Doctoral Dissertation

Exploration of Large Language Models for Noise Detection and Refinement in Knowledge Graphs

Na DONG

Supervisor: Kiyoaki SHIRAI

Graduate School of Advanced Science and Technology
Japan Advanced Institute of Science and Technology
(Information Science)

March 2026

Abstract

Knowledge graphs (KGs) have emerged as a powerful paradigm for representing structured knowledge about the real world. KGs encode information as a collection of triples, each consisting of a head entity, a relation, and a tail entity. This kind of structure enables not only the integration of heterogeneous data sources but also the support of downstream tasks such as question answering, recommender systems, and semantic search. However, the practical application of KGs is often hindered by several common challenges. KGs automatically constructed from large text corpora are prone to errors due to natural language ambiguity and extraction processes; while manually constructed KGs, while more reliable, are still susceptible to human error, have limited scalability, and are incomplete. As a result, KGs frequently contain misinformation, redundancy, and noisy triples, all of which can undermine their reliability and reduce their effectiveness in real-world applications.

This dissertation addresses the critical problem of improving the quality of noisy KGs through a two-step framework: (1) detecting noisy triples, and (2) refining the noisy triples by correcting the entities involved. Noisy KGs not only distort factual knowledge but also propagate errors to downstream reasoning and knowledge completion tasks. Unlike conventional approaches that rely heavily on hand-crafted rules, statistical heuristics, or embedding-based anomaly detection, our work leverages recent advances in large language models (LLMs), which possess strong semantic understanding and contextual reasoning capabilities, making them well-suited for identifying and repairing noisy knowledge structures at scale.

Within this framework, we propose two methods: *LLM_sim* and *LLM_rule*. The first method, *LLM_sim*, evaluates the plausibility of a candidate triple by comparing it with existing triples in the KG, thereby capturing subtle semantic inconsistencies that traditional models often miss. In addition to detection, we designed *LLM_sim* to repair detected noisy triples by grouping and calculating similarity. The second method, *LLM_rule*, induces logical rules directly from the KG to capture semantic constraints between entities and relations. We generate relevant rules through *LLM_rule*, which can propose effective corrections, providing interpretability and transparency that are often missing in similarity-based or embedding-based methods.

To validate the effectiveness of the proposed methods, we conducted extensive experiments on both synthetic and real-world noisy KGs. Synthetic datasets allow controlled injection of noise. These datasets enable systematic evaluation of detection and refinement capabilities under varying noise

ratios. Real-world datasets reflect the complex and heterogeneous nature of naturally occurring noise. Across both settings, our results demonstrate that *LLM_sim* and *LLM_rule* achieve strong performance in identifying and refining noisy triples.

Crucially, our comparative analysis reveals that **LLM_rule** generally emerges as the superior model, outperforming LLM_sim in both detection accuracy and refinement quality across fact-oriented datasets (e.g., FB15k-237, s-NELL). While LLM_sim remains competitive in linguistically rich domains like WN18RR, LLM_rule's integration of explicit logical constraints offers greater robustness against hallucination and structural inconsistencies.

We further investigated the subsequent impact of noise refinement on knowledge graph completion (KGC). KGC aims to infer missing triples in a KG by learning patterns from observed triples. However, noise in the training data severely impairs the generalization performance of KGC models. By incorporating refined KGC generated by *LLM_sim* and *LLM_rule*, we observed significant improvements in standard KGC metrics, including mean reciprocal rank (MRR) and hits@k. These improvements confirm that high-quality input data is a crucial factor in improving the performance of completion models and that noise refinement can serve as a key preprocessing step in the KGC pipeline.

The contributions of this work can be summarized as follows. First, we formulate a general framework for addressing noise in KGs that combines detection and refinement in a unified process. Second, we propose two novel methods, *LLM_sim* and *LLM_rule*, that leverage the semantic reasoning and rule induction capabilities of LLMs to achieve robust noise handling. Third, we provide extensive empirical evidence from both synthetic and real-world datasets, demonstrating that our methods consistently outperform baseline approaches in both detection accuracy and refinement quality. Finally, we show that our noise refinement framework directly enhances the performance of downstream KGC models, thereby underscoring its practical significance for KG applications. This study advances the state of the art in KG refinement by integrating the semantic power of LLMs with interpretable rule-based reasoning. The proposed methods offer a scalable, effective, and interpretable solution to the longstanding problem of noise in KGs. By improving KG quality, our work not only enhances their reliability but also paves the way for more accurate and trustworthy knowledge-driven AI systems. The findings of this dissertation hold broader implications for the development of robust intelligent systems that depend on high-quality structured knowledge.

Keywords: Knowledge Graph, Knowledge Acquisition, Error Correction, Large Language Model, Rule Induction, Knowledge Graph Completion

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The rapid development of artificial intelligence has significantly advanced how knowledge is represented, processed, and utilized in academic research and industrial applications. Central to this progress lies the need to efficiently capture, organize and reason about vast amounts of structured and unstructured information [41]. Traditional knowledge representation methods, while effective within limited domains, often lack the scalability and expressiveness required to handle large amounts of heterogeneous data. This gap has led to the emergence of knowledge graphs (KGs), which provide a structured and interpretable representation of entities and their relationships in the form of triples [21].

Concurrently, the emergence of large language models (LLMs) has driven breakthroughs in natural language processing (NLP), leveraging large-scale pre-training on diverse corpora to capture extensive linguistic and factual knowledge. Unlike earlier approaches, LLMs exhibit emerging capabilities such as contextual reasoning, learning from context, and flexible text generation, making them versatile tools for a wide range of tasks.

This thesis focused on the automatic refinement of KG using an LLM. Section 1.1 introduces the background of this thesis. Section 1.2 addresses the specific goal of this thesis. Section 1.3 discusses the research questions. Section 1.4 describes the structure of the overall thesis.

1.1 Research Background

This section introduces the foundational concepts relevant to this paper. First, we provide an overview of KGs, their architecture, and their role in real-world applications. Next, we review the development of LLM, focusing on its architecture, training paradigm, and emerging properties. Finally, we

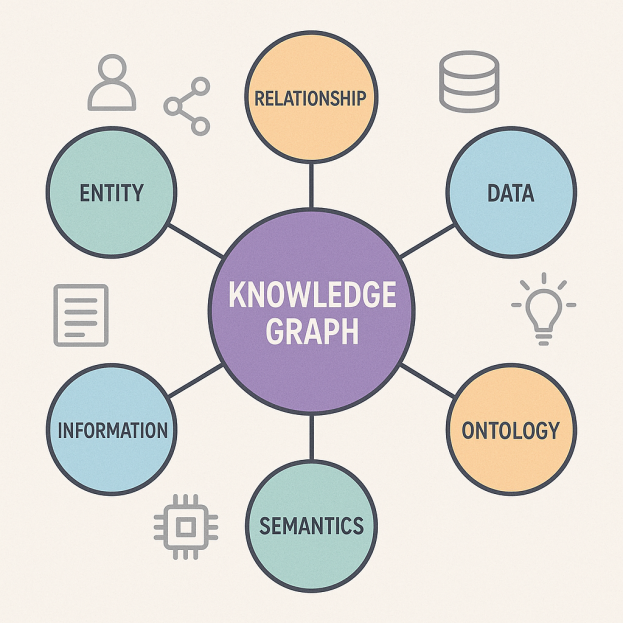

Figure 1.1: Knowledge Graph conception

discuss the synergy between KGs and LLM, outlining the opportunities and challenges driving this research.

