMs on specialized datasets enables them to better capture domain-specific vocabulary, terminology, and patterns [34, 53, 54, 31]. This approach has been successfully applied in areas such as recommendation [17, 123] and node classification [16, 40], enhancing the relevance and accuracy of generated responses [41]. Fine-tuned LLMs have demonstrated effectiveness across various domains. In healthcare, they have been leveraged for clinical note analysis [3], biomedical text mining [54], and medical dialogue [91]. Similarly, in the legal domain, they have proven useful for legal document classification [13], contract analysis [14], and legal judgment prediction [122].

Retrieval-augmented generation (RAG). RAG provides an effective way to tailor LLMs for specialized domains without modifying the model architecture or parameters [56]. Instead of embedding new knowledge through retraining, RAG dynamically retrieves relevant domain-specific information from external sources, enhancing response accuracy and reliability. A typical RAG system operates in three stages: knowledge preparation, retrieval, and integration. First, external textual data is segmented into manageable chunks and transformed into vector representations for efficient indexing. During retrieval, relevant chunks are identified based on keyword matching or vector similarity when a query is submitted. Finally, the retrieved information is combined with the original query to generate well-informed responses.

1.2 Motivation

Despite their success, RAG systems face significant challenges in practical applications due to the inconsistent quality of accessible data. Domain knowledge is frequently

distributed across diverse sources—ranging from textbooks and research articles to technical manuals and industry reports [57]—which may vary in quality, accuracy, and completeness, potentially leading to discrepancies in the retrieved information [124]. A promising strategy to mitigate these issues is to integrate Knowledge Graphs (KGs) with LLMs. KGs offer a structured representation of domain knowledge, built on well-defined ontologies that specify specialized terminologies, acronyms, and their interrelations within a field [58, 80, 116, 117, 118]. The extensive factual content contained in KGs can help anchor model responses in established facts and principles [39, 109, 74].

However, integrating KGs with LLMs is fraught with challenges. Existing KGs are far from perfect since they often suffer from incompleteness, noise (e.g., erroneous or conflicting triples), and rigidity (e.g., inability to generalize to unseen entities). Moreover, LLMs lack mechanisms to dynamically retrieve and reason over structured knowledge during generation, often treating retrieved facts as isolated snippets rather than interconnected evidence. To unlock their full potential, KGs must first be refined into reliable, dynamic repositories of knowledge and then seamlessly integrated into LLMs. This thesis tackles these dual challenges through a systematic framework that spans error detection, graph completion, and synergistic LLM-KG integration, ultimately advancing the robustness and interpretability of AI systems.

1.3 Research Objectives

The central aim of this thesis is to advance the integration of structured knowledge into LLMs by addressing critical challenges in KG reliability, completeness, and synergistic LLM-KG interaction. This goal is operationalized through three interconnected objectives:

- Error Detection in Knowledge Graphs. The first objective focuses on identifying and resolving inaccuracies in KGs caused by noise during construction or

updates. This involves developing methods to detect structural inconsistencies (e.g., conflicting triples violating logical constraints) and semantic mismatches (e.g., discrepancies between entity attributes and their relational context). By unifying structural and attribute-based analysis, the goal is to create a robust framework for error identification and resolution, ensuring KGs serve as trustworthy knowledge bases.

- Knowledge Graph Completion. The second objective addresses the incompleteness of KGs, particularly in dynamic environments where new entities and relationships emerge continuously. Traditional transductive models, which rely on fixed entity sets during training, are inadequate for such scenarios. This objective seeks to design an inductive completion model capable of inferring missing relationships for both existing and unseen entities by leveraging logical rules and relational patterns, thereby enabling KGs to evolve adaptively.

- Enhancing LLMs with Structured Knowledge. The third objective bridges the gap between structured KGs and unstructured LLM reasoning. While LLMs excel at text generation, they lack mechanisms to systematically ground outputs in verified knowledge. This objective involves designing a framework that dynamically retrieves and integrates relevant subgraphs into LLM workflows, structuring the model's reasoning process around relational paths derived from KGs to enhance factual consistency and interoperability.

1.4 Contributions

This thesis contributes five interconnected advancements that systematically address the limitations of KGs and LLMs while establishing a pipeline for their synergistic integration:

- Contrastive Knowledge Graph Error Detection. This thesis introduces a con-

trastive learning framework (CAGED) to identify erroneous triples by analyzing structural plausibility. By training a model to differentiate valid triples from synthetically generated corruptions, this approach detects inconsistencies through margin-based scoring, achieving state-of-the-art precision in structural error detection across benchmark KGs.

- Error-Aware Embedding with Attribute Integration. We extend structural error detection by incorporating entity attributes into a unified embedding space. This hybrid model (AEKE) jointly optimizes structural and semantic signals, enabling the identification of errors that manifest as contradictions between relational patterns and attribute metadata. The integration of attributes improves error correction accuracy significantly compared to structure-only baselines, particularly in complex scenarios requiring contextual understanding.

- Logical Rule-based KG Completion. This thesis proposes a neural-symbolic model (NORAN) that combines relational networks with first-order logic rules to infer missing relationships inductively. By decoupling logical rule application from entity-specific embeddings, the model generalizes to unseen entities while maintaining interpretability. Evaluations demonstrate superior performance in inductive settings, outperforming existing methods in inferring relationships for dynamically evolving KGs.

- KG Prompting for LLMs. We introduce a prompting architecture (KnowGPT) that bridges structured KGs and LLMs. By dynamically retrieving subgraphs relevant to a query and serializing them into natural language prompts, this framework guides LLMs to reason along verified relational paths. Empirical results show marked reductions in factual hallucinations, validating the utility of structured knowledge in grounding LLM outputs.

1.5 Overall Structure

This thesis is structured to reflect the logical progression from KG refinement to LLM integration. Following this introductory chapter, Chapter 2 reviews foundational concepts in knowledge graphs and large language models, highlighting key limitations in existing approaches to error detection, graph completion, and KG-enhanced LLMs. Chapters 3-6 present the five core research papers, each addressing a distinct component of the pipeline. Papers 1 [116] and 2 [117] focus on error detection, first through a structure-based contrastive approach and then by incorporating entity attributes for richer semantic analysis. Paper 3 [118] shifts to KG completion, introducing a logic-guided model that enables inductive reasoning over evolving graphs. Paper 4 [115] bridges the refined KGs with LLMs through KnowGPT, a prompting architecture that structures LLM inferences around retrieved subgraphs. The concluding chapter reflects on the implications of this work and outlines directions for future research.

Chapter 2

Literature Review

2.1 Knowledge Graphs

2.1.1 Knowledge Graph Definition

Knowledge Graphs (KGs) [7, 55, 36, 67] represent entities and their relationships as structured triples, offering a machine-interpretable format for encoding real-world knowledge. Early KGs, such as Freebase [7] and Wikidata [36], laid the groundwork for applications in semantic search, recommendation systems, and question answering. Modern KGs have expanded in scale and complexity, yet they remain constrained by three core challenges: incompleteness (missing entities or relationships), noise (erroneous or conflicting triples), and rigidity (inability to generalize to unseen entities or adapt dynamically). Traditional KG embedding models, such as TransE [9] and RotatE [84], map entities and relations to low-dimensional vectors but struggle to address these challenges, particularly in open-world scenarios where knowledge evolves continuously.

2.1.2 Knowledge Graph Refinement

KG refinement encompasses two interrelated tasks: error detection and knowledge completion. (i) Early error detection methods relied on rule-based heuristics [2] or statistical outlier detection in embedding spaces [63]. While effective for simple inconsistencies, these approaches lacked the contextual awareness to resolve complex errors involving semantic or temporal contradictions. Contrastive learning has proven effective under various error detection scenarios. Based on this paradigm, this thesis introduces a structure-aware contrastive framework (CAGED) [116] that distinguishes valid triples from corrupted ones by maximizing the margin between their confidence scores. This approach is extended with entity attribute (AEKE)[117] to unify structural and semantic error signals. (ii)For knowledge completion, transductive models like Graph Neural Networks (GNNs) assume a fixed entity set during training, rendering them ineffective for evolving KGs. Inductive approaches address this limitation by generalizing to unseen entities, often through meta-learning or rule-based reasoning [98, 92]. However, prior work either sacrifices interpretability (e.g., black-box neural models) or scalability (e.g., handcrafted logical rules). This thesis bridges this gap with a novel model (NORAN) [118] that mines logic rules within a trainable relational network, enabling inductive reasoning while maintaining transparency.

2.2 KG-enhanced LLM

2.2.1 LLM Hallucination

LLMs internalize knowledge implicitly through pretraining on vast text corpora, but their static knowledge base and lack of structured reasoning lead to factual inaccuracies and hallucinations [115]. Retrieval-Augmented Generation (RAG) mitigates this by grounding LLMs in external data, yet most RAG systems treat knowledge as

unstructured text, neglecting the relational structure of KGs. Early attempts to integrate KGs with LLMs focused on static embeddings or predefined prompts [53, 54, 31], limiting their ability to handle dynamic queries requiring multi-hop reasoning. Recent work explores dynamic subgraph retrieval and serialization, but these methods often lack systematic error handling or fail to scale to large KGs.

2.2.2 Knowledge Integration

Earlier studies adopted a heuristic way to inject knowledge from KGs into the LLMs during pre-training or fine-tuning. ERNIE [82] incorporates entity embeddings and aligns them with word embeddings in the pre-training phase, encouraging the model to better understand and reason over entities. UniKGQA [47] is the first approach to leverage LLMs to seamlessly integrate retrieval and reasoning within a unified framework. Another line of work focuses on retrieving relevant knowledge from KGs at inference time to augment the language model's context. Typically, K-BERT [62] uses an attention mechanism to select relevant triples from a KG based on the input context, which are then appended to the input sequence. More recently, KG prompting has been intensively studied for integrating factual knowledge into LLMs. KD-CoT [96] and KG-CoT [121] build upon the concept of chain-of-thought, guiding LLMs through a step-by-step reasoning process while enabling timely correction of erroneous reasoning. Their factuality and faithfulness are validated using an external knowledge graph. RoG [65] presents a planning-retrieval-reasoning framework that synergizes LLMs and KGs for more transparent and interpretable reasoning. Despite their effectiveness, existing models focus on designing methods for knowledge retrieval or generation. They directly feed the retrieved or generated knowledge into the LLM for reasoning, which does not necessarily provide effective guidance to the LLMs. This is because LLMs often struggle to distinguish between valid and redundant knowledge.

Chapter 3

Contrastive Knowledge Graph Error Detection

3.1 Introduction

With the increasing deployment of KG-driven applications such as conversational agents and recommender systems [93, 42], the need for reliable error detection in knowledge graphs (KGs) has become critical. KGs represent assertions as triples—i.e., *(head entity, relation, tail entity)*—offering a structured and scalable way to organize information. However, because many KGs are automatically extracted from web data using heuristic methods, they inevitably incorporate a significant amount of noise [8, 55, 36, 67]. For instance, the widely used NELL KG [12] achieves a precision of only 74%, implying roughly 0.6 million triples may be erroneous. Many existing approaches overlook these errors by assuming the correctness of all triples, which can lead to degraded performance in downstream tasks. Consequently, the development of effective KG error detection algorithms is imperative.

The task of KG error detection is complicated by the diversity and subtlety of error patterns, compounded by the scarcity of ground-truth labels—since obvious errors are

typically rectified during KG construction [75]. Prior work can be grouped into two main categories. The first, *rule-based* methods, identify errors as violations of a set of predefined rules [2, 33, 20, 29, 87]; however, these rules tend to be domain-specific and lack generalizability. The second category, *embedding-based* methods, often rely on simplistic negative sampling strategies, where a positive triple (h, r, t) is corrupted by randomly replacing h or t, to generate synthetic negative examples for training [76, 46, 30, 106, 68]. While useful, these strategies fail to capture the complex nature of real-world errors. For example, randomly transforming a valid triple like *(Newton, Nationality, England)* into *(Newton, Nationality, Google)* does not reflect the nuanced errors encountered in practice. Real errors may involve subtle mismatches—such as *(Bruce_Lee, place_of_birth, China)*—where the entities are contextually related yet incorrect. This discrepancy highlights the need for more advanced error simulation and detection mechanisms.

Contrastive learning has emerged as a powerful self-supervised technique in various tasks such as classification [113], link prediction [100], and recommendation [103]. By generating different views of the same data through transformations and then maximizing the agreement between these views [18, 37, 38, 85, 15], models can learn discriminative features without relying on manual labels. This approach has proven effective in computer vision for error detection, suggesting its potential applicability to KGs. However, applying contrastive learning to KG error detection introduces two primary challenges. First, constructing meaningful views for KGs is nontrivial; traditional graph augmentation methods (e.g., node dropping, edge perturbation, subgraph sampling) [112, 35] are tailored for network embeddings and may disrupt the delicate structure of erroneous triples. Since a KG is essentially a collection of triples, effective error detection demands augmentation strategies that preserve the inherent error patterns. Second, existing graph encoders do not differentiate between reliable and noisy triples during message passing, potentially diluting the learned representations. Thus, an encoder that is aware of and can mitigate the influence of

erroneous triples is needed.

To address these issues, we formally define the problem of KG error detection and explore two central questions: ❶ How can we generate meaningful, distinct views of a KG to facilitate effective contrastive learning? ❷ How can we design an error-aware KG encoder that enhances the detection of noisy triples? In response, we propose the ContrAstive knowledge Graph Error Detection (CAGED) framework. Our key contributions include:

- Introducing CAGED, which integrates contrastive learning with KG embeddings to improve error detection.
- Developing a novel KG augmentation technique that produces triple-level views while preserving the structure of potential errors.
- Proposing an error-aware graph encoder, EaGNN, that employs a gated attention mechanism to suppress the influence of noisy triples during representation learning.
- Demonstrating the effectiveness of CAGED through extensive experiments on three real-world KGs, where it outperforms existing error detection methods.

3.2 Contrastive Knowledge Graph Error Detection (CAGED)

In conventional KG representation learning, a knowledge graph is typically treated as a heterogeneous graph where entities serve as nodes and relations as semantic links. However, this formulation often struggles to capture the intricate interdependencies among triples. In this work, we introduce ContrAstive knowledge Graph

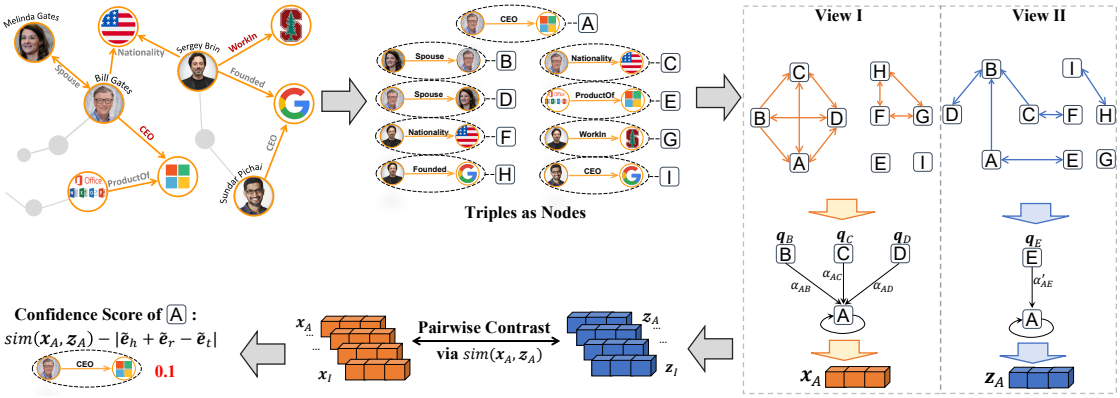


Figure 3.1: Two separate augmentation operators are applied to the original KG, generating two triple graphs as congruent views, i.e., View I and View II. After training, we estimate the confidence score by measuring the consistency of triple representations across multi-views, i.e., $\boldsymbol{x}_A$ and $\boldsymbol{z}_A$, and the self-consistency within the triple, i.e., $(\tilde{\boldsymbol{e}}_h, \tilde{\boldsymbol{e}}_r, \tilde{\boldsymbol{e}}_t)$.

Error Detection (CAGED), a framework designed to pinpoint inaccuracies in large-scale KGs. The key idea is to recast the KG into multiple hyper-views—specifically, triple graphs—by regarding each relational triple as a node, and then to evaluate the reliability of each triple by comparing its representations across these views. The intuition is that contrastive learning can extract rich semantic features for normal triples, whereas anomalous triples, lacking these features, will have representations that diverge in the latent space. Thus, the degree to which a triple's representations from different views converge serves as a reliable signal for error detection.

As depicted in Figure 3.1, CAGED comprises three main components: KG augmentation, an error-aware encoder, and joint confidence estimation. First, we propose an innovative KG augmentation technique that produces two triple-level graphs by treating each relational triple as a node, thereby creating two congruent views (View I and View II). Next, we introduce a tailored error-aware knowledge graph neural network (EaGNN) that suppresses the influence of erroneous triples during representation learning. Finally, to obtain discriminative multi-view representations, we jointly optimize a translation-based KG embedding loss and a contrastive learning

loss, and estimate each triple's confidence by leveraging both the consistency across views and its internal coherence.

3.2.1 Knowledge Graph Augmentation

Data augmentation plays a pivotal role in contrastive learning [52]. Unlike image data—where standard techniques such as rotation, cropping, or distortion can easily yield diverse views [125, 35]—constructing views for KGs is challenging due to their unique structure. Traditional graph contrastive learning methods typically use augmentations like node dropping, edge perturbation, subgraph sampling, or matrix diffusion [112], which have shown promising results in tasks such as node classification, graph classification, and link prediction.

However, these methods are not ideal for KG error detection since a KG is essentially a collection of triples, and error detection entails identifying spurious triples. Approaches that emphasize entity or graph-level contrasts do not adequately capture the nuances required at the triple level. Moreover, while these augmentations can improve the robustness of representations, they may inadvertently distort the underlying error distribution or even introduce additional noise, thereby complicating the detection task.

Definition 1. ***Linking Pattern.*** *Consider two triples, $T_1 = (h_1, r_1, t_1)$ and $T_2 = (h_2, r_2, t_2)$, that share entities. They can be linked in two distinct ways: (i) by sharing a head entity (i.e., $h_1 = h_2$ or $h_1 = t_2$), and (ii) by sharing a tail entity (i.e., $t_1 = h_2$ or $t_1 = t_2$). Based on this observation, we construct two triple graphs using these linking patterns, as they capture different semantic relationships.*

Our augmentation strategy generates two triple graphs, denoted by $\mathcal{T}$ and $\mathcal{T}'$, by treating each relational triple as a node and connecting them according to the linking patterns in Definition 1. Concretely, $\mathcal{T}$ reflects connections based on shared head

entities, whereas $\mathcal{T}'$ is built on shared tail entities. Given that triples sharing an entity are generally semantically related, normal triples typically have ample neighbors in $\mathcal{T}$ (View I) that can mirror the semantics captured in $\mathcal{T}'$ (View II). Thus, assessing the consistency between a triple's representations in these two views provides a reliable measure of its trustworthiness.

3.2.2 Error-aware Encoder – EaGNN

Erroneous triples can severely impair the quality of learned representations by propagating misleading information during message passing. To counteract this, we propose a custom error-aware graph neural network (EaGNN) that captures both the semantic and structural properties of triples while mitigating the influence of noise. EaGNN integrates a local information modeling layer with a global error-aware attention mechanism to generate robust KG embeddings.

Local Information Modeling Layer. Transforming the original KG into a triple graph may result in a loss of the inherent sequential structure (i.e., the pattern $h \rightarrow r \rightarrow t$). To preserve this local information, we initialize the embeddings of entities and relations randomly and employ a set of Bi-LSTM units to learn the internal structure of each triple. For a given triple (h, r, t), the process is formalized as:

$$\tilde{\mathbf{e}}_h, \tilde{\mathbf{e}}_r, \tilde{\mathbf{e}}_t = \text{Bi-LSTM}(\mathbf{e}_h, \mathbf{e}_r, \mathbf{e}_t), \tag{3.1}$$

$$\mathbf{q}_i = [\tilde{\mathbf{e}}_h; \tilde{\mathbf{e}}_r; \tilde{\mathbf{e}}_t]. \tag{3.2}$$

The derived triple embedding $\mathbf{q}_i$ effectively encapsulates the relational structure within the triple and serves as the initial representation in the constructed triple graphs.

Global Error-aware Attention Layer. In addition to local features, the global context provided by neighboring triples is essential for evaluating a triple's reliability.

Standard KG neural networks often assign uniform attention to all neighbors, which can cause noisy information from erroneous triples to contaminate the final representation. To address this, we propose a novel attention mechanism that selectively attenuates the contribution of dubious neighbors.

For an anchor triple $\mathbf{q}_i \in \mathbb{R}^d$ in View I, we update its representation by aggregating information from its neighboring triples $\{\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_m\}$. The attention weight between the anchor triple and a neighbor j is computed as:

$$\hat{\alpha}_{ij} = \mathcal{A}(\mathbf{W}\mathbf{q}_i, \mathbf{W}\mathbf{q}_j), \tag{3.3}$$

where $\mathbf{W} \in \mathbb{R}^{n \times d}$ is a learnable projection matrix, and $\mathcal{A}$ denotes an attention function. We then normalize these weights using softmax:

$$\overline{\alpha}_{ij} = \frac{\exp(\hat{\alpha}_{ij})}{\sum_{k=1}^{m} \exp(\hat{\alpha}_{ik})}. \tag{3.4}$$

To suppress the effect of potential outliers, we introduce a threshold $\mu \in \mathbb{R}$:

$$\alpha_{ij} = \begin{cases} \overline{\alpha}_{ij}, & \text{if } \overline{\alpha}_{ij} > \mu, \\ \\ 0, & \text{otherwise.} \end{cases} \tag{3.5}$$

The final representation for the anchor triple in View I is then obtained by:

$$\mathbf{x}_i = \sigma\left(\sum_{j=1}^{m} \alpha_{ij}\, \mathbf{W}\, \mathbf{q}_j \right), \tag{3.6}$$

and similarly, for an anchor triple in View II:

$$\mathbf{z}_i = \sigma\left(\sum_{j=1}^{m} \alpha'_{ij}\, \mathbf{W}\, \mathbf{q}_j \right). \tag{3.7}$$

3.2.3 Joint Confidence Estimation

To ensure that the model learns rich and discriminative representations across views, we jointly optimize two loss functions: a KG embedding loss and a contrastive learning loss.