1.1.1 Knowledge Graph

KGs have become a central paradigm for representing structured knowledge about the real world. As shown in Figure 1.1, KGs integrate key components such as entities, relations, data, semantics, and ontologies into a unified framework. The inclusion of data and information grounds KGs in factual evidence, while semantics and ontologies provide higher-level meaning and logical organization, enhancing interpretability and reasoning capabilities. In research, KGs typically encode information as triples $\langle h, r, t \rangle$, where h represents the head entity, r represents the relation, and t represents the tail entity [4, 10, 74]. Entities correspond to the basic units of knowledge, while relations capture the connections and interactions between them. This interconnected structure enables KGs to support a wide range of applications, including search engines, digital assistants, recommender systems, scientific discovery, and intelligent question answering [30].

In Web search, KGs enhance query understanding by linking terms to semantic entities, thereby improving relevance and interpretability [7, 80]. In digital assistants, KGs enable context-aware dialogue systems that reason

over factual knowledge [55, 81, 27]. In recommendation systems, KGs capture user preferences and item characteristics, enhancing recommendation accuracy, diversity, and explainability [19, 14, 84, 75]. In biomedicine, KGs facilitate discovery by linking genes, diseases, and treatments in a structured way [33, 57, 63]. These applications underline the central role of KGs as a foundation for modern AI [19, 31]. Prominent large-scale KGs such as Freebase, DBpedia, YAGO, NELL, and Wikidata demonstrate their feasibility and utility [52]. By providing a structured backbone for integrating heterogeneous data, KGs bridge human knowledge and computational intelligence [79].

The construction of KGs generally follows two approaches: manual curation and automatic extraction [29]. Manual curation, common in high-stakes domains such as biomedicine, relies on experts encoding knowledge into ontologies or databases [9, 64]. While these KGs offer high accuracy, they face scalability bottlenecks and are expensive to maintain [88]. Even carefully curated KGs are prone to human error, conceptual inconsistencies, and have limited scalability [50]. Automatic extraction of a KG uses NLP to extract entities and relations from text corpora [68, 31]. While automatically extracted KGs are scalable, they often suffer from errors due to incorrect entity linking, ambiguous relationships, or misinterpreted context. As a result, most KGs contain misinformation, incompleteness, and redundancy, compromising their reliability and downstream applications. Therefore, ensuring the quality of KGs is crucial for trustworthy knowledge-driven AI.

A noisy triple is a triple that is erroneous or outdated and therefore inconsistent with current facts. In this paper, we focus on noisy triples where the entities and relations are incompatible. Noisy triples often originate from *automated extraction errors*, where information extraction systems misinterpret natural language text, from *inaccurate entity linking*, in which mentions are incorrectly mapped to KG nodes, or from *relation classification* errors, where the semantic relation between two entities is misidentified and thus produces factually inconsistent triples [69, 35]. This noise undermines the factual integrity of KGs and can lead to erroneous conclusions in downstream applications [23, 90]. Furthermore, maintaining timely and consistent KGs remains challenging given the vast and constantly evolving nature of real-world knowledge [16, 2].

KGC aims to address the inherent incompleteness of real-world KGs by inferring plausible triples from observed triples to predict missing links in KGs. By learning representations of entities and relations, KGC models can estimate the likelihood of candidate triples, enabling applications such as link prediction, entity alignment, and knowledge-driven question answering. Noisy triples not only degrade KG quality but also propagate misleading

evidence in reasoning tasks and impair the performance of KGC models. Detecting and refining noise is therefore crucial for ensuring trustworthy and effective KG-based AI systems.

Beyond the issues of noise and incompleteness, KGs often contain complex semantic dependencies that are difficult to model using solely structural information. Two significant challenges in this regard are **implicit relations** and **long-range relations**. Implicit relations refer to semantic connections between entities that are not explicitly materialized as direct edges but can be logically inferred from background knowledge. For instance, determining a person's nationality often requires inferring an implicit link between their birth city and the country. On the other hand, long-range relations involve dependencies separated by multiple intermediate nodes, such as predicting a drug's side effects based on extended biological pathways. Capturing such relations requires reasoning over chains of evidence, which is often challenging for traditional embedding-based models that focus primarily on local structures. In this study, we aim to mitigate these challenges by leveraging the semantic reasoning capabilities of LLMs. Specifically, our proposed *LLM_-sim* incorporates Additional Context (AC) to provide the necessary multi-hop evidence, while *LLM_rule* explicitly generates logical rules to capture these complex dependencies, thereby enabling more robust noise detection compared to structure-only baselines.

1.1.2 Large Language Models

Large language models (LLMs) have rapidly become a cornerstone of modern NLP, fundamentally reshaping the approach to language understanding and generation tasks. At their core, LLMs are based on the Transformer architecture, which introduces key innovations such as self-attention and multi-head attention [73]. These advances enable efficient parallelization of training and inference while also facilitating the modeling of long-range dependencies in text. Compared to earlier neural network paradigms such as recurrent neural networks (RNNs) or convolutional neural networks (CNNs), the Transformer framework scales more efficiently in terms of both training data size and the number of model parameters. This scalability paves the way for models with hundreds of billions of parameters, unlocking unprecedented capabilities.

Figure 1.2 illustrates a chronological overview of the major milestones in the development of LLMs. Beginning with the introduction of BERT in 2018, which pioneered bidirectional pre-training and significantly advanced natural language understanding tasks, the field has witnessed rapid and transformative progress [22]. In 2019, GPT-2 demonstrated the potential of autoregressive pre-training, highlighting the ability of generative models to produce

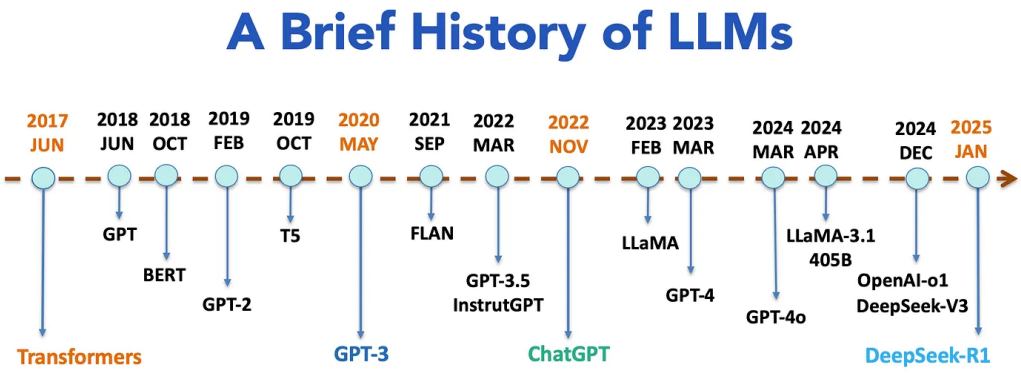

Figure 1.2: Timeline for LLMs [54]

coherent long-form text. This trajectory culminated in GPT-3 (2020), a 175-billion parameter model that established the paradigm of zero-shot and few-shot learning, drastically reducing the reliance on task-specific fine-tuning. The release of ChatGPT in 2022 made LLM capabilities broadly accessible to the public, triggering widespread adoption across domains. In parallel, the open-source LLaMA family of models (2023) offered parameter-efficient yet competitive alternatives, public access to cutting-edge LLMs for the research community. Finally, GPT-4 (2023) represents the current frontier, with further improvements in reasoning ability, robustness, and multimodal integration. This timeline highlights the exponential growth in both model size and capability, as well as the paradigm shift from task-specific architectures toward general-purpose models capable of tackling a wide spectrum of natural language tasks. Such rapid evolution underscores both the promise of LLMs and the pressing challenges related to their scalability, reliability, and integration with external knowledge resources.

LLMs are trained on massive corpora, including web data, books, and scientific papers, enabling them to capture a broad range of linguistic and factual knowledge. Their notable features are their emergent capabilities, including text generation, contextual learning, and combinatorial reasoning, which are not readily available in smaller models [77]. These properties enable LLMs to perform a wide range of tasks, such as summarization, conversation, code generation, and question-answering, without requiring task-specific fine-tuning [1]. This versatility has driven the widespread adoption of LLMs in academia and industry, impacting diverse fields such as education, scientific discovery, and human-computer interaction [11].

Despite their impressive success, LLMs face several fundamental limitations. First, their knowledge is implicitly encoded in model parameters,

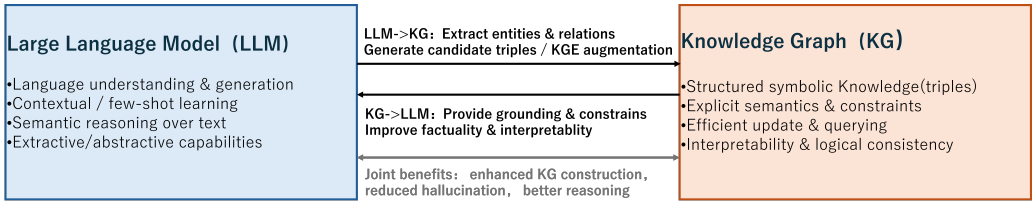

Figure 1.3: LLMs and KGs

making them difficult to dynamically update or verify for correctness [53, 48]. This can lead to problems such as hallucinations, where seemingly plausible but in fact false statements are generated [32]. Second, they lack explicit structured reasoning mechanisms and often fail to provide interpretable justification for their outputs. Third, training and deployment incur significant computational costs, raising concerns about efficiency, reproducibility, and accessibility [66].

These limitations highlight the need to enhance LLMs with external resources that can provide stability, reliability, and interpretability. KGs, as structured repositories of symbolic knowledge, offer complementary advantages, such as explicit semantics, logical consistency, and efficient update mechanisms [30]. Integrating LLMs with KGs represents a promising research direction, aiming to combine the generalization capabilities of neural models with the trustworthiness of symbolic representations. This combination forms the core motivation of this paper.

In recent years, the fusion of LLMs and KGs has attracted considerable attention due to their complementary strengths. On the one hand, LLMs can leverage their language understanding capabilities to enhance KGs, supporting tasks such as entity disambiguation, relation extraction, and knowledge graph embedding (KGE). On the other hand, KGs can benefit LLMs by providing clear grounding and constraints, mitigating hallucinations, and improving interpretability [49].

Figure 1.3 illustrates the complementary relationship between LLMs and KGs. On the left, LLMs excel at understanding and generating natural language, performing contextual and few-shot learning, and reasoning over unstructured text. These capabilities enable LLMs to extract entities and relations from raw text, generate candidate triples, and enhance KGE. LLMs contribute to KG construction by extracting entities and relations from unstructured text with improved accuracy [6]. In addition, LLMs enhance KGC by semantically reasoning about observed knowledge to generate missing triples [39]. On the right, KGs provide structured symbolic knowledge in the form of triples, offering explicit semantics, logical consistency, effi-

cient querying, and interpretability. Through bidirectional interaction, KGs can serve as grounding resources that constrain LLM predictions, mitigating hallucinations and improving factual accuracy and transparency. The synergy between LLMs and KGs thus supports enhanced KG construction, reduced noise, and improved reasoning performance. The synergy between KGs and LLMs presents new opportunities and challenges in integrating symbolic knowledge representations [15]. Temporal dynamics pose particular difficulties: KGs are relatively static and costly to update, while LLMs capture evolving but often inconsistent knowledge from text[48]. Achieving a balance between stability and adaptability is essential for building reliable systems. Furthermore, both KG construction and LLM inference are computationally intensive, underscoring the need for scalable methods that reduce resource consumption without sacrificing accuracy. Finally, while symbolic rules provide interpretability and explicit reasoning that can complement the statistical power of LLMs, effectively combining rule-based reasoning with data-driven LLM inference for KG refinement, noise detection, and KGC remains an open research problem [42, 47].

1.2 Research Objectives

While ensuring the reliability of KGs is a core practical objective, the primary scientific goal of this study is to investigate the semantic reasoning and rule-induction capabilities of Large Language Models (LLMs) in the context of noise mitigation. Specifically, we aim to improve the quality of KGs by developing LLM-based methods that serve not just as knowledge sources, but as active reasoners for structural verification. To this end, we propose two approaches, *LLM_sim* and *LLM_rule*, which respectively leverage semantic reasoning and symbolic rules to enhance KG consistency. Beyond methodological development, we systematically evaluate the robustness of both approaches across datasets with varying noise characteristics and further investigate their impact on downstream KGC tasks. Taken together, these objectives aim to establish LLM-based refinement as an effective strategy for improving KG quality and usability.

The contributions of this thesis are summarized as follows:

- We propose two novel LLM-based KG refinement methods: *LLM_sim* that heavily considers the semantic consistency for triplet refinement. *LLM_rule* implicitly generates rules for interpretable refinement of KGs.
- We demonstrate the effectiveness of our proposed methods using two large-scale datasets where noise is added artificially, as well as one

dataset containing real-world noisy triplets.

- In addition, as a practical evaluation, we assess the impact of noise detection and refinement on the downstream task of KG. The results show that refined KGs significantly enhance downstream KGC performance, highlighting the practical value of noise detection and refinement.

1.3 Research Questions

Our major research question is as follows.

How can LLMs be effectively leveraged to detect and refine noise in KGs, while maintaining both accuracy and interpretability?

To answer this overarching research question, we explore two approaches: a semantic-based refinement strategy, *LLM_sim*, that exploits contextual embeddings to identify and repair noisy triples, and a rule-based refinement strategy, *LLM_rule*, that induces interpretable logical rules for transparent noise detection and refinement. Furthermore, we examine how these refined KGs affect downstream KGC tasks to validate the practical utility of our framework.

The above major research question can be decomposed into three sub-research questions as follows:

RQ1: How can LLMs be adapted to effectively detect noisy triples in KGs, i.e., distinguish between valid and erroneous facts?

RQ2: How can LLM be used to improve or refine detected noisy triples to ensure that the corrected knowledge graph maintains high factual accuracy?

RQ3: How can LLM be integrated into the noise detection and refinement process to maintain or enhance explainability and thus enable transparent reasoning about the decisions made?

1.4 Chapter Organization

As shown in Fig 1.4, this dissertation is organized into several chapters, each addressing a specific aspect of the research problem and progressively building the conceptual and empirical contributions of this study.