KG Embedding Loss. At the level of individual triples, we leverage the translation principle—i.e., $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$—to assess triple plausibility. We use the squared Euclidean distance as an energy function:

$$E(h, r, t) = \|e_h + e_r - e_t\|_2 , \tag{3.8}$$

and define the KG embedding loss as:

$$\mathcal{L}_{kge} = \sum_{(h,r,t)\in\mathcal{G}} \sum_{(\hat{h},\hat{r},\hat{t})\in\hat{\mathcal{G}}} \max\left(0, \gamma + E(h, r, t) - E(\hat{h}, \hat{r}, \hat{t})\right), \tag{3.9}$$

where $\gamma > 0$ is a margin hyperparameter, $\mathcal{G}$ denotes the set of positive triples, and $\hat{\mathcal{G}}$ is constructed by corrupting the head or tail of each triple:

$$\hat{\mathcal{G}} = \{(\hat{h}, r, t) \mid \hat{h} \in \mathcal{G}\} \cup \{(h, r, \hat{t}) \mid \hat{t} \in \mathcal{G}\}. \tag{3.10}$$

Contrastive Loss. While the KG embedding loss focuses on local structural details, contrastive learning enables the model to capture global semantic consistency across views. We randomly sample a minibatch of n triples from the KG, resulting in $2n$ representations: $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ from View I and $\{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n\}$ from View II. For each triple i, the pair $(\mathbf{x}_i, \mathbf{z}_i)$ is treated as positive, while the remaining representations serve as negatives. The contrastive loss is formulated as:

$$\mathcal{L}_{con}(\mathbf{x}_i, \mathbf{z}_i) = -\log \frac{\exp\left(\text{sim}(\mathbf{x}_i, \mathbf{z}_i)/\tau\right)}{\sum_{j\in\{1,2,\ldots,n\}\setminus\{i\}} \exp\left(\text{sim}(\mathbf{x}_i, \mathbf{z}_j)/\tau\right)}, \tag{3.11}$$

where $\text{sim}(\mathbf{x}_i, \mathbf{z}_i)$ denotes the cosine similarity between the two representations and τ is a temperature parameter. The overall contrastive loss is computed over all triples in the minibatch.

3.3 Experiments

To assess the performance of our proposed CAGED framework, we conduct extensive experiments on three real-world knowledge graphs. Our evaluation is structured around the following research questions:

Table 3.1: The statistical information of the datasets.

Dataset	Entities	Relations	Triples	Mean in-degree
FB15K	14,541	237	310,116	18.71
WN18RR	40,943	11	93,003	2.12
NELL-995	75,492	200	154,213	1.98

- **Q1 (Effectiveness):** How does CAGED compare to state-of-the-art methods for KG error detection?
- **Q2 (Ablation Study):** What are the contributions of each individual component within CAGED?
- **Q3 (Parameter Analysis):** How do variations in hyperparameters impact the performance of CAGED?

3.3.1 Experimental Settings

This section details the experimental configuration, including datasets, baseline methods, evaluation metrics, and implementation specifics.

Datasets. Following previous work [104, 46], we evaluate our approach on three datasets derived from established benchmarks: FB15k, WN18RR, and NELL-995. Each dataset is constructed by injecting noise at levels of 5%, 10%, and 15% of the total triples. Table 3.1 summarizes their statistics. In this paper, we detect errors from three categories, including factual errors that include incorrect entity relationships, coverage errors that arise from missing entities or relations, and logical inconsistencies that violate formal rules, like circular dependencies or conflicting facts.

FB15K originates from the Freebase Knowledge Base and is augmented with textual relations extracted from ClueWeb12 and annotated in Freebase.

WN18RR is a refined subset of WordNet designed to eliminate test leakage by removing inverse relations; it comprises 11 relation types and features a relatively simpler structure.

NELL-995 is obtained from the 995th iteration of the NELL system, a large-scale and evolving knowledge base. Triples lacking reasoning values are removed prior to selecting the top-200 unique relations.

Baseline Methods. We compare CAGED with two groups of baseline approaches. The first group consists of KG embedding techniques, including **TransE** [10], **DistMult** [107], and **ComplEx** [90]. In these methods, the confidence score for each triple is computed from the embedding-based score (e.g., $\|\boldsymbol{e}_h+\boldsymbol{e}_r-\boldsymbol{e}_t\|_2$ in TransE). The second group comprises specialized KG error detection methods such as **CKRL** [104], **KGTtm** [46], and **KGIst** [5]. CKRL extends TransE by incorporating all paths between entities, KGTtm further integrates the overall graph structure, and KGIst employs an unsupervised strategy to learn soft rules for error identification.

Evaluation Metrics. We employ ranking-based metrics to assess performance. Triples are sorted in ascending order according to their confidence scores, with lower scores indicating a higher likelihood of error. Specifically, we use:

- **Precision@K**: The proportion of false triples within the top K triples.
- **Recall@K**: The fraction of all erroneous triples identified in the top K.

Implementation Details. All methods are implemented using PyTorch, and we utilize publicly available code for baseline models. Experiments are performed on an Nvidia RTX 3090 GPU. We use the Adam optimizer with a fixed batch size of 256, an initial learning rate of 0.01, and an embedding dimension of 100 for all models. A grid search is conducted to tune hyperparameters: the attention threshold μ (ranging

Table 3.2: Error detection results of Precision@K and Recall@K based on the three datasets with anomaly ratio = 5%.

		FB15K					WN18RR					NELL-995				
		K=1%	K=2%	K=3%	K=4%	K=5%[a]	K=1%	K=2%	K=3%	K=4%	K=5%[a]	K=1%	K=2%	K=3%	K=4%	K=5%[a]
Precision@K	TransE	0.756	0.674	0.605	0.546	0.488	0.581	0.488	0.371	0.345	0.331	0.659	0.550	0.476	0.423	0.383
	ComplEx	0.718	0.651	0.590	0.534	0.485	0.518	0.444	0.382	0.341	0.307	0.627	0.538	0.472	0.427	0.378
	DistMult	0.709	0.646	0.582	0.529	0.483	0.574	0.451	0.390	0.349	0.322	0.630	0.553	0.493	0.446	0.408
	CKRL	0.789	0.736	0.684	0.630	0.574	0.675	0.526	0.436	0.389	0.349	0.735	0.642	0.559	0.498	0.450
	KGTtm	0.815	0.767	0.713	0.612	0.579	0.770	0.628	0.516	0.444	0.396	0.808	0.691	0.602	0.535	0.481
	KGIst	0.825	0.754	0.703	0.617	0.569	0.747	0.599	0.476	0.407	0.379	0.782	0.678	0.584	0.528	0.485
	CAGED	**0.852**	**0.796**	**0.735**	**0.665**	**0.595**	**0.826**	**0.726**	**0.632**	**0.541**	**0.469**	**0.850**	**0.736**	**0.644**	**0.573**	**0.516**
Recall@K	TransE	0.151	0.270	0.363	0.437	0.488	0.116	0.195	0.223	0.276	0.331	0.132	0.220	0.285	0.338	0.383
	ComplEx	0.143	0.260	0.354	0.427	0.485	0.103	0.177	0.229	0.273	0.307	0.125	0.215	0.283	0.341	0.378
	DistMult	0.141	0.258	0.349	0.423	0.483	0.114	0.180	0.234	0.279	0.322	0.126	0.221	0.295	0.357	0.408
	CKRL	0.158	0.294	0.410	0.504	0.574	0.135	0.210	0.261	0.311	0.349	0.147	0.256	0.335	0.398	0.450
	KGTtm	0.163	0.307	0.428	0.490	0.579	0.154	0.251	0.309	0.355	0.396	0.161	0.276	0.361	0.428	0.481
	KGIst	0.165	0.302	0.422	0.494	0.569	0.149	0.240	0.285	0.325	0.379	0.156	0.271	0.350	0.422	0.485
	CAGED	**0.171**	**0.318**	**0.441**	**0.532**	**0.595**	**0.165**	**0.290**	**0.379**	**0.433**	**0.469**	**0.170**	**0.294**	**0.386**	**0.458**	**0.516**

from 0.001 to 0.2), the margin parameter γ (ranging from 0 to 1), and the trade-off coefficient λ (ranging from 0.001 to 1000). The number of neighbors is determined by averaging the neighbor count across all triples in each dataset. To reduce variability, results are averaged over ten runs using a fixed random seed.

3.3.2 Effectiveness (Q1)

To address Q1, we evaluate all methods on the three datasets with a noise level of 5%. Table 3.3 summarizes the results (similar trends are observed for other noise levels, as shown in Table 3.3). Our key observations are:

1. KG error detection methods (including CAGED, CKRL, KGTtm, and KGIst) outperform standard KG embedding techniques (e.g., TransE, ComplEx, and DistMult) because the latter do not explicitly account for erroneous triples.

2. CAGED consistently achieves the highest performance in terms of both recall and precision. For instance, when evaluated at K corresponding to 5% and with a 5% anomaly ratio, CAGED improves over the second-best method by approx-

Table 3.3: Error detection results on NELL-995 with different anomaly ratios.

	Ratio	5%			10%			15%		
	K	K=5%[a]	K=10%	K=15%	K=5%	K=10%[a]	K=15%	K=5%	K=10%	K=15%[a]
Precision@K	TransE	0.383	0.285	0.225	0.626	0.499	0.407	0.702	0.621	0.535
	ComplEx	0.378	0.289	0.231	0.614	0.507	0.402	0.696	0.589	0.528
	DistMult	0.408	0.298	0.227	0.633	0.510	0.414	0.718	0.618	0.548
	CKRL	0.450	0.306	0.236	0.679	0.524	0.421	0.745	0.646	0.560
	KGTtm	0.481	0.320	0.242	0.713	0.527	0.437	0.788	0.673	0.576
	KGIst	0.485	0.317	0.244	0.748	0.552	0.440	0.791	0.663	0.569
	CAGED	**0.516**	**0.325**	**0.251**	**0.799**	**0.585**	**0.458**	**0.823**	**0.729**	**0.599**
Recall@K	TransE	0.383	0.57	0.675	0.313	0.499	0.612	0.234	0.414	0.535
	ComplEx	0.378	0.578	0.693	0.307	0.507	0.603	0.232	0.393	0.528
	DistMult	0.408	0.596	0.681	0.317	0.510	0.621	0.239	0.412	0.548
	CKRL	0.450	0.612	0.708	0.340	0.524	0.632	0.248	0.431	0.560
	KGTtm	0.481	0.640	0.726	0.357	0.527	0.656	0.263	0.449	0.576
	KGIst	0.485	0.634	0.732	0.374	0.552	0.660	0.264	0.442	0.569
	CAGED	**0.516**	**0.650**	**0.753**	**0.400**	**0.585**	**0.687**	**0.274**	**0.486**	**0.599**

[a] Note that when K equals anomaly ratio, *Precision*@K and *Recall*@K have the same value.

imately 1.6%, 7.3%, and 3.1% on FB15K, WN18RR, and NELL-995, respectively. This improvement is attributed to the multi-view contrastive learning strategy that effectively captures both intra-triple and cross-view features.

3. The superiority of CAGED is particularly pronounced when focusing on the top-ranked erroneous triples (i.e., for smaller K values).

Furthermore, experiments on NELL-995 with noise levels of 5%, 10%, and 15% (see Table 3.3) confirm the robustness and stability of our model under varying conditions.

3.3.3 Ablation Study (Q2)

To investigate Q2, we conduct an ablation study on the NELL-995 dataset by evaluating several variants of CAGED (see Table 3.4). Due to space limitations, we only

Table 3.4: Four pairs of variants of CAGED.

	Original component	Replacement
Var_DN	KG augmentation	Node dropping
Var_EP	KG augmentation	Edge perturbation
Var_Concat	Bi-LSTM units	Only concatenation
Var_LSTM	Bi-LSTM units	LSTM units
Var_GCN	EaGNN	R-GCN
Var_GAT	EaGNN	KGAT
Var_Local	Joint optimization	Only negative sampling
Var_Global	Joint optimization	Only contrastive learning

report results for NELL-995, as similar trends are observed for FB15K and WN18RR.

Component 1: KG Augmentation. We compare our tailored KG augmentation strategy with two generic augmentation methods: node dropping (Var_DN) and edge perturbation (Var_EP). As illustrated in Table 3.5, both Var_DN and Var_EP underperform, even relative to some KG embedding baselines like TransE. This suggests that applying standard augmentations can disrupt the triple structure and introduce additional noise, thereby impairing error detection. The significant performance gap highlights the importance of our specialized augmentation approach, which preserves the distribution of potential errors while generating diverse views.

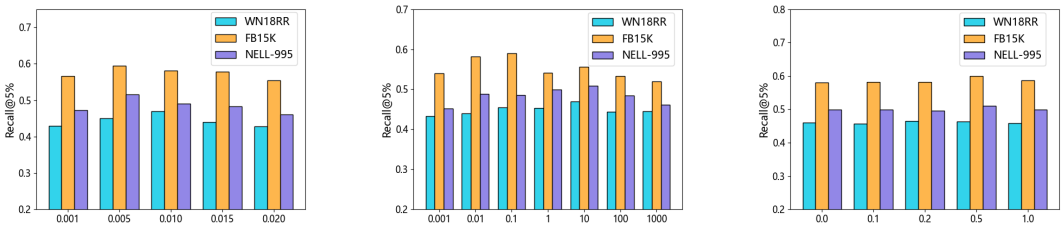
Component 2: EaGNN Encoder. To validate the effectiveness of our EaGNN encoder, we substitute it with leading KG neural networks such as R-GCN [78] (Var_GCN) and KGAT [71] (Var_GAT). Results in Table 3.5 indicate that, although Var_GAT performs better than Var_GCN, both still lag behind the full CAGED model. This outcome underscores that traditional KGNNs, which assume all triples are correct, are less capable of filtering out noise. Additionally, we test variants that

Table 3.5: The comparisons of CAGED and its variants on NELL-995 with ratio of errors equals 5%.

	Precision@K					*Recall@K*				
Top@K	1%	2%	3%	4%	5%	1%	2%	3%	4%	5%
CAGED	.850	.736	.644	.573	.516	.170	.294	.386	.458	.516
Var_DN	.613	.490	.412	.358	.325	.122	.196	.247	.286	.325
Var_EP	.593	.500	.416	.354	.313	.118	.200	.249	.283	.313
Var_Concat	.767	.648	.532	.468	.440	.153	.259	.319	.374	.440
Var_LSTM	.797	.662	.564	.500	.458	.159	.264	.338	.400	.458
Var_GCN	.760	.623	.529	.464	.414	.152	.249	.317	.371	.414
Var_GAT	.782	.629	.551	.471	.426	.156	.251	.330	.377	.426
Var_Local	.724	.603	.504	.433	.383	.144	.241	.302	.346	.383
Var_Global	.783	.652	.560	.492	.443	.156	.261	.336	.393	.443

modify the local structure modeling: Var_LSTM (using LSTM instead of Bi-LSTM) and Var_Concat (direct concatenation of entity, relation, and tail embeddings). The drop in performance for Var_Concat—and to a lesser extent for Var_LSTM—demonstrates the critical role of capturing local sequential structure via Bi-LSTM units.

Component 3: Joint Estimation. We also assess the impact of our joint optimization strategy by comparing two variants: Var_Local, which relies solely on the translation-based KG embedding loss (i.e., using negative sampling), and Var_Global, which employs only the contrastive learning loss. As shown in Table 3.5, both variants are outperformed by the integrated model, with Var_Local performing particularly poorly, even when compared to simple baselines like DistMult. These findings confirm that combining local structural features with cross-view consistency is essential for robust error detection.



(a) Impact of hyper-parameter μ

(b) Hyper-parameter λ

(c) Hyper-parameter γ

Figure 3.2: Impact of hyperparameters on the three datasets.

3.3.4 Parameter Analysis (Q3)

We further explore the influence of key hyperparameters on model performance, with the results summarized in Figure 3.2.

The attention threshold μ, regulates the selectivity of the attention mechanism. A larger μ limits attention to a small set of highly related neighbors, whereas a smaller μ allows a broader neighborhood to influence the representation. As shown in Figure 3.2(a), optimal performance is achieved when μ is approximately 0.005 for FB15K and NELL-995, and 0.01 for WN18RR. Decreasing μ further introduces excessive noise, leading to performance degradation.

The coefficient λ, which balances the contribution of cross-view inconsistency and intra-triple coherence, is varied from 10^{-3} to 10^{3}. Figure 3.2(b) shows that WN18RR and NELL-995 obtain their best results around $\lambda = 10$, while FB15K performs optimally when λ is below 1 (around 0.1). This difference is likely due to the denser structure of FB15K, which allows the cross-view signal to play a more significant role.

Finally, the margin parameter γ is varied from 0.1 to 1.0. As illustrated in Figure 3.2(c), the detection performance remains relatively stable across this range, suggesting that the joint training strategy helps prevent the model from converging to suboptimal solutions.

3.4 Summary

Traditional KG error detection methods typically rely on synthetically generated false triples—created by randomly replacing head or tail entities—which fail to capture the nuanced errors found in practice. In contrast, our CAGED framework leverages contrastive learning to generate multi-view representations centered on triples, allowing the model to discern subtle inconsistencies that indicate errors. Experimental results on three real-world knowledge graphs demonstrate that CAGED outperforms existing state-of-the-art methods across various evaluation metrics and noise levels.

Chapter 4

Integrating Entity Attributes for Error-Aware Reasoning

4.1 Introduction

Knowledge graphs (KGs) consolidate vast amounts of relational data as triples, i.e., *(head entity, relation, tail entity)*. They appear in both general-purpose settings (e.g., YAGO [67], DBpedia [55]) and in specialized domains such as biomedicine or agriculture. KGs serve as the backbone for many AI applications, including recommender systems enhanced by KG information [94] and conversational agents powered by structured knowledge [43].

A prominent research direction involves KG embedding, where entities are represented as continuous vectors and relations as operations (e.g., translations or projections) in a shared latent space. This approach enables efficient inference via simple numerical computations [9, 84, 33]. Despite the extensive development of embedding techniques, the adverse impact of noisy or erroneous triples is frequently neglected. Manual curation is infeasible at scale, and most real-world KGs are automatically extracted from web corpora using heuristic methods [7, 55, 36]. Consequently, a significant

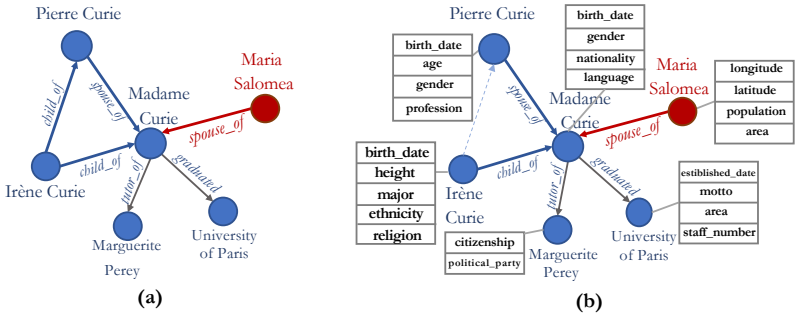


Figure 4.1: Running examples of complex errors in real-world KGs. (a) presents a KG error with mismatched head/tail entities and relations. This error can be detected by reasoning over neighboring triples. But, real-world KGs are often incomplete and noisy. (b) demonstrates a further difficult case with some key links missing. In this case, the rich semantics in entity attribute types can facilitate the detection of errors.

fraction of the triples are noisy. For example, the NELL KG [70] contains about 2.4 million triples with an estimated accuracy of 74%, indicating roughly 0.6 million erroneous triples [46]. Such inaccuracies can severely degrade downstream application performance, underscoring the need for error-aware KG embedding methods.

Developing robust, error-aware embeddings is challenging due to the unknown and diverse nature of KG errors. Recent efforts have explored mitigating noise by assigning reliability scores to triples [79, 72]. Early works such as Vault [26] estimated triple reliability via prior models, while methods like CKRL [104] and NoiGAN [19] further refined this idea by leveraging internal structural cues. However, solely relying on graph topology may not suffice; for example, a triple like {London, is_larger_than, Washington} might be ambiguous without additional context to resolve which "London" is intended.

Many KGs are also accompanied by rich attribute data that describe entity properties. For instance, as illustrated in Figure 4.1, an entity such as *MadameCurie* may be associated with attributes like {birth_date, gender, nationality, language}, highlighting its identity as a person, whereas an entity like *MariaSalomea* might include

{latitude, longitude, population, area}, reflecting its geographical nature. The interplay between an entity's attributes and the KG structure provides valuable signals for error detection. Typically, entities with similar attribute profiles are connected via specific relations; for example, those with geographical attributes are often linked by *live_in* or *born_in*, while entities with literary characteristics (e.g., {pen_name, writing_style}) tend to be connected by *author_of*. Thus, if a triple connects entities whose attributes are inconsistent with the stated relation (e.g., {MadameCurie, spouse_of, MariaSalomea} where a person is linked to a location), it likely indicates an error.

Nonetheless, integrating attribute information into error-aware KG embedding poses two key challenges. First, entity attributes are highly heterogeneous—varying in type, number, and context—so that a uniform encoder is needed to fuse diverse attribute information. An entity may exhibit different attribute sets in different roles (e.g., a scientist versus a writer). Second, aligning the semantics extracted from attributes with the KG's structural information is nontrivial, as these components inherently differ in characteristics. While some approaches directly merge attribute-derived features with entity embeddings, they often fail to capture the nuanced correlations required for effectively filtering out erroneous triples.

This work seeks to answer the following research questions: ❶ How can entities, relations, and attributes be jointly embedded into a unified vector space? ❷ How can we design an effective detector to compute a confidence score for each triple based on the learned features? ❸ How can these confidence scores be utilized to improve error-aware KG embeddings? To address these challenges, we propose a novel framework called **AEKE** (Attributed Error-aware Knowledge Embedding) [117]. The core idea of AEKE is to exploit the correlation between KG structure and entity attributes to emphasize reliable triples during embedding. Specifically, we construct two triple-level hypergraphs—the *relational hypergraph* and the *attribute hypergraph*—to capture, respectively, the structural topology of the KG and the semantics conveyed by entity attributes. A contrastive learning framework is then employed to learn com-

plementary representations from these two views. By jointly analyzing three types of anomaly signals—(i) intra-triple self-contradiction, (ii) cross-triple consistency, and (iii) the alignment between attribute information and graph structure—our method computes a confidence score for each triple. These scores are further used to adaptively adjust both the aggregation weights in the contrastive learning process and the margin loss in KG embedding, thereby reducing the influence of noisy triples.

The primary contributions of this work are as follows:

- We introduce AEKE, a novel KG embedding framework that leverages entity attributes to enable error-aware learning.
- We design a multi-view contrastive learning strategy that uses attribute information as a complementary view to guide KG representation learning.
- We propose a mechanism for computing triple confidence scores by integrating signals from intra-triple, cross-triple, and attribute-structure consistency.
- We employ these confidence scores to adaptively update the aggregation in multi-view learning and the margin loss in KG embedding, thereby mitigating the impact of erroneous triples.

4.2 Error-Aware Knowledge Graph Modeling

We present AEKE, a framework designed to learn error-aware knowledge graph (KG) embeddings by incorporating entity attributes. As shown in Fig. 4.2, AEKE is comprised of three main components: ❶ the KG representation learning model, which integrates a confidence score $C(h, r, t)$ into traditional KG embedding models to mitigate the impact of noise on embedding vectors; ❷ the triple confidence learning module, which calculates a confidence score to assess the correctness and significance of

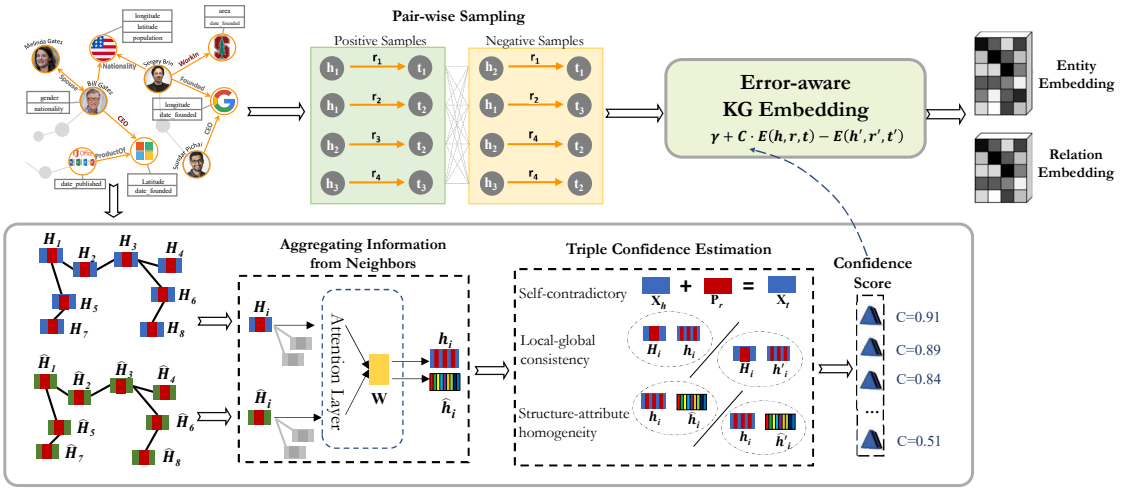


Figure 4.2: We perform a relation-induced construction process to build the *relational hypergraph*, and construct the *attribute hypergraph* based on entity attributes. These two hypergraphs can be regarded as congruent views of the target KG. A contrastive learning framework is used to learn the representation of each instance from these two different views. Meanwhile, a triple confidence estimation module is designed to calculate the confidence score of each triple by considering: self-contradictory within the triple; local-global consistency in graph structure; and structure-attribute homogeneity. Under the joint adaptive training scheme, we leverage confidence scores to adaptively update the weighted aggregation in contrastive learning and margin loss in KG embedding, such that potential errors would contribute little to KG learning.

each triple based on both internal structural information and external heterogeneous attribute data; ❸ a joint adaptive training scheme, which optimizes both the KG representation learning and the confidence learning models in an end-to-end manner.

4.2.1 Knowledge Graph Representation Learning

Knowledge graphs are effective tools for storing and handling structured data on real-world entities and facts, making them essential in many knowledge-driven appli-

cations. However, real-world KGs are often riddled with errors due to noisy sources and imperfect extraction methods during construction. Most existing KG embedding techniques assume all triples are correct, which results in the overfitting of noisy information into embeddings, causing significant performance degradation in downstream tasks.

To address this issue, we introduce the concept of a confidence score to distinguish between noisy and accurate triples. The confidence value ranges from $[0, 1]$, where values closer to 0 indicate a higher likelihood of error. By incorporating confidence, we propose a new error-aware objective function to filter out noisy triples from the embedding model's learning process:

$$L_{emb} = \sum_{(h,r,t)\in\mathcal{G}} \sum_{(h',r',t')\in\mathcal{G}'} \max(0, \gamma + C(h,r,t) \cdot E(h,r,t) - E(h',r',t')), \tag{4.1}$$

where $E(h,r,t) = \|e_h + e_r - e_t\|_2$ is the energy score based on the translational assumption. $\gamma > 0$ is the margin hyperparameter, and $\mathcal{G}$ represents the set of positive triples. The triple confidence $C(h,r,t)$ helps the model focus more on convincing triples, thus improving embedding learning. For negative sample generation, since explicit negative triples are not available, we follow a corruption strategy:

$$\begin{aligned}\mathcal{G}' = &\{(h',r,t) \mid h' \in \mathcal{E}\} \cup \{(h,r,t') \mid t' \in \mathcal{E}\} \\ &\cup \{(h,r',t) \mid r' \in \mathcal{R}\}, \quad (h,r,t) \in \mathcal{G}.\end{aligned} \tag{4.2}$$

This procedure replaces one entity or relation in a positive triple with another randomly chosen entity or relation from the entire set, and any resulting triples already in $\mathcal{G}$ are discarded to ensure the correctness of the negative samples.

4.2.2 Triple Confidence Learning with Entity Attributes

To enhance the robustness of our model, we learn a confidence score $C(h,r,t)$ for each triple, which quantifies its correctness by leveraging both graph structure and

entity attributes. Real-world KGs are often enriched with entity attributes that can provide valuable semantic information. These attributes are highly correlated with the relations in the KG and can help identify potential errors. We observe three key relationships:

- In the triple, relations can be viewed as translations acting on the low-dimensional embeddings of the entities. The better a triple aligns with the translation assumption, i.e., $\mathbf{h} + \mathbf{r} \simeq \mathbf{t}$, the more likely it is to be correct.

- In the graph, connected triples that share the same entity are typically semantically relevant. Intuitively, a KG can be viewed as a social group, where each triple is an individual. The acknowledgment from neighboring triples reflects how well a triple fits within the broader structure.

- Entity attributes are often closely related to the graph structure. Entities with relevant semantic attributes are usually connected by specific relations. A mismatch between an entity's attributes and its related triples suggests a higher likelihood of error.

To incorporate these insights, we propose a multi-view learning framework that models the structural information of the KG and its attributes. We use two triple-level hypergraphs—*relational hypergraph* and *attribute hypergraph*—to capture, respectively, the topological structure of the KG and the semantic information embedded in entity attributes. These hypergraphs are processed in parallel through a unified contrastive learning framework, enabling us to measure the confidence of each triple by considering three types of anomaly signals: self-contradiction in the relational structure, global consistency across triples, and attribute-structure dependency.

Multi-view Construction. Traditional KG embedding models typically only consider entities and relations within triples, overlooking the global correlations among

triples. To capture these correlations, we introduce a novel approach that transforms the KG into hyper-views at the triple level, treating each relational triple as a node in the graph. Specifically, we construct a *relational hypergraph* by linking relational triples that share the same entity in the original KG, ensuring that the transformation preserves the potential for error detection within individual triples.

Definition 2. ***Relational Hypergraph.*** *Given a knowledge graph* $\mathcal{G} = \{(h, r, t)|h, t \in \mathcal{E}, r \in \mathcal{R}\}$*, the corresponding relational hypergraph is the triple-level graph* $\mathcal{G}_r = (\widetilde{\mathcal{V}}, \widetilde{\mathcal{A}_r}, \widetilde{\mathcal{X}_r})$*, where* $\widetilde{\mathcal{V}}$ *and* $\widetilde{\mathcal{A}_r}$ *are the sets of nodes and adjacency matrix, respectively.* $\widetilde{\mathcal{X}_r} = \{\Gamma(h, r, t) \mid (h, r, t) \in \mathcal{G}\}$ *represents the feature matrix of* $\widetilde{\mathcal{V}}$*, where* $\Gamma(\cdot)$ *is a concatenation function.* $\widetilde{\mathcal{A}_r}(v, u|u, v \in \widetilde{\mathcal{V}}) = 1$*, if* u *and* v *share the same entity in the original KG, i.e.,* $\mathcal{G}$*.*

Next, we build an *attribute hypergraph* to capture the relationship between entities and their attributes. This hypergraph uses the attributes of the head and tail entities in a triple to reconstruct the semantics of the triple from the attribute perspective.

Definition 3. ***Attribute Hypergraph.*** *Given the same knowledge graph* $\mathcal{G} = \{(h, r, t)|h, t \in \mathcal{E}, r \in \mathcal{R}\}$*, with entity attribute set* $\mathcal{A} = \{\{a_{h,1}, a_{h,2}, \ldots, a_{h,|A_h|}\}|h \in \mathcal{E}\}$ *where* $a_{h,i}$ *is the* i^{th} *attribute type for entity* h*, the attribute hypergraph can be denoted as* $\mathcal{G}_a = (\mathcal{V}_a, \mathcal{A}_a, \mathcal{X}_a)$*, where* $\mathcal{V}_a$ *and* $\mathcal{A}_a$ *are the set of nodes and adjacency matrix, respectively, which have the same structure as* $\mathcal{G}_r$*.* $\mathcal{X}_a$ *represents the feature matrices of each node* $v_i \in \mathcal{V}_a$ *learned from corresponding entity attributes.*

These two hypergraphs, *relational hypergraph* and *attribute hypergraph*, can be seen as complementary views of the same KG, providing richer information for error detection. The *relational hypergraph* captures the global correlations between relational triples, while the *attribute hypergraph* highlights the semantic meaning derived from entity attributes. By measuring the consistency between these two views, we can more accurately assess the trustworthiness of each triple in the original KG.

Learning from View I: Relational Hypergraph. Traditional KG embedding methods typically focus on the local relationships within individual triples, disregarding the broader context. To address this, we propose a new dual-view encoder that simultaneously learns both local and global context for relational triples.

Local View of Relational Structure Modeling. The *relational hypergraph* transforms triples into a structure that can lose some local relational details, such as the head-relation-tail structure inherent within triples. To preserve this local information, we initialize the embedding of entities and relations in the original KG, then use a BiLSTM layer to encode the local structure of each triple. The local representation p_i for the i^{th} triple (h, r, t) is defined as:

$$p_i = \mathcal{G}_{local}(h, r, t) = f_{concat}(f_{BiLSTM}\left(e_h, e_r, e_t\right)). \tag{4.3}$$

Global View of Neighbor Information Aggregation. Beyond the local structure, the global context of neighboring triples also contains valuable information for detecting anomalies. To model this, we employ a graph attention network (GAT) to selectively aggregate features from neighboring triples. Given an anchor triple $p_i \in \mathbb{R}^d$, its embedding is updated by attending over neighboring triples, $\{p_1, p_2, \ldots, p_m\}$, through a two-layer attention mechanism:

$$att_{ij} = f_{att}\left(\mathbf{W}p_i, \mathbf{W}p_j\right). \tag{4.4}$$

The attention values are normalized using the softmax function:

$$\alpha_{ij} = \frac{\exp\left(att_{ij}\right)}{\sum_{k=1}^{m} \exp\left(att_{ik}\right)}. \tag{4.5}$$

Finally, the updated embedding is computed as:

$$x_i = \sigma\left(\sum_{j=1}^{m} \alpha_{ij} \mathbf{W}p_j\right). \tag{4.6}$$

Learning from View II: Attribute Hypergraph. In addition to relational structure, entity attributes contain valuable semantic information that enhances KG em-

bedding quality. To effectively incorporate this information, we design a relation-specific encoder, g_{attr}, which learns attribute-based representations from the *attribute hypergraph*.

Attribute Hypergraph Embedding. Entity attributes are diverse and unevenly distributed across different entities. An entity can have distinct attribute sets depending on its role in different triples. For instance, a person entity might have attributes related to *research area* when playing the role of a scientist and *pen name* when considered as a writer. To handle these variations, we introduce a relation-specific mechanism that selects relevant attributes based on the relation in the triple.

Given a triple (h, r, t), the attributes of the head and tail entities are aggregated using relation-specific attention mechanisms. The attribute-based entity representations $\hat{e}_h$ and $\hat{e}_t$ are computed as:

$$att_{h,i} = f_{att}\left(f_{emb}\left(e_r\right), f_{emb}\left(a_{h,i}\right)\right), \tag{4.7}$$

$$\alpha_{h,i} = \frac{\exp\left(att_{h,i}\right)}{\sum_{j=1}^{|A_h|}\exp\left(att_{h,j}\right)}, \tag{4.8}$$

and the final aggregated representation for the entity h is:

$$\hat{e}_h = \sum_{i=1}^{|A_h|} \alpha_{h,i} * f_{emb}\left(a_{h,i}\right). \tag{4.9}$$

Model Learning. To learn robust features from both *relational hypergraph* and *attribute hypergraph*, we utilize a contrastive loss function that maximizes the mutual information between the embeddings derived from these two views.

Contrasting Between Structure and Attribute Views. The *relational hypergraph* models the graph structure, while the *attribute hypergraph* captures the entity attributes. These two views are complementary, and we apply the normalized temperature-scaled cross entropy loss to maximize their mutual information:

$$\mathcal{L}_{\text{sa}}(x_i, z_i) = -\log \frac{\exp\left(\text{sim}\left(x_i, z_i\right)/\tau\right)}{\sum_{j\in\{1,2,\ldots,N\}\setminus\{i\}}\exp\left(\text{sim}\left(x_i, z_j\right)/\tau\right)}, \tag{4.10}$$

Contrasting Between Local and Global Views. We also contrast the local and global views of each triple, as the relational structure and its global context provide complementary information. The contrastive loss for this objective is defined as:

$$\mathcal{L}_{\text{lg}}(p_i, x_i) = -\log \frac{\exp\left(\text{sim}\left(p_i, x_i\right)/\tau\right)}{\sum_{j\in\{1,2,\ldots,N\}\setminus\{i\}} \exp\left(\text{sim}\left(p_i, x_j\right)/\tau\right)}. \tag{4.11}$$

Finally, combining these two losses, we obtain the overall contrastive loss:

$$\mathcal{L}_{con} = \frac{1}{2N}\sum_{i=1}^{N}\left(\mathcal{L}_{sa}\left(x_i, z_i\right) + \mathcal{L}_{lg}\left(p_i, x_i\right)\right). \tag{4.12}$$

Triple Confidence Estimation. We calculate the triple confidence score based on three key anomaly signals: self-contradiction within the relational structure, global consistency across triples, and attribute-structure dependency. These signals help us assess the reliability of each triple.

Self-contradictory Measurement. In the relational structure, the closer a triple fits the translation assumption $\mathbf{h} + \mathbf{r} \simeq \mathbf{t}$, the more reliable it is. We define the local triple confidence $LT(h, r, t)$ based on the Euclidean distance between the entities:

$$LT(h, r, t) = \frac{1}{1 + e^{-\|e_h + e_r - e_t\|_2}}. \tag{4.13}$$

Global Acknowledgment Estimation. The degree of acknowledgment from neighboring triples reflects whether a target triple is valid. We define the global triple confidence $GT(h, r, t)$ as the cosine similarity between the local and global representations of the triple:

$$GT(h, r, t) = sim(p_i, x_i), \tag{4.14}$$

Structure-attribute Dependency Estimation. The consistency between the relational and attribute hypergraph views is an important signal for determining the correctness of a triple. We compute the attribute-structure confidence $AT(h, r, t)$ as:

$$AT(h, r, t) = sim(x_i, z_i), \tag{4.15}$$

Final Triple Confidence. We combine these three signals to calculate the final confidence score for each triple:

$$C(h,r,t) = \sigma(LT(h,r,t) + \lambda_1 \cdot GT(h,r,t) + \lambda_2 \cdot AT(h,r,t)), \tag{4.16}$$

4.2.3 Joint Adaptive Training Scheme

To integrate the learned confidence information into the KG embedding process, we define a comprehensive objective function:

$$\mathcal{L} = \mathcal{L}_{con} + \beta \mathcal{L}_{emb}. \tag{4.17}$$

This adaptive training scheme ensures that the KG representation learning and confidence learning processes mutually reinforce each other, allowing the model to progressively improve its ability to filter out erroneous triples while learning accurate KG embeddings.

4.3 Experiments

In this section, we present comprehensive experiments to evaluate the performance of the proposed AEKE framework across various real-world KGs. The primary goals are to answer the following research questions:

- **Q1**: How effectively does AEKE identify noisy triples in knowledge graphs?
- **Q2**: How do the KG embeddings learned by AEKE compare to the state-of-the-art KG embedding models in terms of quality?
- **Q3**: What contributions do the individual components of AEKE make to its overall performance?

Table 4.1: Error detection results of Precision@K and Recall@K based on the three datasets with anomaly ratio = 5%.

		FB15K-237					DB15K					YAGO15K				
		K=1%	K=2%	K=3%	K=4%	K=5%[a]	K=1%	K=2%	K=3%	K=4%	K=5%[a]	K=1%	K=2%	K=3%	K=4%	K=5%[a]
Precision@K	TransE	0.756	0.674	0.605	0.546	0.488	0.671	0.566	0.535	0.472	0.443	0.534	0.455	0.362	0.319	0.279
	ComplEx	0.718	0.651	0.590	0.534	0.485	0.679	0.612	0.532	0.469	0.424	0.503	0.426	0.369	0.310	0.273
	DistMult	0.709	0.646	0.582	0.529	0.483	0.638	0.587	0.523	0.499	0.445	0.513	0.423	0.347	0.347	0.302
	SimplE	0.744	0.667	0.611	0.556	0.515	0.709	0.616	0.554	0.504	0.455	0.560	0.459	0.373	0.319	0.281
	TuckER	0.742	0.680	0.614	0.552	0.514	0.711	0.630	0.550	0.509	0.452	0.549	0.454	0.375	0.331	0.276
	EARL	0.762	0.692	0.639	0.582	0.531	0.729	0.652	0.573	0.530	0.483	0.578	0.487	0.394	0.338	0.303
	TTMF	0.815	0.767	0.713	0.612	0.579	0.744	0.685	0.623	0.557	0.510	0.701	0.569	0.454	0.413	0.346
	CAGED	0.852	0.796	0.735	0.665	0.595	0.802	0.722	0.658	0.596	0.554	0.724	0.599	0.489	0.429	0.373
	CKRL	0.789	0.736	0.684	0.630	0.574	0.787	0.723	0.634	0.599	0.526	0.662	0.531	0.438	0.382	0.327
	NoiGAN	0.837	0.788	0.727	0.649	0.585	0.823	0.718	0.676	0.606	0.553	0.737	0.592	0.477	0.403	0.379
	CrossVal	0.874	0.814	0.742	0.667	0.596	0.819	0.707	0.645	0.572	0.548	0.754	0.647	0.562	0.500	0.424
	AEKE	**0.892**	**0.822**	**0.753**	**0.691**	**0.614**	**0.853**	**0.756**	**0.705**	**0.625**	**0.582**	**0.778**	**0.676**	**0.590**	**0.519**	**0.440**
Recall@K	TransE	0.151	0.269	0.363	0.437	0.488	0.134	0.226	0.321	0.377	0.443	0.106	0.182	0.217	0.255	0.279
	ComplEx	0.144	0.260	0.354	0.427	0.485	0.135	0.244	0.319	0.375	0.424	0.100	0.170	0.221	0.248	0.273
	DistMult	0.142	0.258	0.349	0.423	0.483	0.127	0.234	0.313	0.399	0.445	0.102	0.169	0.208	0.277	0.302
	SimplE	0.149	0.267	0.366	0.445	0.515	0.142	0.246	0.332	0.403	0.455	0.112	0.184	0.224	0.255	0.281
	TuckER	0.148	0.272	0.368	0.442	0.514	0.142	0.252	0.330	0.407	0.452	0.110	0.182	0.225	0.265	0.276
	EARL	0.152	0.277	0.383	0.466	0.531	0.146	0.261	0.344	0.424	0.483	0.116	0.195	0.236	0.270	0.303
	TTMF	0.163	0.306	0.427	0.489	0.579	0.148	0.274	0.373	0.445	0.510	0.140	0.227	0.272	0.330	0.346
	CAGED	0.171	0.318	0.441	0.532	0.595	0.160	0.288	0.395	0.477	0.554	0.145	0.240	0.293	0.343	0.374
	CKRL	0.158	0.294	0.410	0.504	0.574	0.157	0.289	0.380	0.479	0.526	0.132	0.212	0.262	0.305	0.327
	NoiGAN	0.167	0.315	0.436	0.519	0.585	0.164	0.287	0.405	0.484	0.553	0.147	0.236	0.286	0.322	0.379
	CrossVal	0.175	0.325	0.445	0.533	0.596	0.163	0.282	0.387	0.457	0.548	0.150	0.258	0.337	0.400	0.424
	AEKE	**0.178**	**0.328**	**0.452**	**0.552**	**0.614**	**0.170**	**0.302**	**0.423**	**0.500**	**0.582**	**0.155**	**0.270**	**0.354**	**0.415**	**0.440**

4.3.1 Datasets

We evaluate AEKE on three real-world benchmark datasets: FB15K-237, YAGO15K, and DB15K. These datasets are known for their high reliability due to extensive human curation. Following prior works [104, 19], we introduce 5% and 15% noisy triples into each dataset to simulate real-world errors. The noisy triples are generated by replacing the head or tail entity in a given triple (h, r, t) with an entity that has appeared in the same position with the same relation in the dataset, producing a more challenging and realistic form of error.