Chapter 1 established the context and motivation for this work. It introduced the foundational concepts of KGs and LLMs, highlighting the emerging synergy between these two technologies. The chapter further delineated the research objectives, formulated the research questions.

Chapter 2 presents a comprehensive review of the literature on noise in KGs and methods for its detection and refinement. Furthermore, it discusses

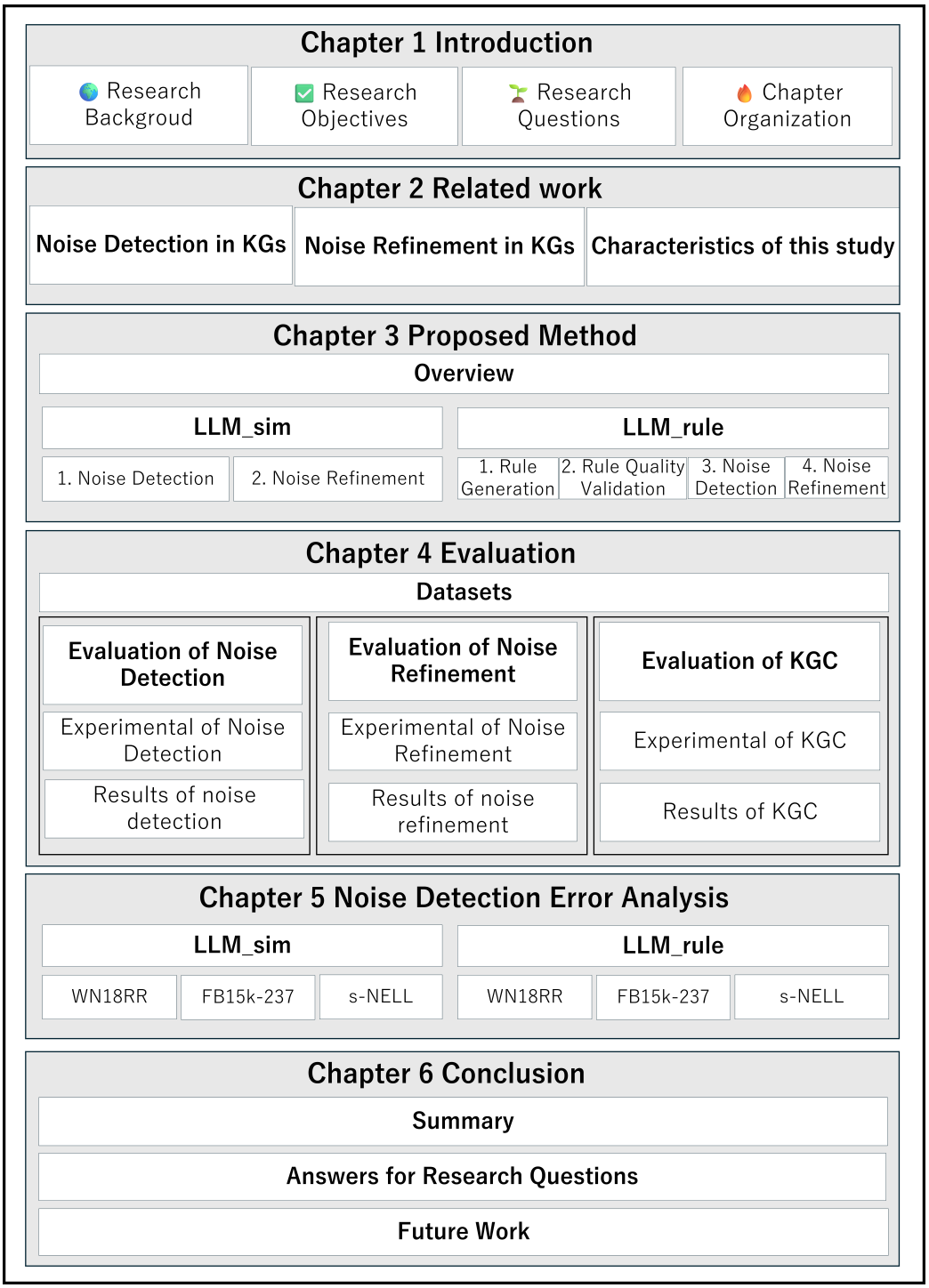

Figure 1.4: Organization of this dissertation

prior strategies for refining erroneous or missing triples, providing a critical assessment that positions this study within the broader research landscape and identifies the gaps that motivate the proposed approaches.

Chapter 3 introduces the core contributions of this dissertation. It begins with an overview of the proposed framework and then details two complementary noise detection and refinement strategies. The first, *LLM_sim*, leverages semantic similarity to identify and correct noisy triples. The second, *LLM_rule*, incorporates rule-based reasoning by generating logical rules from LLMs. Each module is described with sufficient technical detail to ensure reproducibility.

Chapter 4 provides a rigorous empirical assessment of the proposed methods. It begins with a description of the datasets used for evaluation, followed by a detailed account of the experimental settings for noise detection, noise refinement, and downstream KGC. The chapter then presents results and analyses for each evaluation stage, including comparisons with baseline models such as embedding-based KGC methods, handcrafted rules, contrastive noise filtering, and pre-trained language models. Ablation studies are also included to quantify the contributions of individual components, demonstrating the effectiveness and robustness of the proposed framework across diverse noise scenarios.

Chapter 5 provides a systematic error analysis of the proposed approaches for noise detection in KGs. This chapter adopts a qualitative perspective, aiming to uncover the systematic failure modes of both *LLM_sim* and *LLM_rule*. By examining representative cases across multiple datasets, we categorize error patterns, analyze their causes, and highlight implications for future model design.

Chapter 6 synthesizes the findings of this research. It provides a concise summary of the contributions, addresses the overarching research questions with evidence from the experiments, and discusses the implications of the results. The chapter concludes with a discussion of potential directions for future work, highlighting opportunities to scale the proposed methods, enhance robustness and interpretability, extend evaluation to additional downstream tasks, and address ethical considerations associated with LLM-augmented KG refinement.

Chapter 2

Related Work

KGs inevitably contain noise due to errors in automatic extraction, annotation errors, and temporal inconsistencies. To ensure their reliability, researchers have proposed a variety of noise handling methods, which can be broadly categorized into rule-based methods, embedding-based methods, graph neural network (GNN)-based methods, probabilistic/statistical methods, hybrid methods, and LLM-based methods. The primary goals of these methods are either noise detection or refinement, with some approaches pursuing both.

This chapter reviews previous research based on the aforementioned categories of the methods, highlighting their strengths, limitations, and relevance to detection and improvement tasks.

2.1 Sources of Noise in KGs

The sources of noise in KGs are highly diverse and stem from both technical and human-related factors. First, automatic extraction methods that derive triples from unstructured text are a major source of errors. Such pipelines often rely on natural language processing components including named entity recognition, entity linking, and relation classification. When these components fail, they may introduce incorrect entity disambiguation, mismatched relations, or spurious associations, ultimately leading to factual inaccuracies and low-quality triples in the KG [13]. Second, although crowdsourcing and human curation are widely employed to improve data quality and achieve high precision, they are not immune to error. Annotators may introduce inconsistencies due to differences in expertise, subjective interpretation, or fatigue, and such mistakes can propagate through inference chains or iterative curation cycles, thereby magnifying their impact on the graph as

a whole [37]. Third, the inherently dynamic nature of real-world knowledge contributes another important source of noise. Facts are not static but evolve over time, which means that triples once valid may later become outdated or contradictory if temporal validity is not explicitly modeled. For instance, the triple *(Barack Obama, isPresidentOf, USA)* was factually correct until 2016 but became invalid afterward, illustrating how temporal shifts can transform once-accurate knowledge into noise [14]. Taken together, these varied sources of noise highlight the necessity of robust methodologies that can both identify erroneous or inconsistent triples and refine them in a systematic manner, thereby preserving the accuracy, consistency, and long-term usability of KGs at scale.