FB15K-237 is a widely-used subset of Freebase containing 114 relations and 10,054 entities, offering a large-scale knowledge base with over 1 billion triples. FB15K-

237 improves upon the FB15K dataset by addressing issues with inverse relations and handling symmetric, asymmetrical, and combinatorial relationships, as well as attributes.

DB15K is a subset of Wikidata, designed to address the weaknesses of Wikipedia-based datasets. It excludes inverse relations to prevent test leakage, similar to the process used for FB15K-237.

YAGO15K is built by augmenting WordNet with over 1 million entities and aligning it with Freebase to form a knowledge graph.

4.3.2 Capability of AEKE in Distinguishing KG Errors (Q1)

In this section, we evaluate AEKE's ability to distinguish errors within KGs. Specifically, we rank all triples in the target KG by their confidence scores in ascending order. The top-ranked triples are considered potential errors. We conduct experiments on FB15K-237, DB15K, and YAGO15K with 5% and 15% noisy triples. Below, we discuss the experiment setup, baseline models, and evaluation metrics.

Baselines. We compare AEKE to several baseline models from three categories: error-aware embedding methods, error detection alternatives, and KG embedding methods. The baselines are as follows:

TransE: Assumes entities and relations are embedded in the same space, where the tail entity can be predicted by the head entity and relation.

DistMult: A bilinear model that calculates the confidence of potential semantics for entities and relations.

ComplEx: An extension of DistMult that uses complex numbers to model both symmetric and asymmetric relations.

SimplE: An interpretable embedding model that employs weight tying to integrate

specific background knowledge.

TuckER: A linear model for link prediction based on Tucker decomposition of a binary tensor containing known facts.

EARL: Focuses on learning embeddings for reserved entities, considering connected relations and neighbors.

TTMF: Uses semantic information to calculate the trustworthiness of triples, distinguishing between normal and anomalous triples.

CAGED: Combines KG embedding and contrastive learning to assess the trustworthiness of each triple based on multi-view consistency.

NoiGAN: Uses Generative Adversarial Networks (GANs) for noise-aware knowledge graph embeddings.

CKRL: A confidence-aware representation learning method that assigns confidence scores to triples to identify potential noise.

CrossVal: Uses external human-curated knowledge graphs as auxiliary information to improve error detection within the target KG.

Evaluation Protocol. We implement all models using PyTorch, and the experiments are conducted on an Nvidia 3090 GPU. The Adam optimizer is used with a batch size of 256, a learning rate of 0.01, and Xavier initialization for model parameters. The embedding size is set to 100 for all models.

We use ranking measures to evaluate the performance of each model. A triple's confidence score is used to rank it, with lower scores indicating higher likelihood of errors. The evaluation metrics are:

Precision@K: Measures the percentage of true anomalies in the top K rankings.

$$\text{Precision@K} = \frac{|\text{ Errors Found in Top K }|}{K}. \tag{4.18}$$

Recall@K: Measures the percentage of true anomalies found within the total errors in the KG.

$$\text{Recall@K} = \frac{|\text{ Errors Found in Top K }|}{|\text{ Total Errors in KG }|}. \tag{4.19}$$

Experimental Results. The results for Q1 are summarized in Table 4.1. We observe the following:

Obs. 1. AEKE outperforms both embedding methods and state-of-the-art error detection baselines. For example, with a 5% anomaly ratio and $K = 5\%$, AEKE achieves a 1.8% improvement over the second-best method in terms of recall and precision.

Obs. 2. While KG embedding methods like TransE, ComplEx, and DistMult show decent performance, they lag behind tailored error detection methods. This is because these embedding methods do not account for errors in the KG, resulting in less discriminative representations for normal and noisy triples.

Obs. 3. Incorporating auxiliary human-curated information, as seen in CrossVal, improves error detection. However, AEKE surpasses it by leveraging the relational structure within triples, the global context across triples, and the semantics from entity attributes, making AEKE more effective at error detection.

4.3.3 Quality of KG Embeddings Learned by AEKE (Q2)

Next, we evaluate the quality of the KG embeddings learned by AEKE using the KG completion task. In this task, the goal is to predict missing entities in incomplete triples. We compare AEKE with strong baselines to assess the quality of its embeddings.

Baseline Methods. We compare AEKE against the following baselines:

Table 4.2: Results of knowledge graph completion.

	Dataset	FB15K-237				DB15K				YAGO15K			
	Metrics	MRR	Hit@1	Hit@3	Hit@10	MRR	Hit@1	Hit@3	Hit@10	MRR	Hit@1	Hit@3	Hit@10
0%	TransE	0.302	0.211	0.344	0.468	0.420	0.385	0.431	0.509	0.311	0.199	0.369	0.523
	SimplE	0.288	0.202	0.314	0.455	0.437	0.392	0.445	0.505	0.296	0.188	0.344	0.487
	EARL	0.308	0.209	0.338	0.474	0.447	0.389	0.459	0.508	0.317	0.199	0.372	0.509
	RGCN	0.363	0.306	0.387	0.512	0.474	0.410	0.487	0.576	0.381	0.269	0.424	0.588
	RGHAT	0.422	0.362	0.446	**0.535**	0.483	0.422	0.499	0.588	0.412	0.295	0.467	0.618
	NoiGAN	0.424	0.371	0.445	0.526	0.480	0.421	0.483	0.581	0.413	0.310	0.465	0.605
	CKRL	0.410	0.362	0.438	0.536	0.482	0.416	0.490	0.586	0.408	0.298	0.461	0.610
	Ours	**0.427**	**0.375**	**0.453**	0.534	**0.486**	**0.426**	**0.503**	**0.587**	**0.418**	**0.316**	**0.474**	**0.622**
5%	TransE	0.294	0.201	0.335	0.465	0.407	0.358	0.417	0.481	0.288	0.161	0.345	0.493
	SimplE	0.266	0.178	0.281	0.416	0.407	0.354	0.412	0.466	0.277	0.166	0.313	0.455
	EARL	0.284	0.194	0.308	0.437	0.407	0.366	0.416	0.466	0.283	0.180	0.348	0.468
	RGCN	0.348	0.261	0.375	0.493	0.456	0.408	0.472	0.551	0.365	0.250	0.397	0.562
	RGHAT	0.401	0.342	0.442	0.507	0.460	0.411	0.472	0.565	0.392	0.275	0.442	0.580
	NoiGAN	0.418	0.360	0.440	0.515	0.474	0.415	0.479	0.578	0.404	0.305	0.457	0.598
	CKRL	0.400	0.340	0.426	0.512	0.471	0.410	0.483	0.572	0.397	0.292	0.449	0.594
	Ours	**0.424**	**0.369**	**0.447**	**0.519**	**0.482**	**0.424**	**0.499**	**0.583**	**0.412**	**0.308**	**0.465**	**0.612**
15%	TransE	0.267	0.185	0.309	0.451	0.386	0.340	0.401	0.469	0.276	0.149	0.334	0.473
	SimplE	0.242	0.154	0.241	0.361	0.368	0.314	0.358	0.416	0.231	0.134	0.288	0.419
	EARL	0.259	0.169	0.266	0.381	0.368	0.326	0.362	0.416	0.237	0.147	0.32	0.431
	RGCN	0.325	0.237	0.356	0.476	0.437	0.398	0.455	0.534	0.343	0.238	0.375	0.549
	RGHAT	0.395	0.332	0.416	0.498	0.442	0.398	0.454	0.540	0.371	0.258	0.427	0.565
	NoiGAN	0.409	0.348	0.427	0.506	0.463	0.401	0.462	0.566	0.385	0.295	0.448	0.589
	CKRL	0.396	0.334	0.422	0.501	0.458	0.400	0.467	0.559	0.375	0.275	0.437	0.579
	Ours	**0.417**	**0.363**	**0.440**	**0.512**	**0.475**	**0.411**	**0.492**	**0.579**	**0.401**	**0.299**	**0.451**	**0.600**

- *Embedding-based*: **TransE**, **SimplE**, **EARL**
- *State-of-the-art completion models*: **RGCN**, **RGHAT**
- *Error-aware embedding methods*: **NoiGAN**, **CKRL**

Evaluation Protocol We focus on entity prediction, where the task is to predict the head or tail entity of a given triple $(?, r, t)$ or $(h, r, ?)$, respectively. We use the Mean Rank (MRR) and Hits@K as evaluation metrics.

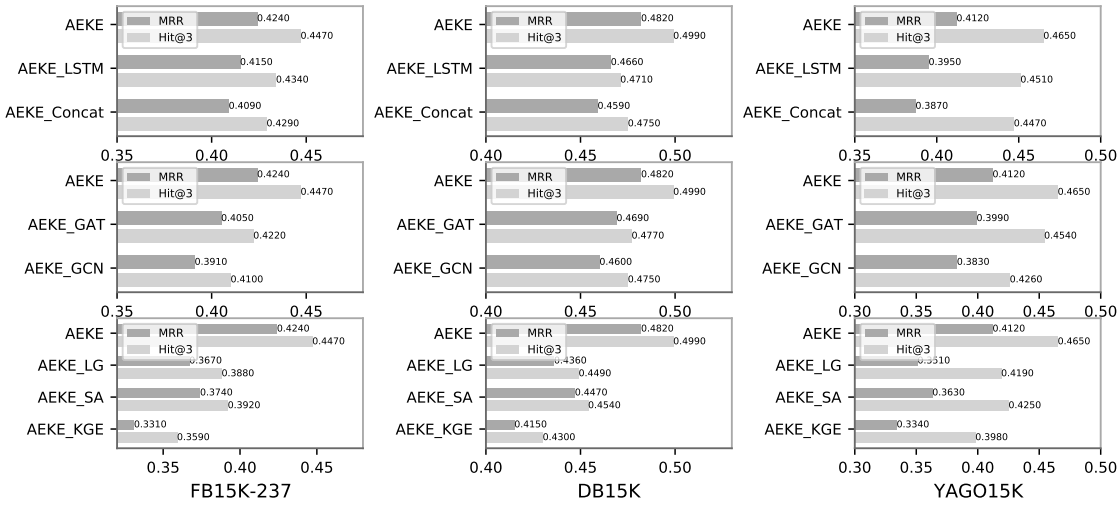


Figure 4.3: KG completion results of AEKE variants based on the three datasets with anomaly ratio = 5%.

Experimental Results The results on FB15K-237, DB15K, and YAGO15K with 5% and 15% noisy triples are presented in Table 4.2. Key observations include:

Obs. 1. AEKE consistently outperforms embedding-based models and other error-aware methods, demonstrating the effectiveness of the learned KG embeddings.

Obs. 2. Error-aware embedding methods like NoiGAN and CKRL perform better than traditional embedding methods, but AEKE shows superior performance due to its ability to model both relational structure and entity attributes.

Obs. 3. As the anomaly rate increases, AEKE's performance gap over other methods grows. For example, with a 5% anomaly ratio, AEKE shows 0.3%, 0.6%, and 0.5% improvements on FB15K-237, DB15K, and YAGO15K, respectively. At a 15% anomaly ratio, the improvements are more significant: 0.8%, 1.2%, and 1.6%.

4.3.4 Ablation Study (Q3)

We conduct an ablation study to assess the contribution of each component of AEKE. Four variants of AEKE are evaluated, and the effects of different modules are discussed.

Role of Attribute Hypergraph Encoder We evaluate the importance of the attribute encoder by comparing AEKE with and without the attribute view. Removing the attribute encoder leads to a significant drop in performance, confirming the importance of entity attributes in error-aware embedding.

Role of Relational Hypergraph Encoder To assess the effectiveness of capturing the relational structure, we compare AEKE with variants using different methods for encoding relational triples. AEKE outperforms these variants due to its tailored attention mechanism, which effectively filters out noisy triples.

4.4 Summary

Learning accurate and reliable embeddings for entities and relations in knowledge graphs (KGs) plays a critical role in enabling various downstream applications. While many existing approaches leverage the internal structure of KGs to train models for error detection, they are limited by the fact that the topological information present in KGs alone is insufficient for validation tasks, particularly in real-world scenarios. In this work, we introduce a novel framework for KG representation learning, named AEKE, which integrates semantic data from entity attributes to automatically validate triples in KGs. We conceptualize the original KG, which lacks attribute information, as a *relational hypergraph*, and construct an *attribute hypergraph* using the supplementary entity attribute information. This attribute hypergraph serves as an

alternative view of the target KG. The confidence score for each triple is computed by considering multiple factors: inconsistency within the triple itself, alignment between local and global graph structures, and the compatibility between structure and attributes across triple-level views. Experimental results show that AEKE outperforms current state-of-the-art error detection methods for KGs. Given that real-world KGs continuously evolve with new data, extending AEKE to handle temporal KG representations would be a highly valuable direction for future work. Furthermore, we aim to explore the use of AEKE's error-aware KG representation learning in practical downstream tasks such as question answering and recommender systems.

Chapter 5

Logical Reasoning for Inductive Knowledge Graph Completion

5.1 Introduction

Knowledge graphs (KGs) are structured collections of real-world information, typically represented as triples (h, r, t), where h is the head entity, r is the relation, and t is the tail entity [8, 55, 67, 58, 23]. These graphs provide a way to model complex relationships between entities across various domains. However, KGs are often incomplete [116, 24], and manually curating facts is expensive, leading to the need for automated knowledge graph completion (KGC) methods [117, 1].

Currently, embedding-based techniques [10, 59, 90] dominate the field of KGC. These methods embed entities and relations into continuous vector spaces and predict missing relations between entities. While effective, these techniques generally operate under a transductive setting, where they assume all entities are present during both training and inference. This limitation arises when new entities are introduced, as the embeddings need to be retrained, which is impractical due to the constant growth and scale of real-world KGs.

Recent research has turned towards inductive KGC [98, 92], where models are designed to predict missing relations for unseen entities, which more accurately reflects real-world scenarios. In dynamic KGs, such as those used in e-commerce or biomedical domains, new entities, like users or products, are constantly added. To address this, several approaches have incorporated external resources, such as entity attributes, textual descriptions, and ontologies, to aid inductive KGC. However, these external data sources are often difficult to obtain, limiting their practical applicability. Alternatively, some methods have attempted to learn entity-independent rules for KGC, either through statistical [29] or differentiable [108] approaches, treating the problem as rule mining. However, rule-based techniques often struggle with scalability and generalization due to the domain-specific nature of rules across different KGs.

The rise of graph neural networks (GNNs) has led to the development of inductive KGC techniques based on message passing (MP) [78, 92]. GraIL [89] is a notable early work that learns logical rules through reasoning over subgraphs surrounding a target triple. It aggregates structural information from neighboring entities within the subgraph to generate entity representations. While GraIL and subsequent methods [114] have shown promising results, they face two main challenges: (i) Data sparsity, especially when new entities lack sufficient connections to the existing graph, and (ii) The limitation of traditional MP approaches that treat messages as entity-specific, which overlooks the importance of relation semantics [92]. This is problematic for inductive KGC, where the reasoning process should be entity-independent.

In real-world KGs, the surrounding subgraph of a given triple contains the necessary logical evidence to infer the relationship between the entities. For instance, in Figure 5.1, learning the relation pattern between *Elon Musk*, *TESLA*, and *California State* can help predict the relationship *(Martin Eberhard, :LiveIn, California State)* based on the inferred pattern "$:LiveIn \simeq :WorkAt \wedge :LocatedIn$". To address this, we introduce a novel framework called NORAN, which focuses on mining relation semantics for inductive KGC.

Our approach first redefines the KGC problem to bridge the gap between traditional embedding-based methods and inductive settings. Inspired by the insights gained, we propose the *relation network*, a novel graph constructed by focusing on relations in the original KG. This relation network provides a hypergraph-like structure that represents the distribution and correlation of relations. By modeling this relation network through message-passing, we can capture entity-independent relation patterns, enabling more effective inductive KGC. We formally define the logical evidence extracted from the relation network and demonstrate its effectiveness in inductive KG completion.

Our main contributions are summarized as follows:

- We introduce NORAN, a novel framework for inductive KG completion that learns latent relation semantics.

- We propose the *relation network*, a hypergraph-based representation of the KG that centers on relations, and formally define inductive KGC as *k-hop logic reasoning* over this hypergraph.

- We introduce the *informax* training objective, which enables the effective capture of logical evidence from the relation network for KGC.

- We provide a theoretical analysis to identify the most effective message-passing strategies and offer guidelines for selecting models that enable inductive reasoning over the relation network.

- Extensive experiments on five real-world KG benchmarks demonstrate the superior performance of NORAN.

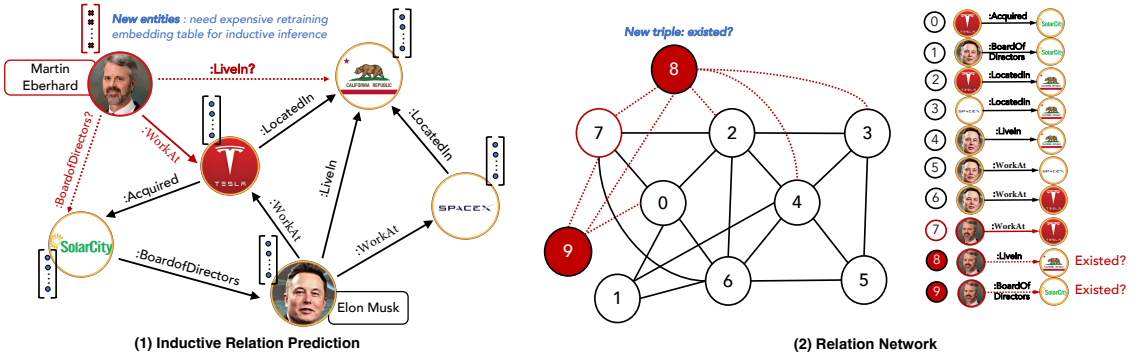


Figure 5.1: *(i)* A toy example of inductive knowledge graph completion, i.e. predicting the relationship ("?") for unseen entities, e.g. "Martin Eberhard", provided with a few links, e.g. (Martin, :WorkAt, TESLA); *(ii)* Illustration of the corresponding *relation network* for knowledge graph, which regards each triple as a relational node and thus could aggregate context information for inductive inference without expensive retraining look-up embedding tables as embedding-based paradigm.

5.2 Inductive Knowledge Graph Completion

In this section, we introduce the NORAN framework, designed to uncover latent patterns among relational instances for inductive KG completion. As illustrated in Figure 5.1, NORAN is composed of three core components: (i) Relation network construction, which redefines KG modeling with a focus on relations. (ii) Relational message passing, a framework that implicitly extracts logical evidence independent of entities through the relation network. (iii) Training scheme and KG inference, which includes a detailed theoretical analysis to identify effective message-passing strategies, offering insights into model selection for inductive reasoning over the relation network.

5.2.1 Relation Network Construction

Conventional KG representation methods often represent a KG as a heterogeneous graph centered on entities, with relations serving as semantic edges. However, such approaches face limitations in capturing intricate relational semantics and fail to provide expressive representations for inductive KG completion. To bridge the gap between traditional KG representation learning and the requirements of inductive KGC, we propose a new perspective that focuses on relation correlations. This novel modeling approach introduces a hyper-level view of KGs based on relations, and we formally define inductive KG completion as *k-hop logical reasoning* over an entity-independent network.

Table 5.1: The general description framework for MP layers

MP layer	Motivation	Convolutional Matrix $\mathcal{C}$	Feature Transformation f	Category of $\mathcal{C}$
GCN	Spatial Convolution	$\mathcal{C} = \widetilde{\boldsymbol{D}}^{1/2}(\boldsymbol{A}+\boldsymbol{I})\widetilde{\boldsymbol{D}}^{1/2}$	$f = \boldsymbol{W} \in \mathbb{R}^{d_k \times d_{k+1}}$	Fixed
GraphSAGE	Inductive Learning	$\mathcal{C} = \widetilde{\boldsymbol{D}}^{-1}(\boldsymbol{A}+\boldsymbol{I})$	$f = \boldsymbol{W} \in \mathbb{R}^{d_k \times d_{k+1}}$	Fixed
GIN	WL-Test	$\mathcal{C} = \boldsymbol{A}+\boldsymbol{I}$	f is a two-layer MLP	Fixed
SGC	Scalability	$\mathcal{C} = \widetilde{\boldsymbol{D}}^{1/2}(\boldsymbol{A}+\boldsymbol{I})\widetilde{\boldsymbol{D}}^{1/2}$	$f = lambda\ x : x$	Fixed
GAT	Self Attention	$\begin{cases}\mathcal{C} = (\boldsymbol{A}+\boldsymbol{I})\cdot\mathcal{T} \\ \mathcal{T}_{(i,j)} = \frac{exp([\Theta\boldsymbol{x}_i \| \Theta\boldsymbol{x}_j]\cdot\boldsymbol{a})}{\sum_{k\in\mathcal{N}(i)\cup i} exp([\Theta\boldsymbol{x}_i \| \Theta\boldsymbol{x}_j]\cdot\boldsymbol{a})}\end{cases}$	$f = \Theta \in \mathbb{R}^{d_k \times d_{k+1}}$	Learnable

[a] $\widetilde{\boldsymbol{D}}$ is the diagonal matrix of $(\boldsymbol{A}+\boldsymbol{I})$

Relation network. To enhance the ability to capture logical evidence for reasoning, we propose a new KG modeling perspective, which transforms the target KG into hyper-level views by representing each triple as a relational node. Specifically, we construct the *relation network* via a relation-driven process. Following a systematic approach, we treat each triple as a node in the relational graph and establish connections between nodes if they involve a shared entity within the original KG.

Definition 4. ***Relation network.*** *Given a knowledge graph* $\mathcal{G} = \{(h,r,t)|h,t \in \mathcal{E}, r \in \mathcal{R}\}$*, the corresponding relation network is the triple-level graph* $\mathcal{G}_r = (\widetilde{\mathcal{V}}, \widetilde{\mathcal{A}}, \widetilde{\mathcal{X}_r})$*,*

where $\widetilde{\mathcal{V}}$ *and* $\widetilde{\mathcal{A}}$ *are the sets of relational nodes and adjacency metrics, respectively.* $\widetilde{\mathcal{X}_r} = \{\Gamma(h,r,t) \mid (h,r,t) \in \mathcal{G}\}$ *represents the feature matrix of* $\widetilde{\mathcal{V}}$*, where* $\Gamma(\cdot)$ *is the concatenation function that transforms each relational triple into a node, i.e.,* $v = \Gamma(h,r,t)$*.* $\widetilde{\mathcal{A}}(v,u|v,u \in \widetilde{\mathcal{V}}) = 1$*, if* v *and* u *share the same entity in the original KG.*

Interpretation with logic reasoning. The topology of the relation network effectively depicts relational distributions, while the connections between nodes convey relation correlations. By modeling the relation network using a k-layer message-passing framework, we can capture relational patterns as entity-independent context for inductive KG completion. This entity-independent context is formally defined as *k-hop logical evidence* over the relation network.

Definition 5. ***K-hop logic evidence.*** *Let* $\widetilde{\mathcal{G}} = (\widetilde{\mathbf{V}}, \widetilde{\mathcal{A}})$ *be the relation network of a knowledge graph* $\mathcal{G} = (\mathcal{E}, \mathcal{R})$*. Given any center node* $v_i \in \widetilde{\mathbf{V}}$*, the k-hop ego graph* $\widetilde{\mathcal{G}}_i = (\widetilde{\mathbf{V}}_i, \widetilde{\mathbf{A}}_i)$ *centering at* v_i *contains the relational contextual information for logical reasoning. In this paper, we model such contextual information as* $\Lambda^k(v_i)$ *via a k-layer message-passing network* Ω*. Thus, the overall logic evidence of knowledge graph* $\mathcal{G}$ *can be modeled by traversing all k-hop ego graphs, i.e.,* $\Lambda(\mathcal{G}) = \Omega_{GNN}(\Lambda^k(v) \mid v \in \widetilde{\mathcal{G}})$*.*

Taking the logical evidence "*:LiveIn = :WorkAt* $\wedge$ *:LocatedIn*" in Fig. 5.1 as an example, the relation pattern among *Elon Musk*, *TESLA*, and *California State* illustrates a concrete case of logical evidence. The triple *(Martin Eberhard, :LiveIn, California State)* can then be inferred with confidence. According to Definition 5, such logical evidence corresponds to k-hop ego graphs centered on target nodes in the relation network (Fig. 5.1 (*ii*)), describing the relational context required for reasoning.

The construction of the relation network offers two main advantages: (*i*) It introduces a novel perspective for KG modeling, enabling entity-independent logic evidence to be naturally captured via k-hop ego graphs sampled from the relation network. This

eliminates the need for fine-grained embeddings of unseen entities. (*ii*) Compared to the original KG, the relation network exhibits a denser structure and is compatible with any GNN model. An incident graph (incidence matrix representation) could also encode relationships between nodes (entities) and edges (relations) in a bipartite structure, where rows represent nodes and columns represent edges. While the incidence matrix captures explicit connections between entities and relations, it lacks the higher-order relational semantics that NORAN's Relation Network emphasizes.

5.2.2 Relational Message Passing

To effectively extract entity-independent contextual information as logical evidence, we introduce a novel relational message-passing framework tailored for inductive KG completion.

Feature initialization. Since each instance in the relation network corresponds to a triple (h, r, t) from the original KG, we first initialize the embeddings of entities and relations in the original KG randomly. To capture the local relational structures within each triple, we utilize a set of Bi-LSTM units as a local information modeling layer. The embedding of each triple is obtained as follows:

$$\boldsymbol{x}_i = \Gamma(h, r, t) = concat(\Phi(\boldsymbol{e}_h, \boldsymbol{e}_r, \boldsymbol{e}_t)), \tag{5.1}$$

where $\Phi(\cdot)$ is a Bi-LSTM unit, and $\boldsymbol{e}_h, \boldsymbol{e}_r, \boldsymbol{e}_t$ are the initial embeddings of h, r, and t. The resulting triple embedding $\boldsymbol{x}_i$ effectively encodes the relational structure of the input triple, and it is used as the initial node embedding in the relation network. During both training and inference, the entity embeddings are kept *fixed*.

Message passing. The message-passing layer is defined as:

$$\boldsymbol{X}^{(k+1)} = \sigma\big(\mathcal{C}^{(k)}\boldsymbol{X}^{(k)} \circ f^{(k)}\big), \tag{5.2}$$

where $\boldsymbol{X}^{(k)}$ is the relational embedding at the k-th layer; $\mathcal{C}^{(k)}$ is the convolutional matrix for the k-th layer; $f^{(k)} : \mathbb{R}^{d_k} \to \mathbb{R}^{d_{k+1}}$ is the linear transformation matrix; and σ is the activation function.

The reformulated message-passing layers are summarized in Table 5.1. By performing message passing over relational nodes, we can naturally capture relation patterns as entity-independent contextual information, which is essential for inductive KG completion.

In this work, we train two message-passing GNNs, Ω and Ψ: Ω is applied to the k-hop ego graph to extract logical evidence, while Ψ is applied to the *relation network* to learn relational embeddings for inductive inference.

Mutual information maximization. Reasoning with logical evidence is pivotal for inductive KG completion, as it emulates human-like inference. However, as mentioned earlier, logical evidence is challenging to represent and has typically been used as a regularization mechanism in prior work. To address this, we propose *Logic Evidence Information Maximization* (LEIM), which aims to maximize the mutual information between the logical evidence extracted from k-hop ego graphs and the relational semantics of the center node. The optimization objective is defined as:

$$\mathcal{L}_{\omega,\psi} = -\mathcal{I}_{\omega}(\Lambda(\mathcal{G}), \Psi^{k}(\widetilde{\mathcal{G}})), \tag{5.3}$$

where $\Lambda(\mathcal{G}) = \Omega_{GNN}(\Lambda^{k}(v) \mid v \in \widetilde{\mathcal{G}})$ is the logical evidence extracted by Ω, $\mathcal{I}(\cdot,\cdot)$ is the mutual information, $\Psi^{k}(\cdot)$ is a k-layer GNN, and ω, ψ are the trainable parameters for $\mathcal{I}$ and Ψ, respectively. The objective minimizes the loss to find the optimal parameters $\hat{\psi}$ that preserve the mutual information between the logical evidence $\Lambda(\mathcal{G})$ and the relational embedding of the target node v learned by $\Psi^{k}(\widetilde{\mathcal{G}})$.

In practice, we adopt a Jensen-Shannon MI estimator (JSD) to estimate the mutual

information:

$$\mathcal{I}^{JSD}_{\omega,\psi}(\Lambda,\Psi) = \mathbb{E}_{\mathbb{P}}\Big(- sp(-T_{\omega,\psi}(\Lambda^k(v),\Psi^k(v)))\Big) \\ - \mathbb{E}_{\mathbb{P}\times\mathbb{P}'}\Big(- sp(T_{\omega,\psi}(\Lambda^k(v'),\Psi^k(v)))\Big), \quad (5.4)$$

where v is a node sampled from the relation network under distribution $\mathbb{P}$, v' is sampled from a negative distribution $\mathbb{P}' = \mathbb{P}$, and sp represents the softplus activation function. To encourage alignment of positive pairs, e.g., $(\Lambda(v), \Psi(v))$, and distinguish them from negative pairs, e.g., $(\Lambda(v'), \Psi(v))$ where $v \neq v'$, we use $T_{\omega,\psi}(\Lambda, \Psi)$ to predict the correlation between Λ and Ψ as follows:

$$T_{\omega,\psi}(\Lambda^k(v), \boldsymbol{x}_v) = \sum_{(p,q)\in\Lambda^k(v)} \log(f_\omega([\boldsymbol{h}_p||\boldsymbol{h}_q||\boldsymbol{x}_v])), \quad (5.5)$$

where $\boldsymbol{h}_p$ and $\boldsymbol{h}_q$ are the features of source and target nodes p and q obtained from the k-hop logical ego graph $\Lambda^k(\mathcal{G})$, and $\boldsymbol{x}_v = \Psi^k(v)$ is the relational embedding of node v. $f : \mathbb{R}^{|\boldsymbol{h}|+|\boldsymbol{h}|+|\boldsymbol{x}|} \to \mathbb{R}^{(0,1)}$ is a fully connected discriminator used to differentiate positive and negative pairs.

5.2.3 Training Scheme and KG Inference

Training objective. This section outlines the complete algorithm for *NORAN* along with the training procedure. After training, we obtain the model $\Psi \circ \Gamma$, which is used to infer representations for inductive triples. Given the relational embedding learned for an inductive triple via $\Psi \circ \Gamma$, we perform link prediction using a classifier based on *logistic regression*, defined as $p(\boldsymbol{x}) = Sigmoid(\boldsymbol{w}^T\boldsymbol{x} + b)$.

Inductive inference and complexity analysis. For inductive inference, given a triple $t = (h, r, t)$ with an unseen entity h or t, we first create a new node $v = \Gamma(h, r, t)$ and update the original relation network $\widetilde{\mathcal{G}}$ by adding node v to it. The probability of triple existence is then predicted using $p \circ \Psi \circ \Gamma$.

Unlike the *Embedding-based Paradigm*, which requires retraining for inductive inference, our framework only updates the relation network and performs inference with the trained model $p \circ \Psi \circ \Gamma$. The time complexity of inference in our framework is:

$$\mathcal{O}(\underbrace{3bf^2}_{\Gamma} + \underbrace{bd^L f + bLf^2}_{\Psi}),$$

where b is the number of inductive triples, f is the hidden dimension (assuming all embeddings have the same dimension), d is the average node degree in relation network $\widetilde{\mathcal{G}}$, and L is the number of layers in Ψ. The time complexity of p and relation network updates is negligible compared to the rest. Thus, the primary computational cost arises from the message-passing operation in Ψ, which grows exponentially with L. However, empirical results indicate that a two-layer GNN typically suffices, as the local view of GNNs is naturally suited to this task. This aligns with existing research advocating for shallower GNNs.

5.3 Experiments

This section provides an empirical evaluation of the proposed framework and its performance across five KG datasets. The study is guided by the following research questions:

- ***Q1*** (5.3.2): How does our proposed framework compare against the strongest baselines, including traditional embedding-based, rule-based, and other message-passing (MP) methods?
- ***Q2*** (5.3.3): With respect to Remark 1 and Remark 2, does our proposed *relation network* effectively model inductive KG completion?
- ***Q3*** (5.3.4). Is our proposed *logic evidence information maximization* a more effective training objective than standard negative sampling for inductive KGC?

Table 5.2: Main results of inductive KGC on five benchmark datasets. We underline the best results within each of the three categories and bold NORAN's results that are better than all baselines.

Categories	Datasets	FB15K-237			WN18RR			NELL995			OGBL-WIKIKG2			OGBL-BIOKG		
	Metrics	MRR	Hit@1	Hit@3	MRR	Hit@1	Hit@3	MRR	Hit@1	Hit@3	MRR	Hit@1	Hit@3	MRR	Hit@1	Hit@3
Emb. Based	TransE	0.289	0.198	0.324	0.265	0.058	0.445	0.254	0.169	0.271	0.213	0.122	0.229	0.317	0.26	0.345
	DistMult	0.241	0.155	0.263	0.430	0.390	0.440	0.267	0.174	0.295	0.199	0.115	0.210	0.341	0.278	0.363
	ComplEx	0.247	0.158	0.275	0.440	0.410	0.460	0.227	0.149	0.249	0.236	0.136	0.254	0.322	0.257	0.360
	SimplE	0.338	0.241	0.375	0.476	0.428	0.492	0.291	0.198	0.314	0.220	0.131	0.239	0.319	0.246	0.358
	QuatE	0.319	0.241	0.358	0.446	0.382	0.478	0.285	0.201	0.307	0.248	0.139	0.262	0.363	0.294	0.382
Rule Based	RuleN	0.453	0.387	0.491	0.514	0.461	0.532	0.346	0.279	0.366	-	-	-	-	-	-
	DRUM	0.447	0.373	0.478	0.521	0.458	0.549	0.340	0.261	0.363	-	-	-	-	-	-
MP based	RGCN	0.427	0.367	0.451	0.501	0.458	0.519	0.329	0.256	0.348	0.285	0.176	0.324	0.381	0.319	0.399
	RGHAT	0.440	0.361	0.483	0.518	0.460	0.540	0.337	0.274	0.351	0.301	0.192	0.329	0.395	0.334	0.418
	GraIL	0.465	0.389	0.482	0.512	0.453	0.539	0.355	0.282	0.367	0.327	0.201	0.336	0.434	0.379	0.451
	ConGLR	0.463	0.402	0.483	0.512	0.452	0.541	0.352	0.276	0.366	0.318	0.219	0.338	0.422	0.365	0.431
	PATHCON	0.483	0.425	0.499	0.522	0.462	0.546	0.349	0.276	0.369	0.339	0.243	0.347	0.457	0.395	0.472
	RMPI	0.459	0.396	0.480	0.514	0.454	0.544	0.339	0.277	0.365	0.313	0.213	0.336	0.414	0.358	0.421
	MBE	0.477	0.410	0.495	0.519	0.451	0.549	0.344	0.270	0.359	0.331	0.234	0.345	0.452	0.380	0.470
Ours[a]	NORAN(GS)	**0.483**	**0.440**	**0.504**	**0.535**	**0.471**	**0.564**	**0.364**	**0.298**	**0.381**	**0.349**	0.237	**0.361**	0.454	**0.395**	**0.479**
	NORAN(GIN)	**0.499**	**0.451**	**0.519**	**0.530**	**0.467**	**0.560**	**0.370**	**0.308**	**0.379**	**0.353**	**0.251**	**0.367**	**0.469**	**0.407**	**0.492**
	NORAN(GAT)	0.468	0.422	0.489	**0.540**	**0.499**	**0.575**	**0.374**	**0.310**	**0.392**	**0.358**	**0.260**	**0.371**	**0.475**	**0.411**	**0.495**
NORAN - Emb.[b]	+2.2% (Avg.)	+1.6%	+2.6%	+2.0%	+1.8%	+3.7%	+2.9%	+1.9%	+2.8%	+2.3%	+1.9%	+1.7%	+2.4%	+1.8%	+1.6%	+2.3%
NORAN - GNN[b]	+11.2% (Avg.)	+16.1%	+21.0 %	+14.4%	+6.4%	+7.1%	+8.3%	+8.3%	+10.9%	+7.8%	+11.0 %	+12.1%	+10.9%	+11.2%	+11.7%	+11.5%

[a] We test our NORAN with three backbones: GraphSAGE, GIN, and GAT, denoted as NORAN(GS), NORAN(GIN), NORAN(GAT), respectively.

[b] We show the margin between the best results of NORAN and the ones of the two branches of baselines methods.

5.3.1 Experimental Setup

Datasets. We evaluate our framework on five widely-used KG benchmarks: (*i*) **FB15K-237**, a subset of FB15K with inverse relations removed; (*ii*) **WN18RR**, a subset of WN18 with inverse relations excluded; (*iii*) **NELL995**, derived from the 995-th iteration of the NELL system and containing general knowledge; (*iv*) **OGBL-WIKIKG2**, extracted from the Wikidata knowledge base; and (*v*) **OGBL-BIOKG**, constructed from biomedical repositories.

Baselines. To validate the effectiveness of NORAN, we compare it against state-of-the-art baselines. These baselines are categorized into three groups: (*i*) *embedding-based* methods, including **TransE**, **DistMult**, **ComplEx**, **SimplE**, and **QuatE**; (*ii*) *rule-based* methods, such as **RuleN** and **DRUM**; and (*iii*) *MP-based* methods, including **RGCN**, **RGHAT**, **GraIL**, **ConGLR**, **PATHCON**, **RMPI**, and **MBE**.

Notably, the embedding-based methods are not inherently inductive, so we retrain their embedding tables for inductive inference. For MP-based methods, we select the most effective ones for KGC in the same setting. We exclude certain related works from our experiments as their original implementations are tailored for KGC with unseen relations, whereas our model focuses on KGC with unseen entities.

Implementation Details. The five benchmark datasets were originally designed for the *transductive setting*, where test entities are subsets of training entities. To adapt them for inductive testing, we create new inductive datasets by sampling disjoint subgraphs from the original KGs. Specifically, we randomly select a subset of entities from the test set, remove them and their associated edges from the training set, and use the remaining training set for model training. The removed edges are reintroduced during evaluation. We use **MRR** (mean reciprocal rank) and **Hit@1, 3** as evaluation metrics.

All baseline methods are implemented using their open-source codebases. The models are implemented in PyTorch and trained on an RTX 3090 GPU with 24 GB RAM. For fairness, the embedding size for entities and relations is set to 100 for the input, latent, and output layers across all models, except for PATHCON, which uses dataset-specific node labeling for its input embedding size. We optimize all models using the Adam optimizer with a batch size of 256, except for the larger OGBL datasets, where the batch size is reduced to 32 to avoid memory issues. Model parameters are initialized using the Xavier initializer, with an initial learning rate of 0.005. Hyperparameters are tuned via grid search, with the learning rate selected from $\{0.05, 0.01, 0.005, 0.001\}$ and the margin parameter for negative sampling ranging from 0 to 1. The best configurations are used as defaults for other hyperparameters. To ensure reproducibility, we use a fixed random seed and report the average results of three runs.

Table 5.3: Results of ablation study for relation network construction rules on three benchmark datasets.

Construction Rule	Datasets	WN18RR			NELL995			OGBL-WIKIKG2		
	Metrics	MRR	Hit@1	Hit@3	MRR	Hit@1	Hit@3	MRR	Hit@1	Hit@3
Default	NORAN(GAT)	0.540	0.499	0.575	0.374	0.310	0.392	0.358	0.260	0.371
w.o. head-head	NORAN(GAT)	0.521	0.457	0.558	0.348	0.273	0.375	0.329	0.230	0.341
w.o. tail-tail	NORAN(GAT)	0.517	0.454	0.549	0.351	0.288	0.364	0.327	0.224	0.342
w.o. head-tail	NORAN(GAT)	0.501	0.446	0.532	0.339	0.275	0.359	0.314	0.212	0.319

5.3.2 Main Results: Q1

To address Q1, we conduct extensive experiments comparing NORAN with the strongest baselines. The results are summarized in Table 5.2, with key observations outlined below.

Obs. 1. NORAN significantly outperforms state-of-the-art KGC baselines. We compare NORAN against 5 embedding-based, 2 rule-based, and 7 MP-based methods on 5 KG datasets. As shown in Table 5.2, NORAN achieves superior performance across all evaluation metrics (MRR, Hit@1, Hit@3). Results for NORAN are **bolded** when they outperform all baselines. We evaluate NORAN with three backbones, and the bolded results dominate, highlighting the framework's effectiveness. Additionally, we compute the margin between NORAN's best results (underlined) and those of the baselines. NORAN outperforms the best embedding-based and MP-based baselines by an average margin of 11.2% and 2.2%, respectively.

Obs. 2. NORAN with GAT (learnable convolution matrix) generally outperforms MP layers with fixed convolution matrices. We test NORAN with three backbones: GAT, GraphSAGE, and GIN. GAT, which uses a learnable convolution matrix, achieves the best results (underlined) on four datasets, while GraphSAGE and GIN exhibit comparable performance.