2.2 Previous Survey of Noise Detection and Refinement in Knowledge Graph

Noise detection focuses on identifying erroneous, inconsistent, or unreasonable triples, typically by applying plausibility scores, logical constraints, or structural verification. It serves as a diagnostic, flagging suspicious triples but not necessarily providing corrective suggestions. The goal is to separate trustworthy knowledge from noisy content, providing a foundation for subsequent refinement, pruning, or human validation. Noise refinement refers to the systematic process of detecting, correcting, and preventing noise in KGs with the goal of enhancing their accuracy, consistency, and completeness. It encompasses a wide range of tasks, including error correction, triple validation, entity alignment, relation disambiguation, and constraint enforcement, thereby ensuring that KGs remain reliable and usable for downstream applications such as reasoning, question answering, and integration.

The work by Paulheim introduced one of the earliest systematic surveys on KG refinement [50], defining it as a process encompassing error correction, information completion, and redundancy removal. The survey also categorized refinement strategies into schema-based, data-driven, and hybrid approaches, and discussed typical noise sources such as extraction errors, incorrect typing, and relational inconsistencies. While highly influential in establishing conceptual boundaries and taxonomy, the survey is limited by its descriptive nature, offering little methodological innovation, empirical comparison, or practical benchmarking across datasets and domains.

Subagdja et al. provided a comprehensive survey focusing on machine learning approaches for KG refinement [67], covering symbolic learning, embedding-based methods, reinforcement learning strategies, and interactive or semi-

supervised pipelines. The survey further highlights applications in triple correction, completion, and constraint validation, and discusses the role of supervision signals and external knowledge sources. However, the limitation of this work lies in its scope, as it primarily summarizes existing techniques without introducing novel models, theoretical unification, or standardized evaluation protocols, making it less instructive for method development.

Zhong et al. offered a broad survey on automatic KG construction [91], explicitly identifying refinement as one of the core stages alongside acquisition and evolution. Their survey decomposes refinement into sub-tasks such as entity disambiguation, relation validation, triple filtering, and temporal updates, and situates these within the broader KG lifecycle. Although useful for contextualization, the limitation is that refinement is not its primary focus, resulting in less technical depth regarding detection and correction mechanisms, model architectures, and empirical performance in noisy or heterogeneous KGs.

Zhang et al. reviewed neural, symbolic, and hybrid reasoning methods for KGs, emphasizing their role in both noise handling and refinement tasks [85]. The survey systematically categorizes neural methods such as graph embedding models and GNN-based architectures, symbolic approaches grounded in logical rules and ontological constraints, and hybrid frameworks that integrate statistical signals with interpretable reasoning. It also discusses how these paradigms contribute to tasks such as triple validation, conflict detection, relation inference, and schema enhancement. While informative and conceptually comprehensive, the limitation is the lack of empirical comparison across techniques in real-world noisy environments. Specifically, the survey does not benchmark methods on heterogeneous KGs with varying noise distributions, nor does it analyze scalability, adaptability, or robustness under weak supervision or incomplete schemas.

2.3 Rule-Based Approaches

Rule-based approaches constitute a longstanding and well-studied family of methods for KG refinement, validation, and inference. These approaches typically learn or encode symbolic rules that express regularities in the KG and then apply those rules to (i) detect inconsistencies, (ii) infer missing facts, or (iii) propose corrections for suspicious triples. A primary advantage of rule-based methods is interpretability: validation and refinement decisions can be traced to explicit rules and supporting instantiations, which is particularly important for high-stakes or human-in-the-loop applications. Although many of these methods were originally developed for link prediction or reasoning,

they have also been adapted for detecting erroneous triples in noisy KGs.

The AMIE framework, and its optimized variant AMIE+, represent one of the most influential families of rule-based approaches for mining logical regularities from large-scale KGs [24]. At their core, these systems automatically discover Horn-like rules of the form

$$r_1(x, y) \wedge r_2(y, z) \Rightarrow r_3(x, z), \tag{2.1}$$

where the rule body is composed of multi-hop relational patterns and the head denotes a predicted fact. AMIE employs a breadth-first rule exploration strategy, iteratively extending rule bodies by adding atoms and evaluating them against the KG. Candidate rules are then ranked using scoring heuristics such as *standard confidence*, defined as the ratio of correct predictions among all predictions made by the rule, and *PCA confidence*, which corrects for the incompleteness of KGs by assuming partial closed-world semantics. The variant AMIE+ introduces several optimizations to improve scalability and precision. In particular, it adopts aggressive pruning strategies to discard low-confidence or low-support candidates early in the search process, and leverages efficient indexing structures to accelerate rule evaluation. These enhancements make AMIE+ applicable to industrial-scale KGs containing millions of triples, while still maintaining computational efficiency. However, both AMIE and AMIE+ depend heavily on sufficient pattern support, which makes them less effective in sparse regions of a KG where relational evidence is limited. Moreover, the discovered rules can be overly specific or brittle, for example when rules contain too many conjunctive atoms, resulting in low recall and poor generalization to novel or heterogeneous domains. In the context of noisy KGs, such rules can be exploited for noise detection, since triples that systematically violate high-confidence rules are likely to represent erroneous or inconsistent facts.

AnyBURL (Anytime Bottom-Up Rule Learning) is another representative rule-based approach that focuses on learning Horn rules for KGs in an efficient and scalable manner [43]. Unlike AMIE, which explores rules in a top-down fashion, AnyBURL employs a bottom-up sampling strategy. Specifically, it begins by randomly sampling concrete paths in the KG, such as

$$(h, r_1, e_1), (e_1, r_2, e_2), \ldots, (e_{k-1}, r_k, t), \tag{2.2}$$

and then generalizes these paths into variable-containing Horn rules by replacing entities with variables. For instance, the path

$$(\text{Obama}, \text{bornIn}, \text{Hawaii}) \wedge (\text{Hawaii}, \text{locatedIn}, \text{USA})$$

can be generalized into the rule

$$\text{bornIn}(x, y) \wedge \text{locatedIn}(y, z) \Rightarrow \text{nationality}(x, z). \tag{2.3}$$

A key feature of AnyBURL is its anytime property, meaning that the algorithm can be interrupted at any point while still producing a set of usable rules. Over time, as more samples are drawn and generalized, the quality and coverage of the learned rules progressively improve. This makes AnyBURL particularly suitable for large-scale and dynamic KGs where computational budgets may be limited. Moreover, by focusing on frequently sampled paths, the method is able to quickly discover high-precision rules that capture common relational regularities. The rules generated by AnyBURL have also been used for noise detection, as they provide interpretable patterns against which suspicious triples can be validated. However, the performance of AnyBURL strongly depends on the sampling strategy and the representativeness of the observed paths. Low-frequency relations or long-range dependencies may be underrepresented in the sampled paths, leading to limited recall and weaker generalization across domains. Furthermore, although the bottom-up approach improves scalability, the resulting rules can still suffer from redundancy and may require extensive post-processing to filter out spurious or low-quality patterns.