Table 5.4: Ablation study of training objective, i.e., *Logic Evidence Information Maximization* (LEIM)

Training Objective	Datasets	WN18RR			NELL995			OGBL-WIKIKG2		
	Metrics	MRR	Hit@1	Hit@3	MRR	Hit@1	Hit@3	MRR	Hit@1	Hit@3
Naive Negative Sampling	NORAN(GS)	0.529	0.469	0.554	0.351	0.290	0.357	0.324	0.215	0.322
	NORAN(GIN)	0.537	0.490	0.567	0.357	0.291	0.374	0.335	0.231	0.349
	NORAN(GAT)	0.534	0.481	0.562	0.361	0.299	0.377	0.342	0.238	0.355
LEIM (JSD)	NORAN(GS)	0.535	0.471	0.564	0.364	0.298	0.381	0.349	0.237	0.361
	NORAN(GIN)	0.530	0.467	0.560	0.370	0.308	0.379	0.353	0.251	0.367
	NORAN(GAT)	0.540	0.499	0.575	0.374	0.310	0.392	0.358	0.260	0.371
LEIM (InfoNCE)	NORAN(GS)	0.532	0.476	0.558	0.358	0.294	0.369	0.331	0.220	0.343
	NORAN(GIN)	0.541	0.495	0.583	0.365	0.301	0.376	0.339	0.231	0.359
	NORAN(GAT)	0.544	0.506	0.581	0.367	0.306	0.383	0.337	0.232	0.352

5.3.3 Ablation Study on Relation Network: Q2

The relation network's construction involves two key aspects: (i) the semantics of various '*entity sharing*' patterns (Remark 1) and (ii) the semantics of '*linking direction*' (Remark 2). We conduct an ablation study to validate these aspects.

Remark 1. ***Linking pattern***. *For any two triples sharing entities, i.e.,* $T_1 = (h_1, r_1, t_1) \cap T_2 = (h_2, r_2, t_2)$*, there are three linking patterns:* (i) *head-head sharing* $(h_1 = h_2)$*,* (ii) *tail-tail sharing* $(t_1 = t_2)$*, and* (iii) *head-tail sharing* $(t_1 = h_2 \oplus t_2 = h_1)$*. Our construction criterion links all three patterns, as they carry distinct semantic meanings.*

Remark 2. ***Linking direction***. *As per Def. 2, edges in the relation network are bidirectional, regardless of the linking pattern. This design is based on the rationale that relation semantics in KGs are not direction-sensitive. For example, the triple (Elon Musk, :WorkIn, TESLA) in Figure 5.1 (i) is semantically equivalent to (TESLA, :Foundedby, Elon Musk).*

To address Q2, we perform an ablation study on the relation network's construction. For Remark 1, we iteratively remove each linking pattern (head-head, head-tail, tail-

tail) and evaluate the resulting network using GAT as the backbone. The results, shown in Table 5.3, demonstrate that removing any pattern significantly degrades NORAN's performance, validating our construction approach.

Obs. 3. The head-tail pattern is the most critical for relation network construction. Ablating the head-tail pattern results in a larger performance drop compared to the other patterns. This pattern aligns with translation-based logic rules, which are central to multi-hop KG reasoning. In contrast, head-head and tail-tail patterns correspond to union and intersection operations, respectively, as defined in complex KG reasoning. While these operations also contribute to KGC, they are often overlooked by translation-based embedding methods, explaining their weaker inductive performance in Table 5.2.

5.3.4 Ablation Study for Training Objective: Q3

To address Q3, we evaluate the effectiveness of LEIM by comparing it with an InfoNCE estimator and naive negative sampling (NS). The InfoNCE estimator is defined as:

$$\mathcal{I}_{\omega,\psi}^{InfoNCE}(\Lambda,\Psi) = \mathbb{E}_{\mathbb{P}}\Bigg(T_{\omega,\psi}(\Lambda^k(v),\Psi^k(v)) - \mathbb{E}_{\mathbb{P}'}\Big(\log \sum_{v'\sim\mathbb{P}'} e^{T_{\omega,\psi}(\Lambda^k(v'),\Psi^k(v))}\Big)\Bigg). \quad (5.6)$$

We evaluate LEIM using both JSD (default) and InfoNCE estimators and compare them to naive NS. Results are shown in Table 5.4, with the best results bolded by column. LEIM consistently outperforms naive NS across all backbones, demonstrating its effectiveness. The key difference between JSD and InfoNCE lies in their negative sampling strategies. JSD uses a $1:1$ negative sampling ratio within a batch, while InfoNCE employs a contrastive learning approach, using all other instances in the batch as negative samples. Given the computational efficiency and empirical results, we adopt JSD as the default. Additionally, we observe:

Obs. 4. GAT benefits the most from LEIM. On three datasets, GAT achieves an average performance improvement of 1.2% in MRR and 1.5% in Hit@3 when trained with LEIM, outperforming other backbones. Notably, on WN18RR, GAT transitions from the worst-performing backbone under naive NS to the best-performing one under LEIM. In contrast, GIN, which uses a fixed convolution matrix, shows minimal or negative improvement with LEIM.

5.4 Summary

Inductive KG completion, which infers relations for newly introduced entities, aligns with real-world KGs that continuously evolve. In this work, we propose NORAN, a novel message-passing framework that leverages latent relation semantics for inductive KG completion. Our framework introduces a unique perspective on KG modeling through the *relation network*, which treats relations as indicators of logical rules. Additionally, we propose *logic evidence information maximization* (LEIM), an innovative training objective that preserves logical semantics in KGs. NORAN combines the relation network and LEIM to enable effective inductive KGC. Extensive experiments on five KG benchmarks demonstrate NORAN's superiority over state-of-the-art methods.

Chapter 6

Knowledge Graph Prompting for Large Language Models

6.1 Introduction

Large language models (LLMs), such as the Clude [4] and GPT series [73], have demonstrated exceptional capabilities across diverse tasks, achieving breakthroughs in text comprehension [11], question answering [50], and content generation [21]. Despite their success, LLMs face persistent criticism for their limitations in knowledge-intensive tasks, particularly those requiring domain expertise [115]. Their application in specialized domains remains challenging due to three key factors: (i) Knowledge limitations: LLMs possess broad but superficial knowledge in specialized fields, as their training data primarily consists of general-domain content, leading to gaps in domain-specific depth and alignment with current professional standards. (ii) Reasoning complexity: Specialized domains demand precise, multi-step reasoning governed by domain-specific rules and constraints. LLMs often struggle to maintain logical consistency and accuracy throughout extended reasoning chains, especially in technical or highly regulated contexts. (iii) Context sensitivity: Professional fields rely on

nuanced, context-dependent interpretations where identical terms or concepts may carry different meanings based on specific scenarios. LLMs frequently fail to grasp these subtleties, resulting in misinterpretations or overly generalized responses.

To adapt LLMs for specific domains, researchers have explored various approaches, which can be broadly classified into two categories: (i) Fine-tuning LLMs with Domain-specific Data. Fine-tuning pre-trained LLMs on specialized datasets enables them to better capture domain-specific vocabulary, terminology, and patterns [34, 53, 54, 31]. This approach has been successfully applied in areas such as recommendation [17, 123] and node classification [16, 40], enhancing the relevance and accuracy of generated responses [41]. Fine-tuned LLMs have demonstrated effectiveness across various domains. In healthcare, they have been leveraged for clinical note analysis [3], biomedical text mining [54], and medical dialogue [91]. Similarly, in the legal domain, they have proven useful for legal document classification [13], contract analysis [14], and legal judgment prediction [122]. (ii) Retrieval-augmented generation (RAG). RAG provides an effective way to tailor LLMs for specialized domains without modifying the model architecture or parameters [56]. Instead of embedding new knowledge through retraining, RAG dynamically retrieves relevant domain-specific information from external sources, enhancing response accuracy and reliability. A typical RAG system operates in three stages: knowledge preparation, retrieval, and integration. First, external textual data is segmented into manageable chunks and transformed into vector representations for efficient indexing. During retrieval, relevant chunks are identified based on keyword matching or vector similarity when a query is submitted. Finally, the retrieved information is combined with the original query to generate well-informed responses.

Despite their success, RAG systems face significant challenges in practical applications due to the inconsistent quality of accessible data. Domain knowledge is frequently distributed across diverse sources—ranging from textbooks and research articles to technical manuals and industry reports [57]—which may vary in quality, accuracy, and

completeness, potentially leading to discrepancies in the retrieved information [124]. A promising strategy to mitigate these issues is to integrate Knowledge Graphs (KGs) with LLMs. KGs offer a structured representation of domain knowledge, built on well-defined ontologies that specify specialized terminologies, acronyms, and their interrelations within a field [58, 80, 116, 117, 118]. The extensive factual content contained in KGs can help anchor model responses in established facts and principles [39, 109, 74].

6.2 Introduction

Earlier research explored heuristic methods to integrate knowledge from KGs into LLMs during pre-training or fine-tuning. For instance, ERNIE [82] aligns entity embeddings with word embeddings during pre-training to enhance entity understanding and reasoning. Similarly, KnowBERT [77] integrates entity linkers with BERT to inject entity knowledge during fine-tuning for knowledge-intensive tasks. Another approach involves retrieving relevant knowledge from KGs at inference time to augment the LLM's context. For example, K-BERT [62] uses an attention mechanism to select relevant triples from KGs based on the query context, appending them to the input sequence. KEPLER [99] learns joint embeddings of text and KG entities to improve model predictions. Subsequent works have combined graph neural networks with LLMs for joint reasoning [111, 119, 23] and introduced interactions between text tokens and KG entities within LLM layers [83, 88].

However, as LLMs continue to evolve, most state-of-the-art models remain *closed-source*. For example, GPT-4 [73] and Claude 3 [4] are accessible only through APIs, limiting access to model internals. Consequently, research has shifted toward KG prompting, which enhances fixed LLMs with KG-based hard prompts [61, 74]. KG prompting has emerged as a new paradigm in natural language processing. For instance, CoK [95] introduces Chain-of-Knowledge prompting, decomposing LLM-generated reasoning chains into evidence triples and verifying their accuracy using

external KGs. Mindmap [101] enhances transparency by enabling LLMs to reason over structured KG inputs. RoG [66] proposes a planning-retrieval-reasoning framework that synergizes LLMs and KGs for interpretable reasoning. KGR [32] autonomously refines LLM-generated responses by leveraging KGs to extract, verify, and refine factual information, mitigating hallucinations in knowledge-intensive tasks.

Despite the promising results of existing KG prompting methods, three critical challenges hinder their practical application. ❶ Vast search space. Real-world KGs often contain millions of triples, creating a large search space when retrieving relevant knowledge for prompting. ❷ High API costs. Accessing closed-source LLMs like GPT-4 and Claude 3 through proprietary APIs can be prohibitively expensive at scale [25], necessitating careful selection of the most informative knowledge. ❸ Labor-intensive prompt design. LLMs are highly sensitive to prompts, with minor variations potentially yielding drastically different responses. Existing methods rely on manually designed or rule-based prompts, which are inflexible and lack adaptability to varying question semantics and KG structures.

To address these challenges, we propose **Know**ledge **G**raph-based **P**romp**T**ing, or **KnowGPT**, a framework that leverages factual knowledge from KGs to ground LLM responses in established facts. This paper addresses two key research questions: ❶ Given a query and a large-scale KG, how can we efficiently retrieve relevant factual knowledge? ❷ Given the extracted knowledge, how can we construct an effective prompt for LLMs?

We address these questions with a novel prompt learning method that is effective, generalizable, and cost-efficient. For question ❶, we employ deep reinforcement learning (RL) to extract the most informative knowledge from KGs. A tailored reward scheme encourages the agent to discover concise, context-relevant knowledge chains. For question ❷, we introduce a prompt construction strategy based on Multi-Armed Bandit (MAB). Given multiple knowledge extraction strategies and prompt templates, MAB selects the most effective combination for each question by balancing exploration and

exploitation.

Our contributions are summarized as follows:

- We formally define the problem of KG-based prompting, which leverages structured knowledge from KGs to ground LLM responses in established facts.
- We propose *KnowGPT*, a novel prompting framework that combines deep RL and MAB to generate effective prompts for domain-specific queries.
- We implement *KnowGPT* on GPT-3.5. Experiments on three QA datasets demonstrate that KnowGPT outperforms state-of-the-art baselines by a significant margin. Notably, *KnowGPT* achieves an average improvement of 23.7% over GPT-3.5 and 2.9% over GPT-4, with 92.6% accuracy on the OpenbookQA leaderboard, comparable to human performance.

6.3 KnowGPT Framework

Learning the prompting function $f_{\text{prompt}}(\mathcal{Q}, \mathcal{G})$ involves two challenges: determining what knowledge to use from $\mathcal{G}$ and how to construct the prompt. KnowGPT addresses these challenges by extracting knowledge with deep RL and constructing prompts with MAB. An overview of the framework is shown in Figure 6.1.

6.3.1 Knowledge Extraction with Reinforcement Learning

Intuitively, the relevant reasoning background lies in a question-specific subgraph $\mathcal{G}_{\text{sub}}$ containing all *source* entities $\mathcal{Q}_s$, *target* entities $\mathcal{Q}_t$, and their neighbors. An ideal subgraph $\mathcal{G}_{\text{sub}}$ should: (i) include as many source and target entities as possible, (ii) exhibit strong relevance to the question context, and (iii) be concise to fit within LLM input length constraints.

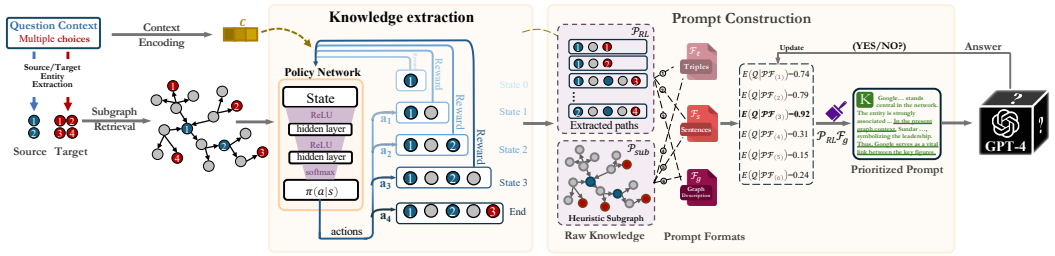


Figure 6.1: The overall architecture of our proposed knowledge graph prompting framework, i.e., KnowGPT. Given the question context with multiple choices, we first retrieve a question-specific subgraph from the real-world KG. *Knowledge Extraction* is first dedicated to searching for the most informative and concise reasoning background. Then the *Prompt Construction* module is optimized to prioritize the combination of knowledge and formats subject to the given question.

However, extracting such a subgraph is NP-hard. To address this, we develop $\mathcal{P}_{\text{RL}}$, a deep RL-based method that samples reasoning chains in a trial-and-error fashion. We assume $\mathcal{G}_{\text{sub}}$ is constructed from a set of reasoning chains $\mathcal{P} = \{\mathcal{P}_1, \mathcal{P}_2, ..., \mathcal{P}_m\}$, where each chain $\mathcal{P}_i = \{(e_i, r_1, t_1), (t_1, r_2, t_2), ..., (t_{|\mathcal{P}_i|-1}, r_{|\mathcal{P}_i|}, t_{|\mathcal{P}_i|})\}$ is a path in $\mathcal{G}$ starting from the *i-th* source entity in $\mathcal{Q}_s$, and $|\mathcal{P}_i|$ is the path length. $\mathcal{G}_{\text{sub}}$ includes all entities and relations in $\mathcal{P}$.

- **State:** A state represents the current entity in the KG. Specifically, it captures the spatial change from entity h to t. Following prior work, we define the state vector $\boldsymbol{s}$ as:

$$\boldsymbol{s}_t = (\boldsymbol{e}_t, \boldsymbol{e}_{target} - \boldsymbol{e}_t), \tag{6.1}$$

  where $\boldsymbol{e}_t$ and $\boldsymbol{e}_{target}$ are embeddings of the current and target entities. Initial node embeddings are obtained by transforming KG triples into sentences and feeding them into a pre-trained LM.

- **Action:** The action space includes all neighboring entities of the current entity, enabling flexible exploration of the KG.

- **Transition:** The transition model P measures the probability of moving to a new state (s') given the current state (s) and action (a). In KGs, $\text{P}(s'|s, a) = 1$

if s is connected to s' via a; otherwise, $\mathrm{P}(s'|s,a) = 0$.

- **Reward:** The reward is based on reachability:

$$r_{reach} = \begin{cases} +1, & \textit{if target}; \\ -1, & \textit{otherwise}, \end{cases} \tag{6.2}$$

indicating whether the path reaches the target within K steps.

To promote context-relatedness and conciseness, we design two auxiliary rewards.

Context-relatedness Auxiliary Reward. This reward encourages paths closely related to the question context. We evaluate the semantic relevance of a path $\mathcal{P}_i$ to the context $\mathcal{Q}$ using a fixed matrix $\boldsymbol{W}$ to map the path embedding $\boldsymbol{\mathcal{P}}$ to the same semantic space as the context embedding $\boldsymbol{c}$. The reward is:

$$r_{\mathrm{cr}} = \frac{1}{|i|} \sum_{source}^{i} cos(\boldsymbol{W} \times \boldsymbol{\mathcal{P}}_i, \boldsymbol{c}), \tag{6.3}$$

where $\boldsymbol{c}$ is the context embedding from a pre-trained LM [49], and $\boldsymbol{\mathcal{P}}_i$ is the average embedding of entities and relations in the path.

Conciseness Auxiliary Reward. This reward encourages concise paths to fit within LLM input constraints and reduce API costs. The reward for path $\mathcal{P}_i$ is:

$$r_{\mathrm{cs}} = \frac{1}{|\mathcal{P}_i|}. \tag{6.4}$$

Training Policy Network. We train a policy network $\pi_\theta(s,a) = p(a|s;\theta)$ using policy gradient [105] to maximize accumulated rewards. The optimal policy navigates from the source to the target entity while maximizing rewards. Additional training details are provided in the Appendix.

6.3.2 Prompt Construction with Multi-armed Bandit

We design a prompt construction strategy using Multi-Armed Bandit (MAB) to select the best combination of knowledge extraction strategies and prompt templates. Suppose we have knowledge extraction strategies $\{\mathcal{P}_1, \mathcal{P}_2, ..., \mathcal{P}_m\}$ and prompt templates $\mathcal{F} = \{\mathcal{F}_1, \mathcal{F}_2, ..., \mathcal{F}_n\}$. The goal is to identify the best combination for a given question.

We define the selection process as a reward maximization problem:

$$\sigma(f(\mathcal{PF}_{(i)})) = \begin{cases} 1 & \textit{if accurate;} \\ 0 & \textit{otherwise.} \end{cases} \tag{6.5}$$

Here, $\mathcal{PF}_{(i)}$ is a combination of extraction strategy and prompt template, and $r_{pf} \in \{0, 1\}$ indicates the LLM's performance.

We formulate the selection mechanism with an expectation function $E(\cdot)$:

$$E(\mathcal{Q}|\mathcal{PF}_{(i)}) = \boldsymbol{c} \times \boldsymbol{\alpha}_{(i)} + \beta_{(i)}. \tag{6.6}$$

Here, $\boldsymbol{c}$ is the context embedding, $\boldsymbol{\alpha}(i)$ are learned parameters, and $\beta_{(i)}$ introduces noise for exploration.

We update $\boldsymbol{\alpha}_{(i)}$ using:

$$\begin{aligned} J(\mathbf{C}_{(i)}^{(k)}, \mathbf{r}_{pf}^{(i)(k)}) &= \sum_{k=1}^{K} (\mathbf{r}_{pf}^{(i)(k)} - \mathbf{C}_{(i)}^{(k)} \boldsymbol{\alpha}^{(i)})^2 + \lambda^i \parallel \boldsymbol{\alpha}^{(i)} \parallel_2^2 . \\ \rightarrow \boldsymbol{\alpha}^{(i)} &= \left((\mathbf{C}_{(i)}^{(k)})^\top \mathbf{C}_{(i)}^{(k)} + \lambda^i \mathbf{I} \right)^{-1} (\mathbf{C}_{(i)}^{(k)})^\top \mathbf{r}_{pf}^{(i)(k)}. \end{aligned} \tag{6.7}$$

The exploration term $\beta^{(i)}$ is:

$$\beta^{(i)} = \quad \gamma \times \sqrt{\mathbf{c}_i \left((\mathbf{C}_{(i)}^{(k)})^\top \mathbf{C}_{(i)}^{(k)} + \lambda^i \mathbf{I} \right)^{-1} (\mathbf{c}_{(i)})^\top}, \tag{6.8}$$

where γ is a fixed constant.

Implementation. We implement the aforementioned Multi-Armed Bandit (MAB) strategies using two distinct knowledge extraction approaches and three prompt templates. The MAB framework is flexible, enabling integration with additional knowledge extraction methods and prompt templates to optimize performance. The two knowledge extraction strategies we employ are:

- $\mathcal{P}_{\text{RL}}$: This strategy utilizes reinforcement learning (RL)-based extraction, as detailed in the previous subsection.
- $\mathcal{P}_{\text{sub}}$: This heuristic sub-graph extraction method focuses on gathering a 2-hop subgraph around both the source and target entities. Given the inherent instability of RL approaches, we introduce $\mathcal{P}_{\text{sub}}$ as a backup strategy in the MAB selection process, ensuring robustness if the RL-based method underperforms.

For the prompt templates, we incorporate the following:

- **Triples**, represented as $\mathcal{F}_t$, correspond to the extracted knowledge in its original triple form, such as (*Sergey_Brin*, *founder_of*, *Google*), (*Sundar_Pichai*, *ceo_of*, *Google*), (*Google*, *is_a*, *High-tech Company*). This representation has been empirically shown to be comprehensible by black-box LLMs.
- **Sentences** serve as a transformation method to reframe the knowledge into natural language, $\mathcal{F}_s$, such as: "*Sergey Brin, the founder of Google, a high-tech company, has handed over the reins to Sundar Pichai, the current CEO of the company.*"
- **Graph Description**, $\mathcal{F}_g$ prompts the LLM by treating the knowledge as a structured graph. This preprocessing is done via the LLM itself to generate descriptions that emphasize key entities, such as: "*Google, a leading high-tech company, stands central in the network. The entity has strong ties to significant figures in the tech industry. Sergey Brin, one of the founders, created Google,*

marking its historical foundation. Today, Sundar Pichai is the CEO, symbolizing its present leadership. Consequently, Google serves as a pivotal link among these key figures."

Given the two knowledge extraction methods: $\mathcal{P}_{\text{sub}}$ and $\mathcal{P}_{\text{RL}}$, along with the three prompt translation methods: $\mathcal{F}_t$, $\mathcal{F}_s$, and $\mathcal{F}_g$, the MAB is trained to prioritize the most suitable combination of extraction method and prompt format based on feedback from the LLMs. The goal is to select the best pairing for different real-world question scenarios, i.e., $\mathcal{PF} = \{(\mathcal{P}_{sub}\mathcal{F}_t), (\mathcal{P}_{sub}\mathcal{F}_s), (\mathcal{P}_{sub}\mathcal{F}_g), (\mathcal{P}_{RL}\mathcal{F}_t), (\mathcal{P}_{RL}\mathcal{F}_s), (\mathcal{P}_{RL}\mathcal{F}_g)\}$.

6.4 Experiments

In this section, we perform comprehensive experiments to assess the effectiveness of `KnowGPT` on three widely used question-answering datasets, spanning both commonsense reasoning and domain-specific tasks. The implementation of `KnowGPT` is based on GPT-3.5. Our experiments are designed to address the following research questions:

- **RQ1 (Main results)**: How does `KnowGPT` compare to the current state-of-the-art large language models (LLMs) and knowledge graph-enhanced question answering (QA) baselines?
- **RQ2 (Ablation Study)**: What contribution does each key component of `KnowGPT` make to the overall performance?
- **RQ3 (Case study)**: How does knowledge graph integration enhance the handling of complex reasoning tasks?

Table 6.1: Performance comparison among baseline models and KnowGPT.

Catagory	Model	CommonsenseQA		OpenBookQA		MedQA	
		IHdev-Acc.	IHtest-Acc.	Dev-Acc.	Test-Acc.	Dev-Acc.	Test-Acc.
LM + Fine-tuning	Bert-base	0.573	0.535	0.588	0.566	0.359	0.344
	Bert-large	0.611	0.554	0.626	0.602	0.373	0.367
	RoBerta-large	0.731	0.687	0.668	0.648	0.369	0.361
KG-enhanced LM	MHGRN	0.745	0.713	0.786	0.806	-	-
	QA-GNN	0.765	0.733	0.836	0.828	0.394	0.381
	HamQA	0.769	0.739	0.858	0.846	0.396	0.385
	JointLK	0.777	0.744	0.864	0.856	0.411	0.403
	GreaseLM	0.785	0.742	0.857	0.848	0.400	0.385
	GrapeQA	0.782	0.749	0.849	0.824	0.401	0.395
LLM + Zero-shot	ChatGLM	0.473	0.469	0.352	0.360	0.346	0.366
	ChatGLM2	0.440	0.425	0.392	0.386	0.432	0.422
	Baichuan-7B	0.491	0.476	0.411	0.395	0.334	0.319
	InternLM	0.477	0.454	0.376	0.406	0.325	0.348
	Llama2 (7b)	0.564	0.546	0.524	0.467	0.338	0.340
	Llama3 (8b)	0.745	0.723	0.771	0.730	0.639	0.697
	GPT-3	0.539	0.520	0.420	0.482	0.312	0.289
	GPT-3.5	0.735	0.710	0.598	0.600	0.484	0.487
	GPT-4	0.776	0.786	0.878	0.910	0.739	0.763
LLM + KG Prompting	CoK	0.759	0.739	0.835	0.869	0.706	0.722
	RoG	0.750	0.734	0.823	0.861	0.713	0.726
	Mindmap	0.789	0.784	0.851	0.882	0.747	0.751
Ours	KnowGPT	**0.827**	**0.818**	**0.900**	**0.924**	**0.776**	**0.781**
KnowGPT vs. GPT-3.5	+ 23.7% (Avg.)	+ 9.2%	+ 10.8%	+ 31.2%	+ 32.4%	+ 29.2%	+ 29.4%
KnowGPT vs. GPT-4	+2.9% (Avg.)	+ 5.1%	+ 3.3%	+ 2.2%	+ 1.4%	+ 3.7%	+ 1.8%

*We used 'text-davinci-002' and 'gpt-3.5-turbo' provided by OpenAI as the implementation of GPT models.

6.4.1 Experimental Setup

Datasets. We evaluate `KnowGPT` on three question-answering datasets from two distinct fields: CommonsenseQA [86] and OpenBookQA [69] serve as benchmarks for commonsense reasoning tasks, while MedQA-USMLE [48] is used for domain-specific question answering. For detailed statistics of these datasets, please refer to Table 3.1 in the Appendix.

Background Knowledge Graph To facilitate common-sense reasoning, we employ `ConceptNet` [81], an extensive commonsense knowledge graph comprising more than 8 million interconnected entities through 34 concise relationships. For tasks specific to the medical domain, we leverage `USMLE` [111] as our foundational knowledge source. USMLE is a biomedical knowledge graph that amalgamates the Disease Database segment of the Unified Medical Language System (UMLS) [6] and DrugBank [102]. This repository encompasses 9,958 nodes and 44,561 edges.

Baselines. We select baseline models from four different categories for a thorough comparison. *LM + Fine-tuning*. We compare `KnowGPT` with standard fine-tuned language models (LMs). In particular, we use Bert-base, Bert-large [49], and RoBerta-large [64] as representative fine-tuned LMs. For both commonsense and biomedical QA tasks, we fine-tune these models by adding linear layers.

KG-enhanced LM. We also compare with several recent models that integrate knowledge graphs into QA tasks, including MHGRN [28], QA-GNN [111], HamQA [23], JointLK [83], GreaseLM [120], and GrapeQA [88]. For a fair comparison, we implement these methods using advanced language models optimized for the respective datasets. RoBerta-large [64] is used for CommonsenseQA, while AristoRoBERTa [22] is employed for OpenBookQA, and for MedQA, we use the top biomedical model, SapBERT [60]. These methods, being white-box models, are computationally expensive and therefore cannot be used with modern LLMs like GPT-3.5 or GPT-4.

LLM + Zero-shot. We include several well-known generative LLMs, including Chat-

GLM, ChatGLM2, Baichuan-7B, InternLM, GPT-3, GPT-3.5, and GPT-4, as knowledge-agnostic models. We utilize 'text-davinci-002' for GPT-3, 'gpt-3.5-turbo' for GPT-3.5, and 'gpt-4' for GPT-4 (additional details on the implementations of these models are provided in Appendix A.4). These models are evaluated in a zero-shot setting using the test set questions.

LLM + KG Prompting. To test the efficacy of our prompting strategy, we also include state-of-the-art knowledge graph prompting methods, such as CoK [95], RoG [66], and Mindmap [101]. Notably, KGR [32] is excluded from the comparison as the authors have not released their code.

6.4.2 Main Results (RQ1)

To address **RQ1**, we compare the performance of `KnowGPT` against state-of-the-art models across three benchmark datasets. We use accuracy as our performance metric, which measures the percentage of questions that the model answers correctly from the total test set questions. The following observations are made:

- `KnowGPT` outperforms all categories of baseline models, including 16 different methods, across all datasets and architectures. This indicates that `KnowGPT` is highly effective in leveraging knowledge from KGs to enhance LLMs.
- `KnowGPT` outperforms GPT-3.5 and even GPT-4 in terms of accuracy. On average, `KnowGPT` achieves 23.7% higher accuracy than GPT-3.5. Specifically, it outperforms GPT-3.5 by 10.8%, 32.4%, and 29.4% on the CommonsenseQA, OpenBookQA, and MedQA datasets, respectively. Even more impressively, despite being based on GPT-3.5, `KnowGPT` surpasses GPT-4 by 3.3%, 1.4%, and 1.8% on these datasets, respectively. These results emphasize that integrating black-box knowledge can significantly enhance LLM capabilities.
- `KnowGPT` also outperforms all KG-enhanced LMs, demonstrating the effective-

Table 6.2: OpenBookQA Leaderboard records of three groups of related models.

Human Performance (0.917)			
w/o KG	0.778	UnifiedQA [51]	0.872
MHGRN [28]	0.806	DRAGON [110]	0.878
QA-GNN [111]	0.828	GenMC [45]	0.898
GreaseLM [120]	0.848	Human Performance	0.917
HamQA [23]	0.850	GenMC Ensemble [45]	0.920
JointLK [83]	0.856	X-Reasoner [44]	0.952
GSC [97]	0.874	`KnowGPT`	**0.926**

ness of our black-box knowledge injection method. This approach offers the advantage of being adaptable to GPT-3.5 through the model API, which is not possible with white-box methods.

Leaderboard Ranking. We submitted our results to the official OpenBookQA leaderboard, maintained by the dataset authors. The full leaderboard records are available on the official site[1], and our submission can be accessed at[2].

In Table 6.2, we summarize related submissions, which fall into three categories: traditional KG-enhanced LMs, fine-tuning of LLMs (e.g., T5-11B used in UnifiedQA), and ensemble methods. `KnowGPT` outperforms traditional KG-enhanced LMs by a significant 5.2% improvement over the best baseline. Additionally, although ensemble methods dominate the leaderboard, `KnowGPT` achieves competitive performance without ensembling, outperforming GenMC Ensemble [45] by 0.6%. Remarkably, `KnowGPT` also approaches human-level performance.

[1] `https://leaderboard.allenai.org/open_book_qa/submissions/public`

[2] `https://leaderboard.allenai.org/open_book_qa/submission/cp743buq4uo7qe4e9750`

6.4.3 Ablation Studies (RQ2)

To answer **RQ2**, we conduct two ablation studies. **First**, in Table 6.3, we evaluate the impact of the tailored reinforcement learning-based knowledge extraction module, $\mathcal{P}_{\text{RL}}$, by comparing it with the heuristic sub-graph extraction strategy, $\mathcal{P}_{\text{sub}}$. We assess the performance by feeding the extracted knowledge into GPT-3.5 using the 'Sentence' prompt format, $\mathcal{F}_s$. The 'w/o KG' baseline is also included, where GPT-3.5 answers the questions without any external reasoning context. The results clearly highlight the significance of our proposed knowledge extraction strategies. **Second**, we examine the effect of the three prompt formats using the same extracted knowledge. The results, shown in Table 6.4, reveal that although the performance differences between formats are small (2.2% - 3.3%), each format is better suited to certain types of questions. We further explain these findings in the case study section. These ablation studies underscore the importance of each module in `KnowGPT`, demonstrating that combining deep RL-based knowledge extraction with context-aware prompt translation leads to the best performance across all datasets. The MAB-based prompt selection module used in KnowGPT dynamically selects the most effective prompt templates and knowledge combinations for a given query. This adapts to incompleteness and noises in KGs since the MAB balances exploration (trying new strategies) and exploitation (using known effective ones), optimizing for robustness even with imperfect KG data.

6.4.4 Case Studies (RQ3)

To address **RQ3**, we provide a detailed case study from CommonsenseQA. In Figure 6.2, we visualize the extracted knowledge and the corresponding textual input for GPT-3.5. In this case, GPT-3.5 correctly answers the question when the sentence-based prompt is used, but struggles when the question is framed using the triple or graph description formats. This comparison highlights the superior ability of `KnowGPT`

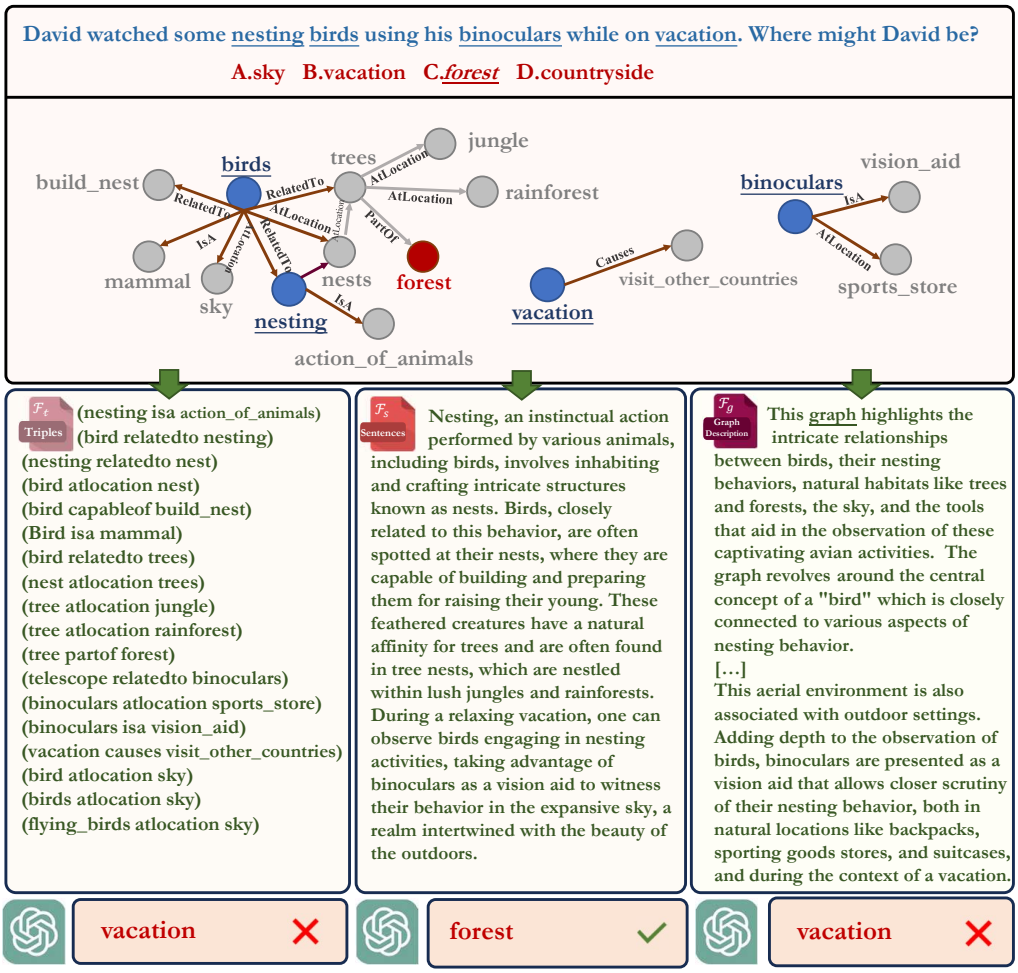


Figure 6.2: A case study on exploring the effectiveness of different prompt formats for particular questions. The extracted knowledge is shown in the middle of this figure in the form of a graph, where the nodes in blue are the key topic entities and the red is the target answer. The text boxes at the bottom are the final prompts generated based on three different formats.

Table 6.3: Ablation study on the effectiveness of two knowledge extraction methods.

Knowledge Extraction	**Model**	**CSQA**		**OBQA**	**MedQA**
		IHdev	IHtest	Test	Test
w/o KG	GPT-3	0.539	0.520	0.482	0.289
	GPT-3.5	0.735	0.710	0.598	0.487
	GPT-4	0.776	0.786	0.910	0.763
$\mathcal{P}_{\text{sub}}$	GPT-3.5	0.750	0.739	0.865	0.695
$\mathcal{P}_{\text{RL}}$	GPT-3.5	0.815	0.800	0.889	0.755
Ours	`KnowGPT`	**0.827**	**0.818**	**0.924**	**0.781**

in automatically generating contextually appropriate prompts. The following observations are made: (*i*) The triple format $\mathcal{F}_t$ is most suitable for simple questions that rely on one-hop knowledge. (*ii*) The graph description format may introduce noise by overemphasizing certain terms, leading to misleading predictions. (*iii*) `KnowGPT` excels in constructing suitable prompts based on the specific nature of each question.

6.5 Summary

This paper addresses the challenge of hallucination in large language models (LLMs), particularly in specialized domains. While LLMs have impressive reasoning capabilities, they often struggle with professional questions in specialized fields due to insufficient relevant knowledge in their pre-training datasets. To resolve this issue, we propose `KnowGPT`, a model that augments LLMs with domain-specific knowledge from knowledge graphs (KGs) to improve the accuracy of answering professional queries. `KnowGPT` integrates KGs into LLMs using model APIs, requiring no modifications to the underlying LLM architecture. Our approach utilizes a deep reinforcement learning

Table 6.4: Ablation study on the effectiveness of two knowledge extraction methods.

Knowledge Extraction	Prompts	CSQA		OBQA	MedQA
		IHdev	IHtest	Test	Test
$\mathcal{P}_{\text{sub}}$	$\mathcal{F}_t$	0.728	0.701	0.832	0.589
	$\mathcal{F}_s$	0.750	0.739	0.865	0.695
	$\mathcal{F}_g$	0.737	0.715	0.871	0.680
$\mathcal{P}_{\text{RL}}$	$\mathcal{F}_t$	0.782	0.769	0.853	0.739
	$\mathcal{F}_s$	0.815	0.800	0.889	0.755
	$\mathcal{F}_g$	0.806	0.793	0.906	0.762
Full `KnowGPT`		**0.827**	**0.818**	**0.924**	**0.781**

policy to extract concise reasoning background from KGs and employs a Multi-Armed Bandit (MAB) strategy to select the most effective knowledge extraction method and prompt template for each query. Extensive experiments across both general and domain-specific QA tasks demonstrate that `KnowGPT` outperforms all baseline models, providing strong evidence for the effectiveness of KG-enhanced LLMs.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

This thesis has presented a systematic framework for enhancing LLMs with reliable KGs, addressing critical challenges in KG refinement, completion, and dynamic alignment with LLMs. The four interconnected papers form a lifecycle solution that ensures LLMs are grounded in accurate and reliable knowledge:

- Contrastive Knowledge Graph Error Detection (Paper 1): By leveraging contrastive learning to identify semantic contradictions in KGs, this work establishes a robust foundation for KG reliability.
- Attribute-Aware KG Embedding (Paper 2): Recognizing that structural patterns alone cannot resolve ambiguities in entities with shared names, this work integrates entity attributes (e.g., geospatial coordinates, temporal metadata) into error detection.
- Inductive Knowledge Graph Completion (Paper 3): Addressing the incompleteness of even refined KGs, this work introduces a graph neural network (GNN) model that combines structural patterns with logical reasoning to infer miss-

ing relationships. The model achieves state-of-the-art performance on inductive benchmarks, enabling applications to dynamic and emerging domains.

- Inductive Knowledge Graph Completion (Paper 3): Addressing the incompleteness of even refined KGs, this work introduces a graph neural network (GNN) model that combines structural patterns with logical reasoning to infer missing relationships. The model achieves state-of-the-art performance on inductive benchmarks, enabling applications to dynamic and emerging domains.

Collectively, these contributions advance the field of AI reliability by leveraging structural information from reliable KGs, enabling LLMs to generate responses that are not only fluent but also factually grounded.

7.2 Future Work

While this thesis makes significant strides in enhancing LLMs with reliable KGs, several challenges remain unaddressed, offering fertile ground for future research:

- Scalability: Real-time retrieval from billion-scale KGs, such as Wikidata or enterprise knowledge bases, remains computationally intensive. Future work could explore distributed graph processing techniques or approximate nearest neighbor search algorithms to improve retrieval efficiency without sacrificing accuracy.

- Multimodal Knowledge Integration: Most existing systems focus on textual KGs, ignoring non-textual data such as images, time series, or geospatial information. Integrating multimodal KGs with LLMs could enable applications in domains like medical imaging (e.g., combining MRI scans with patient records) or urban planning (e.g., linking geospatial data with textual reports).

- Temporal Knowledge Integration: Real-world knowledge is dynamic, with facts evolving over time. Extending the framework to handle temporal KGs, where relationships are timestamped, could enable LLMs to generate contextually accurate outputs based on the most up-to-date information.

- Domain-Specific Customization: While the proposed framework is general-purpose, tailoring it to more specific domains (e.g., law, medicine, or education) could yield significant performance gains. For instance, legal KGs could incorporate case law hierarchies, while medical KGs could integrate ontologies like SNOMED-CT [27]. Future research could investigate domain-specific adaptations of the error detection, completion, and alignment modules.

References

[1] A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[2] Rakesh Agrawal, Tomasz Imielinski, and Arun N. Swami. Mining association rules between sets of items in large databases. In Peter Buneman and Sushil Jajodia, editors, *SIGMOD*, pages 207–216, 1999.

[3] Emily Alsentzer, John Murphy, William Boag, Wei-Hung Weng, Di Jindi, Tristan Naumann, and Matthew McDermott. Publicly available clinical bert embeddings. In *Proceedings of the 2nd Clinical Natural Language Processing Workshop*, pages 72–78, 2019.

[4] AI Anthropic. The claude 3 model family: Opus, sonnet, haiku. *Claude-3 Model Card*, 2024.