The Path Ranking Algorithm (PRA) and related path-centric techniques model multi-hop connectivity patterns as features for relation prediction, effectively learning which sequences of relations imply a target relation [36]. Given a candidate triple (h, r, t), PRA searches for relational paths linking h and t, and uses the frequency and reliability of these paths as features for classification. Concretely, each distinct path type is treated as a binary feature indicating whether it connects the head and tail, and these features are then fed into a logistic regression classifier to predict the plausibility of the triple. From the perspective of *noise detection*, a triple is likely to be erroneous if no reliable or semantically meaningful paths connect its head and tail entities. PRA is intuitive, as paths can be interpreted as soft rules, and has been successfully applied to both inference and identifying suspicious triples. However, the set of candidate paths can grow combinatorially with graph size, making exhaustive enumeration computationally expensive. Moreover, path-based features do not directly yield compact logical rules and can be highly sensitive to noisy or spurious paths, which may cause false positives in noise detection.

A subsequent line of work integrates symbolic rules with statistical models, for example, by imposing logical constraints during embedding learning or by coupling rule application with neural inference [40]. Representative systems include NeuralLP, which encodes first-order logic rules into

differentiable objectives [83]. In such frameworks, rules are translated into soft constraints or regularization terms in the loss function, thereby forcing the learned embeddings to satisfy logical patterns. For instance, if a rule $r_1(x, y) \wedge r_2(y, z) \Rightarrow r_3(x, z)$ is known, a penalty is introduced whenever the embeddings violate this constraint. This hybrid paradigm attempts to combine the interpretability of symbolic rules with the generalization ability of continuous embeddings. Nevertheless, balancing symbolic constraints with statistical objectives is non-trivial: the optimization process may overfit to one objective at the expense of the other, and integration often increases computational cost and engineering complexity.

More recently, Zhao et al. proposed TRAIL, a framework that integrates dynamic validation into LLM reasoning [90]. TRAIL performs iterative validation and pruning of generated triples by combining LLM predictions with confidence-based filtering. In each iteration, newly generated candidate facts are validated against confidence estimates, and low-confidence triples are pruned from the pool. This process repeats until the fact set converges to a stable and reliable subset. Such iterative refinement enables both inference and noise detection, since systematically invalid triples are discarded during successive validation rounds. Its main limitation lies in its reliance on accurate confidence estimation. If these scores are miscalibrated, valid knowledge may be incorrectly discarded, reducing recall.

Significant efforts have been made to improve scalability and robustness in real-world settings, proposing advances in indexing, sampling, parallelization, and rule pruning [47, 24]. These engineering optimizations make rule learning feasible for industrial-scale KGs and support downstream refinement tasks. Nonetheless, achieving high coverage and high precision simultaneously across heterogeneous domains remains a challenge. Many scalable systems sacrifice interpretability or generate rules that require extensive post-filtering or human validation to ensure reliability.

2.4 Embedding-Based Approaches

Beyond rule-based methods, a major line of research on noise detection and refinement in KGs leverages representation learning through embedding models. The core idea is to encode entities and relations into continuous vector spaces, where geometric operations reflect semantic plausibility. Embedding-based approaches have the advantage of scalability and the ability to generalize from sparse observations, making them attractive for large-scale and automatically constructed KGs. However, their reliance on latent representations often comes at the cost of interpretability, and their ability to pinpoint

the source or nature of noise is limited. In what follows, we review representative methods and discuss their respective capabilities in noise detection and refinement.

Bordes et al. introduced the TransE model, one of the earliest and most influential embedding approaches for knowledge representation learning [12]. In TransE, relations are modeled as translation vectors in the embedding space, and the plausibility of a triple (h, r, t) is evaluated by the distance function

$$f(h, r, t) = \|\mathbf{h} + \mathbf{r} - \mathbf{t}\|. \tag{2.4}$$

From the perspective of noise handling, triples with very low plausibility scores can be flagged as potential noise (detection), while alternative entities that minimize this distance may serve as candidate replacements (refinement). This simple yet elegant formulation made TransE highly scalable and effective on benchmark datasets. Nevertheless, its limitation lies in handling complex relational patterns such as one-to-many, many-to-one, and many-to-many relations, where the translation assumption is too restrictive and can lead to distorted embeddings. As a result, its ability to validate and repair noisy triples in realistic KGs with heterogeneous relations is reduced.

Building upon TransE, various extensions have been proposed to capture richer relational semantics. For instance, Wang et al. developed TransH [76], which allows entities to have different representations under different relations by projecting them onto hyperplanes specific to each relation. This extension alleviates some of the limitations of TransE in handling multi-mapping relations, yet it still inherits the translation-based assumption and introduces additional hyperparameters that complicate training. Similarly, TransR [38] employs relation-specific spaces for projecting entities, improving expressivity but at the cost of increased model complexity and sensitivity to parameter tuning. Both models extend the potential for detecting noise by better modeling relation-specific interactions, while also improving refinement accuracy by providing more precise candidate replacements.

To move beyond translation-based models, latent factorization approaches were introduced. RESCAL, proposed by Nickel et al. [46], employs a tensor factorization framework where entities are represented as vectors and relations as full-rank matrices. Its scoring function is defined as

$$f(h, r, t) = \mathbf{h}^\top \mathbf{M}_r \mathbf{t}, \tag{2.5}$$

where $\mathbf{M}_r$ is a relation-specific matrix. This formulation enables RESCAL to capture complex and asymmetric relational patterns, providing stronger detection ability for erroneous triples that violate such patterns. However, its quadratic parameterization with respect to the number of entities and

relations leads to scalability issues on large KGs. Moreover, the lack of strong regularization mechanisms makes RESCAL prone to overfitting and less effective in robustly repairing noise during refinement.

Another prominent family of models are bilinear and complex-valued embeddings. Trouillon et al. proposed ComplEx [71], which extends bilinear models into the complex domain. By representing entities and relations as complex vectors, ComplEx employs the scoring function

$$f(h, r, t) = \Re\langle \mathbf{h}, \mathbf{r}, \bar{\mathbf{t}} \rangle, \tag{2.6}$$

where $\bar{\mathbf{t}}$ is the complex conjugate of $\mathbf{t}$. This enables the model to capture both symmetric and antisymmetric relations, addressing a major limitation of earlier bilinear approaches such as DistMult [82]. From a noise-handling perspective, ComplEx can effectively detect implausible triples across diverse relation types, while also supporting refinement by providing semantically grounded candidate replacements. However, its reliance on dense embeddings makes it difficult to provide interpretable explanations for repair decisions, and its performance may degrade when noise distributions deviate from the training data.