[5] Caleb Belth, Xinyi Zheng, Jilles Vreeken, and Danai Koutra. What is normal, what is strange, and what is missing in a knowledge graph: Unified characterization via inductive summarization. In *WWW*, 2020.

[6] Olivier Bodenreider. The Unified Medical Language System (UMLS): integrating biomedical terminology. *Nucleic Acids Research*, 32:D267–D270, 01 2004.

[7] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human

knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250. ACM, 2008.

[8] Kurt D. Bollacker, Robert P. Cook, and Patrick Tufts. Freebase: A shared database of structured general human knowledge. In *AAAI*, 2007.

[9] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. *NeurIPS*, 26, 2013.

[10] Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *NeurIPS*, 2013.

[11] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.

[12] Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka Jr., and Tom M. Mitchell. Toward an architecture for never-ending language learning. In *AAAI 2010*, 2010.

[13] Ilias Chalkidis, Ion Androutsopoulos, and Nikolaos Aletras. Neural legal judgment prediction in english. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4317–4323, 2019.

[14] Ilias Chalkidis, Abhik Jana, Dirk Hartung, Michael Bommarito, Ion Androutsopoulos, Daniel Martin Katz, and Nikolaos Aletras. Lexglue: A bench-

mark dataset for legal language understanding in english. *arXiv preprint arXiv:2110.00976*, 2021.

[15] Chengwei Chen, Yuan Xie, Shaohui Lin, Ruizhi Qiao, Jian Zhou, Xin Tan, Yi Zhang, and Lizhuang Ma. Novelty detection via contrastive learning with negative data augmentation. *arXiv preprint arXiv:2106.09958*, 2021.

[16] Hao Chen, Zhong Huang, Yue Xu, Zengde Deng, Feiran Huang, Peng He, and Zhoujun Li. Neighbor enhanced graph convolutional networks for node classification and recommendation. *Knowledge-Based Systems*, 246:108594, 2022.

[17] Hao Chen, Yue Xu, Feiran Huang, Zengde Deng, Wenbing Huang, Senzhang Wang, Peng He, and Zhoujun Li. Label-aware graph convolutional networks. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, page 1977–1980, 2020.

[18] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, 2020.

[19] Kewei Cheng, Yikai Zhu, Ming Zhang, and Yizhou Sun. Noigan: Noise aware knowledge graph embedding with gan. 2019.

[20] Yurong Cheng, Lei Chen, Ye Yuan, and Guoren Wang. Rule-based graph repairing: Semantic and efficient repairing methods. In *ICDE*, 2018.

[21] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, and et al. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research (JMLR)*, 2023.

[22] Peter Clark, Oren Etzioni, Tushar Khot, Daniel Khashabi, Bhavana Mishra, Kyle Richardson, Ashish Sabharwal, Carissa Schoenick, Oyvind Tafjord, Niket Tandon, et al. From 'f'to 'a'on the ny regents science exams: An overview of the aristo project. *AI Magazine*, 41(4):39–53, 2020.

[23] Junnan Dong, Qinggang Zhang, Xiao Huang, Keyu Duan, Qiaoyu Tan, and Zhimeng Jiang. Hierarchy-aware multi-hop question answering over knowledge graphs. In *WWW*, pages 2519–2527, 2023.

[24] Junnan Dong, Qinggang Zhang, Xiao Huang, Qiaoyu Tan, Daochen Zha, and Zhao Zihao. Active ensemble learning for knowledge graph error detection. In *WSDM*, pages 877–885, 2023.

[25] Junnan Dong, Qinggang Zhang, Chuang Zhou, Hao Chen, Daochen Zha, and Xiao Huang. Cost-efficient knowledge-based question answering with large language models. *arXiv preprint arXiv:2405.17337*, 2024.

[26] Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 601–610, 2014.

[27] Kevin Donnelly et al. Snomed-ct: The advanced terminology and coding system for ehealth. *Studies in health technology and informatics*, 121:279, 2006.

[28] Yanlin Feng, Xinyue Chen, Bill Yuchen Lin, Peifeng Wang, Jun Yan, and Xiang Ren. Scalable multi-hop relational reasoning for knowledge-aware question answering. In *EMNLP*, pages 1295–1309, 2020.

[29] Luis Antonio Galárraga, Christina Teflioudi, Katja Hose, and Fabian M. Suchanek. AMIE: association rule mining under incomplete evidence in ontological knowledge bases. In *WWW*, 2013.

[30] Congcong Ge, Yunjun Gao, Honghui Weng, Chong Zhang, Xiaoye Miao, and Baihua Zheng. Kgclean: An embedding powered knowledge graph cleaning framework. *arXiv preprint arXiv:2004.14478*, 2020.

[31] Yingqiang Ge, Wenyue Hua, Kai Mei, Juntao Tan, Shuyuan Xu, Zelong Li, Yongfeng Zhang, et al. Openagi: When llm meets domain experts. *Advances in Neural Information Processing Systems*, 36, 2024.

[32] Xinyan Guan, Yanjiang Liu, Hongyu Lin, Yaojie Lu, Ben He, Xianpei Han, and Le Sun. Mitigating large language model hallucinations via autonomous knowledge graph-based retrofitting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18126–18134, 2024.

[33] Shu Guo, Quan Wang, Lihong Wang, Bin Wang, and Li Guo. Knowledge graph embedding with iterative guidance from soft rules. In *AAAI*, 2018.

[34] Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, 2020.

[35] Kaveh Hassani and Amir Hosein Khas Ahmadi. Contrastive multi-view representation learning on graphs. In *ICML*, 2020.

[36] Stefan Heindorf, Martin Potthast, Benno Stein, and Gregor Engels. Vandalism detection in wikidata. In *CIKM*, 2016.

[37] Olivier J. Hénaff. Data-efficient image recognition with contrastive predictive coding. In *ICML*, 2020.

[38] R. Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Philip Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. In *ICLR*, 2019.

[39] Linmei Hu, Zeyi Liu, Ziwang Zhao, Lei Hou, Liqiang Nie, and Juanzi Li. A survey of knowledge enhanced pre-trained language models. *IEEE Transactions on Knowledge and Data Engineering*, 2023.

[40] Feiran Huang, Zefan Wang, Xiao Huang, Yufeng Qian, Zhetao Li, and Hao Chen. Aligning distillation for cold-start item recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, page 1147–1157, 2023.

[41] Feiran Huang, Zhenghang Yang, Junyi Jiang, Yuanchen Bei, Yijie Zhang, and Hao Chen. Large language model interaction simulator for cold-start item recommendation. *arXiv preprint arXiv:2402.09176*, 2024.

[42] Xiao Huang, Jingyuan Zhang, Dingcheng Li, and Ping Li. Knowledge graph embedding based question answering. In *WSDM*, 2019.

[43] Xiao Huang, Jingyuan Zhang, Dingcheng Li, and Ping Li. Knowledge graph embedding based question answering. In *WSDM*, pages 105–113, 2019.

[44] Yongfeng Huang, Yanyang Li, Yichong Xu, Lin Zhang, Ruyi Gan, Jiaxing Zhang, and Liwei Wang. Mvp-tuning: Multi-view knowledge retrieval with prompt tuning for commonsense reasoning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13417–13432, 2023.

[45] Zixian Huang, Ao Wu, Jiaying Zhou, Yu Gu, Yue Zhao, and Gong Cheng. Clues before answers: Generation-enhanced multiple-choice qa. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3272–3287, 2022.

[46] Shengbin Jia, Yang Xiang, Xiaojun Chen, Kun Wang, and Shijia E. Triple trustworthiness measurement for knowledge graph. In *WWW*, 2019.

[47] Jinhao Jiang, Kun Zhou, Xin Zhao, and Ji-Rong Wen. UniKGQA: Unified retrieval and reasoning for solving multi-hop question answering over knowledge

graph. In *The Eleventh International Conference on Learning Representations*, 2023.

[48] Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *Applied Sciences*, 11(14), 2021.

[49] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186, 2019.

[50] Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. UnifiedQA: Crossing format boundaries with a single qa system. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1896–1907, 2020.

[51] Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. Unifiedqa: Crossing format boundaries with a single qa system. In *EMNLP 2020*, pages 1896–1907, 2020.

[52] Thomas N. Kipf, Elise van der Pol, and Max Welling. Contrastive learning of structured world models. In *ICLR*, 2020.

[53] Jieh-Sheng Lee and Jieh Hsiang. Patent classification by fine-tuning bert language model. *World Patent Information*, 61:101965, 2020.

[54] Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240, 2020.

[55] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef,

Sören Auer, and Christian Bizer. Dbpedia - A large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6(2):167–195, 2015.

[56] Patrick Lewis, Ethan Perez, Aleksandra Piktus, and et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[57] Huayang Li, Yixuan Su, Deng Cai, Yan Wang, and Lemao Liu. A survey on retrieval-augmented text generation. *arXiv preprint arXiv:2202.01110*, 2022.

[58] Wentao Li, Huachi Zhou, Junnan Dong, and et al. Constructing low-redundant and high-accuracy knowledge graphs for education. In *International Conference on Web-Based Learning (ICWL)*, 2022.

[59] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. Learning entity and relation embeddings for knowledge graph completion. In *AAAI*, 2015.

[60] Fangyu Liu, Ehsan Shareghi, Zaiqiao Meng, Marco Basaldella, and Nigel Collier. Self-alignment pretraining for biomedical entity representations. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4228–4238, 2021.

[61] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35, 2023.

[62] Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. K-bert: Enabling language representation with knowledge graph. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 2901–2908, 2020.

[63] Yezi Liu, Qinggang Zhang, Mengnan Du, Xiao Huang, and Xia Hu. Error detection on knowledge graphs with triple embedding. In *2023 31st European Signal Processing Conference (EUSIPCO)*, pages 1604–1608. IEEE, 2023.

[64] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[65] LINHAO LUO, Yuan-Fang Li, Reza Haf, and et al. Reasoning on graphs: Faithful and interpretable large language model reasoning. In *International Conference on Learning Representations (ICLR)*, 2024.

[66] Linhao Luo, Yuan-Fang Li, Gholamreza Haffari, and Shirui Pan. Reasoning on graphs: Faithful and interpretable large language model reasoning. *arXiv preprint arXiv:2310.01061*, 2023.

[67] Farzaneh Mahdisoltani, Joanna Biega, and Fabian M. Suchanek. YAGO3: A knowledge base from multilingual wikipedias. In *CIDR*, 2015.

[68] André Melo and Heiko Paulheim. Detection of relation assertion errors in knowledge graphs. In *Knowledge Capture Conference*, 2017.

[69] Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *EMNLP*, pages 2381–2391, 2018.

[70] Tom Mitchell, William Cohen, Estevam Hruschka, Partha Talukdar, Bishan Yang, Justin Betteridge, Andrew Carlson, Bhavana Dalvi, Matt Gardner, Bryan Kisiel, et al. Never-ending learning. *Communications of the ACM*, 61(5):103–115, 2018.

[71] Deepak Nathani, Jatin Chauhan, Charu Sharma, and Manohar Kaul. Learning attention-based embeddings for relation prediction in knowledge graphs. In *ACL*, 2019.

[72] Mojtaba Nayyeri, Sahar Vahdati, Emanuel Sallinger, Mirza Mohtashim Alam, Hamed Shariat Yazdi, and Jens Lehmann. Pattern-aware and noise-resilient embedding models. In *European Conference on Information Retrieval*, pages 483–496. Springer, 2021.

[73] OpenAI. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[74] Shirui Pan, Linhao Luo, Yufei Wang, Chen Chen, Jiapu Wang, and Xindong Wu. Unifying large language models and knowledge graphs: A roadmap. *IEEE Transactions on Knowledge and Data Engineering*, 2024.

[75] Heiko Paulheim. Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic Web*, 2017.

[76] Heiko Paulheim and Aldo Gangemi. Serving dbpedia with DOLCE - more than just adding a cherry on top. In *ISWC*, 2015.

[77] Matthew E Peters, Mark Neumann, Robert Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A Smith. Knowledge enhanced contextual word representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 43–54, 2019.

[78] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *European semantic web conference*, pages 593–607. Springer, 2018.

[79] Yingchun Shan, Chenyang Bu, Xiaojian Liu, Shengwei Ji, and Lei Li. Confidence-aware negative sampling method for noisy knowledge graph em-

bedding. In *2018 IEEE International Conference on Big Knowledge (ICBK)*, pages 33–40. IEEE, 2018.

[80] Chen Shengyuan, Yunfeng Cai, Huang Fang, Xiao Huang, and Mingming Sun. Differentiable neuro-symbolic reasoning on large-scale knowledge graphs. *Advances in Neural Information Processing Systems*, 36, 2024.

[81] Robyn Speer, Joshua Chin, and Catherine Havasi. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31, 2017.

[82] Yu Sun, Shuohuan Wang, Shikun Feng, Siyu Ding, Chao Pang, Junyuan Shang, Jiaxiang Liu, Xuyi Chen, Yanbin Zhao, Yuxiang Lu, et al. Ernie 3.0: Large-scale knowledge enhanced pre-training for language understanding and generation. *arXiv preprint arXiv:2107.02137*, 2021.

[83] Yueqing Sun, Qi Shi, Le Qi, and Yu Zhang. Jointlk: Joint reasoning with language models and knowledge graphs for commonsense question answering. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5049–5060, 2022.

[84] Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and et al. Rotate: Knowledge graph embedding by relational rotation in complex space. *arXiv preprint arXiv:1902.10197*, 2019.

[85] Jihoon Tack, Sangwoo Mo, Jongheon Jeong, and Jinwoo Shin. Csi: Novelty detection via contrastive learning on distributionally shifted instances. *Advances in neural information processing systems*, 33:11839–11852, 2020.

[86] Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. Commonsenseqa: A question answering challenge targeting commonsense knowledge. 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), pages 4149–4158, 2019.

[87] Thomas Pellissier Tanon, Daria Stepanova, Simon Razniewski, Paramita Mirza, and Gerhard Weikum. Completeness-aware rule learning from knowledge graphs. In *IJCAI*, 2018.

[88] Dhaval Taunk, Lakshya Khanna, Siri Venkata Pavan Kumar Kandru, Vasudeva Varma, Charu Sharma, and Makarand Tapaswi. Grapeqa: Graph augmentation and pruning to enhance question-answering. In *Companion Proceedings of the ACM Web Conference 2023*, pages 1138–1144, 2023.

[89] Komal Teru, Etienne Denis, and Will Hamilton. Inductive relation prediction by subgraph reasoning. In *International Conference on Machine Learning*, pages 9448–9457. PMLR, 2020.

[90] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. In *ICML*, 2016.

[91] Mina Valizadeh and Natalie Parde. The ai doctor is in: A survey of task-oriented dialogue systems for healthcare applications. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6638–6660, 2022.

[92] Hongwei Wang, Hongyu Ren, and Jure Leskovec. Relational message passing for knowledge graph completion. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 1697–1707, 2021.

[93] Hongwei Wang, Miao Zhao, Xing Xie, Wenjie Li, and Minyi Guo. Knowledge graph convolutional networks for recommender systems. In *WWW*, 2019.

[94] Hongwei Wang, Miao Zhao, Xing Xie, Wenjie Li, and Minyi Guo. Knowledge graph convolutional networks for recommender systems. In *The world wide web conference*, pages 3307–3313, 2019.

[95] Jianing Wang, Qiushi Sun, Nuo Chen, and et al. Boosting language models reasoning with chain-of-knowledge prompting. *arXiv preprint arXiv:2306.06427*, 2023.

[96] Keheng Wang, Feiyu Duan, Sirui Wang, Peiguang Li, Yunsen Xian, Chuantao Yin, Wenge Rong, and Zhang Xiong. Knowledge-driven cot: Exploring faithful reasoning in llms for knowledge-intensive question answering. *arXiv preprint arXiv:2308.13259*, 2023.

[97] Kuan Wang, Yuyu Zhang, Diyi Yang, Le Song, and Tao Qin. Gnn is a counter? revisiting gnn for question answering, 2021.

[98] Peifeng Wang, Jialong Han, Chenliang Li, and Rong Pan. Logic attention based neighborhood aggregation for inductive knowledge graph embedding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7152–7159, 2019.

[99] Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. Kepler: A unified model for knowledge embedding and pre-trained language representation. *Transactions of the Association for Computational Linguistics*, 9:176–194, 2021.

[100] Yingheng Wang, Yaosen Min, Xin Chen, and Ji Wu. Multi-view graph contrastive representation learning for drug-drug interaction prediction. In *WWW*, 2021.

[101] Yilin Wen, Zifeng Wang, and Jimeng Sun. Mindmap: Knowledge graph prompting sparks graph of thoughts in large language models. *arXiv preprint arXiv:2308.09729*, 2023.

[102] David S Wishart, Yannick D Feunang, An C Guo, Elvis J Lo, Ana Marcu, Jason R Grant, Tanvir Sajed, Daniel Johnson, Carin Li, Zinat Sayeeda, Nazanin Assempour, Ithayavani Iynkkaran, Yifeng Liu, Adam Maciejewski, Nicola Gale, Alex Wilson, Lucy Chin, Ryan Cummings, Diana Le, Allison Pon, Craig Knox, and Michael Wilson. DrugBank 5.0: a major update to the DrugBank database for 2018. *Nucleic Acids Research*, 46:D1074–D1082, 2017.

[103] Xin Xia, Hongzhi Yin, Junliang Yu, Qinyong Wang, Lizhen Cui, and Xiangliang Zhang. Self-supervised hypergraph convolutional networks for session-based recommendation. In *AAAI*, pages 4503–4511, 2021.

[104] Ruobing Xie, Zhiyuan Liu, Fen Lin, and Leyu Lin. Does william shakespeare really write hamlet? knowledge representation learning with confidence. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[105] Wenhan Xiong, Thien Hoang, and William Yang Wang. Deeppath: A reinforcement learning method for knowledge graph reasoning. In *EMNLP*, pages 564–573, 2017.

[106] Zhiming Xu, Xiao Huang, Yue Zhao, Yushun Dong, and Jundong Li. Contrastive attributed network anomaly detection with data augmentation. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 444–457. Springer, 2022.

[107] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. In *ICLR*, 2015.

[108] Fan Yang, Zhilin Yang, and William W Cohen. Differentiable learning of logical rules for knowledge base completion. *CoRR, abs/1702.08367*, 2017.

[109] Linyao Yang, Hongyang Chen, Zhao Li, Xiao Ding, and Xindong Wu. Give us the facts: Enhancing large language models with knowledge graphs for fact-

aware language modeling. *IEEE Transactions on Knowledge and Data Engineering*, 2024.

[110] Michihiro Yasunaga, Antoine Bosselut, Hongyu Ren, Xikun Zhang, Christopher D Manning, Percy S Liang, and Jure Leskovec. Deep bidirectional language-knowledge graph pretraining. *Advances in Neural Information Processing Systems*, 35:37309–37323, 2022.

[111] Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. Qa-gnn: Reasoning with language models and knowledge graphs for question answering. *NAACL*, 2021.

[112] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations. In *NeurIPS*, 2020.

[113] Jiaqi Zeng and Pengtao Xie. Contrastive self-supervised learning for graph classification. In *AAAI*, 2021.

[114] Muhan Zhang and Yixin Chen. Link prediction based on graph neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[115] Qinggang Zhang, Junnan Dong, Hao Chen, and et al. Knowgpt: Knowledge graph based prompting for large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[116] Qinggang Zhang, Junnan Dong, Keyu Duan, Xiao Huang, Yezi Liu, and Linchuan Xu. Contrastive knowledge graph error detection. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 2590–2599, 2022.

[117] Qinggang Zhang, Junnan Dong, Qiaoyu Tan, and Xiao Huang. Integrating entity attributes for error-aware knowledge graph embedding. *IEEE Transactions on Knowledge and Data Engineering*, 2023.

[118] Qinggang Zhang, Keyu Duan, Junnan Dong, Pai Zheng, and Xiao Huang. Logical reasoning with relation network for inductive knowledge graph completion. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 4268–4277, 2024.

[119] Xikun Zhang, Antoine Bosselut, Michihiro Yasunaga, Hongyu Ren, Percy Liang, Christopher D Manning, and Jure Leskovec. Greaselm: Graph reasoning enhanced language models for question answering. *arXiv preprint arXiv:2201.08860*, 2022.

[120] Xikun Zhang, Antoine Bosselut, Michihiro Yasunaga, Hongyu Ren, Percy Liang, Christopher D Manning, and Jure Leskovec. Greaselm: Graph reasoning enhanced language models for question answering. *arXiv preprint arXiv:2201.08860*, 2022.

[121] Ruilin Zhao, Feng Zhao, Long Wang, Xianzhi Wang, and Guandong Xu. Kg-cot: Chain-of-thought prompting of large language models over knowledge graphs for knowledge-aware question answering. In Kate Larson, editor, *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24*, pages 6642–6650. International Joint Conferences on Artificial Intelligence Organization, 8 2024. Main Track.

[122] Haoxi Zhong, Chaojun Xiao, Cunchao Tu, Tianyang Zhang, Zhiyuan Liu, and Maosong Sun. How does nlp benefit legal system: A summary of legal artificial intelligence. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5218–5230, 2020.

[123] Huachi Zhou, Hao Chen, Junnan Dong, Daochen Zha, Chuang Zhou, and Xiao Huang. Adaptive popularity debiasing aggregator for graph collaborative filtering. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 7–17, 2023.

[124] Fengbin Zhu, Wenqiang Lei, Chao Wang, Jianming Zheng, Soujanya Poria, and Tat-Seng Chua. Retrieving and reading: A comprehensive survey on open-domain question answering. *arXiv preprint arXiv:2101.00774*, 2021.

[125] Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. Deep graph contrastive representation learning. *arXiv preprint arXiv:2006.04131*, 2020.