Arora et al. introduced IterefinE, which combines probabilistic soft logic reasoning with embedding models in an iterative framework [3]. This method effectively integrates symbolic and sub-symbolic knowledge to enhance KG quality. However, the limitation is its complexity and limited scalability to very large graphs. Saeedizade et al. developed KGRefiner, which introduces auxiliary nodes based on ontological knowledge to improve translational embedding models [60]. This approach increases prediction accuracy, but it is constrained by dependency on high-quality ontologies, which are not always available. Zhao et al. proposed BioGRER, a biomedical KG refinement approach combining embeddings with logical rules in a variational EM framework [89]. While effective in biomedical contexts, its limitation is domain specificity, making it less generalizable to heterogeneous KGs.

Ge et al. proposed KGClean, a framework powered by graph embeddings and active learning for detecting and repairing noisy triples [25]. The method first learns low-dimensional representations of entities and relations, which are used to identify triples with low plausibility scores or structural inconsistencies. To improve detection precision with limited supervision, KGClean employs an active learning strategy that selectively queries labels for the most uncertain or influential triples, thereby reducing annotation costs while refining the decision boundary. For the repair phase, the framework either removes invalid triples or replaces them with more plausible candidates inferred from embedding proximity or relational patterns. This two-stage

pipeline allows KGClean to iteratively enhance KG quality while leveraging feedback to correct model biases. Despite its robustness, the limitation is its sensitivity to the choice of embedding model, which may affect consistency across domains. Different embedding techniques capture varying relational structures and semantic assumptions, making performance less stable when applied to heterogeneous KGs or transferred across application settings.

More recently, expressive embedding architectures such as ExpressivE [51] have been proposed. ExpressivE generalizes traditional embedding models by introducing box embeddings that represent entities and relations as multi-dimensional regions rather than points or vectors. This allows ExpressivE to capture a wider range of logical patterns and relational dependencies, offering stronger representational power for modeling complex knowledge structures. For noise detection, ExpressivE demonstrates high accuracy in identifying anomalous triples, and for refinement, it provides more precise alternatives by modeling structured dependencies. Nevertheless, its increased expressivity comes with higher computational cost and training complexity.

In addition to deterministic embeddings, probabilistic and uncertainty-aware approaches have been proposed to explicitly model noise. For example, KG2E [28] represents entities and relations as Gaussian distributions rather than point vectors, thereby capturing uncertainty in the embedding space. The similarity between triples is then measured by the Kullback–Leibler (KL) divergence between distributions. This probabilistic modeling enables KG2E to detect noisy or incomplete triples through high uncertainty values, while refinement can be guided by selecting replacements with lower uncertainty estimates. However, the computational cost of distributional representations is significantly higher, and the interpretability of uncertainty estimates remains limited in practice.

Embedding-based approaches provide a scalable and flexible mechanism for both noise detection and refinement. By mapping entities and relations into continuous vector spaces and defining plausibility scoring functions $f(h, r, t)$, they can identify suspicious triples with low scores and suggest alternative head or tail entities with higher scores. Their advantages include scalability to very large KGs, generalization from sparse data, and adaptability to diverse relational patterns. However, the reliance on latent representations limits interpretability, making it difficult to explain why a triple is flagged as noisy or why a specific correction is proposed. Moreover, their effectiveness is highly dependent on the quality and coverage of training data: in highly noisy or sparse KGs, embeddings may amplify existing biases rather than mitigate them. Consequently, embedding-based approaches are often most effective when combined with symbolic or rule-based constraints that provide explicit interpretability and logical grounding.

2.5 Hybrid and Neuro-Symbolic Approaches

Beyond purely rule-based and embedding-based strategies, a growing body of research has explored hybrid and neuro-symbolic approaches for noise detection and refinement in KGs. These methods aim to combine the interpretability of symbolic reasoning with the scalability and representation power of neural embeddings, thereby addressing limitations inherent in either paradigm alone. In particular, they seek to leverage explicit logical rules or constraints while simultaneously benefiting from the robustness of latent vector representations, offering a promising pathway toward both accuracy and explainability.

One representative line of work integrates embedding models with rule-based reasoning by injecting first-order logic constraints into neural objectives. Rocktäschel et al. proposed a framework that translates logical rules into differentiable penalties added to the embedding loss [58]:

$$L = L_{\text{embed}} + \lambda \sum_{r \in \mathcal{R}} \text{penalty}(r), \tag{2.7}$$

where L_{embed} is the embedding loss and $\text{penalty}(r)$ enforces the satisfaction of rule r. This integration encourages embeddings to respect logical regularities while still capturing distributional semantics. From a noise-handling perspective, implausible triples that violate high-confidence rules can be detected, although explicit refinement (i.e., suggesting corrected triples) is limited by the reliance on predefined rules. The scalability of this approach is further constrained by the quality and coverage of the available rule set.

To reduce dependence on manually crafted rules, Guo et al. introduced a joint framework that iteratively integrates embedding updates with rule induction [26]. In this pipeline, candidate rules are mined from the KG, filtered, and then used to regularize embeddings, while updated embeddings in turn improve subsequent rule mining. This alternating optimization reduces human supervision and adapts better to heterogeneous domains. For noise detection, spurious triples are filtered when they fail to align with both induced rules and embedding scores. For refinement, embeddings provide plausible candidates consistent with induced patterns. Nevertheless, rule induction itself may introduce noisy or brittle patterns, and without careful filtering, errors can propagate across iterations.

Another research direction integrates symbolic reasoning paths with neural policy learning. Das et al. proposed MINERVA, a reinforcement learning model for KG question answering that learns to traverse paths guided by both symbolic constraints and embedding-based rewards [20]. Given a query

$(h, r, ?)$, MINERVA incrementally chooses outgoing edges to reach the correct tail t, optimizing a policy that maximizes expected rewards:

$$J(\theta) = \mathbb{E}_{\pi_\theta} \left[\sum_t R(s_t, a_t) \right], \tag{2.8}$$

where R measures whether the traversal reaches the correct answer. For noise detection, implausible triples correspond to traversal failures or paths with low policy probability. However, refinement is limited because the model mainly focuses on path selection rather than candidate generation. Moreover, the learned policies may overfit to frequent paths, reducing robustness for rare or noisy relations.

Recent advancements have also explored the intersection of LLMs and factual verification. For instance, Sornlertlamvanich et al. investigated methods for verifying the correctness of LLM-generated content by leveraging external knowledge and semantic reasoning [65]. Specifically, they proposed a Semantically Aware Reasoning (SAR) framework that mimics human "System 2" analytical thinking to augment the generative capabilities of LLMs ("System 1"). By employing Semantic Textual Similarity (STS) metrics, their approach rigorously compares LLM-generated outputs against retrieved external knowledge to ensure semantic alignment. This methodology highlights the critical importance of cross-referencing generative outputs with structured evidence to mitigate hallucinations, aligning with the growing trend of utilizing LLMs as active verifiers in the knowledge engineering pipeline.

Chen et al. presented PatReFormer, a transformer-based patch refinement model that mitigates biases in linear and convolutional approaches for KG completion by introducing localized attention over triple embeddings and relation-specific contexts [18]. Instead of relying on global representations, PatReFormer adopts a patch-wise encoding mechanism that captures fine-grained semantic interactions between entities and relations, enabling more precise correction of structurally inconsistent or low-plausibility triples. This design improves the expressiveness of the refinement process, particularly in scenarios where long-range dependencies or subtle semantic variations are overlooked by conventional models. However, its limitation lies in the increased computational cost and training complexity caused by multi-head self-attention, larger parameterization, and the need for extensive negative sampling to ensure convergence.

Bikaun et al. developed CleanGraph, an interactive system that integrates human feedback with model predictions for KG refinement and completion [8]. The framework operates in a human-in-the-loop fashion, where candidate erroneous or incomplete triples are first identified by automated

predictors, and users can approve, modify, or reject suggestions through an intuitive interface. CleanGraph further supports incremental updates by iteratively retraining or adjusting its internal models based on user revisions, which improves adaptation to domain-specific knowledge and reduces the risk of overcorrection. This design notably enhances transparency, interpretability, and controllability, allowing practitioners to directly intervene in the refinement process and correct system biases that may arise from purely automated approaches. However, the limitation is reliance on sustained user engagement, as large-scale or high-frequency refinement tasks may require substantial manual involvement. This dependency limits scalability in scenarios where expert supervision is costly or infeasible, and the effectiveness of the system may vary depending on the quality and availability of human feedback.

Recent work explored multimodal KG completion through KICGPT, which integrates structural signals from KGs with contextualized textual information from large language models (LLMs) to improve triple prediction [78]. A distinctive feature of this model is its design of a knowledge-in-context mechanism, which dynamically incorporates graph-based neighborhood structures, entity descriptions, and relation semantics into the reasoning process. By aligning pre-trained language representations with KG embeddings, the model is able to attend to both symbolic structure and unstructured text, enabling more expressive reasoning across modalities and capturing triples that are difficult to infer from graph structure alone. However, despite these advantages, the method suffers from substantial computational overhead due to the joint encoding of graph and textual features, as well as the cost of prompt construction and inference through LLMs. Additionally, it faces challenges in achieving robust modality alignment, particularly when entity descriptions are noisy or incomplete, which restricts its scalability and generalizability in large or domain-specific real-world KGs.

A more recent advance is DRUM (Differentiable Rule Mining), which jointly learns symbolic rules and embeddings in an end-to-end manner [59]. Unlike two-step pipelines, DRUM discovers Horn-like rules through gradient-based optimization while simultaneously updating embeddings. This tight integration enables both detection (via rule violations) and refinement (via rule-guided candidate generation). Nevertheless, due to optimization constraints, DRUM tends to favor short and simple rules, limiting its ability to capture long reasoning chains. Its performance also deteriorates in highly complex domains with deeply nested dependencies.

Hybrid and neuro-symbolic approaches represent a promising middle ground between symbolic precision and neural scalability. They improve robustness to noise and provide more interpretable validation compared to pure embed-

ding models, while offering better scalability than purely rule-based systems. For noise detection, they benefit from combining plausibility scoring with rule enforcement. For refinement, they can propose candidates guided by both embeddings and symbolic constraints. However, their effectiveness remains dependent on rule quality, they incur nontrivial computational overhead, and conflicts between symbolic and sub-symbolic signals can reduce stability. As such, they provide valuable building blocks but are not yet a comprehensive solution to KG noise handling.

2.6 GNN-based Methods

GNN-based methods represent a rapidly growing line of work for KG refinement and noise detection. These models extend deep learning to multi-relational graphs by propagating and transforming information across typed edges, thereby producing context-aware representations of entities and relations. In principle, such methods can exploit structural context to attenuate local errors and highlight anomalous triples. A key strength lies in their ability to model complex relational dependencies in an end-to-end fashion. However, they also face challenges with scalability, interpretability, and noise propagation, especially when applied to large, automatically constructed KGs.

Schlichtkrull et al. introduced Relational Graph Convolutional Networks (R-GCN), which extend standard GCNs to multi-relational settings by defining relation-specific transformations and parameter sharing via basis and block-diagonal decompositions [61]. The message passing in R-GCN is defined as:

$$h_i^{(l+1)} = \sigma\left(\sum_{r\in\mathcal{R}}\sum_{j\in\mathcal{N}_i^r}\frac{1}{c_{i,r}}W_r^{(l)}h_j^{(l)} + W_0^{(l)}h_i^{(l)}\right), \tag{2.9}$$

where $\mathcal{N}_i^r$ denotes the neighbors of node i under relation r, $c_{i,r}$ is a normalization constant, and $W_r^{(l)}$ are relation-specific weights. R-GCN enables entity classification and link prediction by aggregating relational neighborhood signals. For noise detection, low-scoring triples can be flagged as implausible if their aggregated neighborhood representations are inconsistent. However, R-GCN tends to oversmooth in dense subgraphs and becomes computationally expensive for graphs with large relation vocabularies, while offering limited transparency for refinement decisions.

Shang et al. proposed **Structure-Aware Convolutional Networks (SACN)**, which combine graph convolutional encoders with ConvE-style decoders to capture multi-hop relational structures [62]. SACN uses a GCN

to aggregate neighborhood features and then applies a ConvE-based decoder to score triples, effectively modeling both structural and semantic features. By jointly leveraging structural encoders and convolutional scoring functions, SACN improves link prediction accuracy and robustness to incomplete graphs. From a noise-handling perspective, SACN can identify erroneous triples when convolutional scores and structural encodings diverge, providing implicit anomaly signals. However, it inherits the interpretability limitations of convolutional decoders and is sensitive to noise introduced during negative sampling, which may bias predictions and hinder reliable refinement.

Nathani et al. developed **KBAT**, which applies multi-head attention over relation-specific neighborhoods to prioritize informative neighbors while down-weighting noisy ones [45]. Formally, the attention mechanism computes weights over neighbors $j \in \mathcal{N}_i^r$ as

$$\alpha_{ij}^r = \frac{\exp(e_{ij}^r)}{\sum_{k \in \mathcal{N}_i^r} \exp(e_{ik}^r)},$$

where e_{ij}^r measures the compatibility of neighbor j under relation r. These weights allow the model to act as a soft denoising mechanism, highlighting suspicious triples with low neighbor support.

It is particularly worth noting that the attention mechanism itself can function as a powerful, intrinsic noise detection signal beyond specific models like KBAT. In general Graph Attention Networks (GATs) or Transformer-based KG encoders, the learned attention weights quantify the semantic relevance between connected nodes. Empirical observations suggest that noisy, irrelevant, or factually inconsistent edges typically receive significantly lower attention scores compared to valid relations during the aggregation process. Consequently, analyzing the distribution of these attention weights provides a data-driven heuristic to identify and prune likely noisy triples, even in the absence of explicit supervision.

Nonetheless, attention distributions are not faithful causal explanations and may still overfit to spurious correlations, limiting their reliability for refinement.

Vashishth et al. introduced **Compositional GCN (CompGCN)**, which composes entity and relation embeddings during message passing to better model relational properties such as directionality and symmetry [72]. The key idea is to define a composition operator $\phi(h, r)$ (e.g., circular correlation, subtraction) that combines an entity embedding h and relation embedding r before neighborhood aggregation. This design enables CompGCN to capture asymmetric and complex relation types, enhancing expressivity for link prediction and anomaly detection. For refinement, CompGCN provides more

accurate candidate scoring by modeling structural dependencies. However, the additional computations increase training costs and the latent compositions remain opaque to human inspection.

Teru et al. presented **GraIL**, a framework that generalizes to unseen entities by learning from enclosing subgraphs and structural identifiers [70]. GraIL extracts the k-hop enclosing subgraph around a candidate triple (h, r, t), encodes it using GNNs, and applies relation-specific labeling to distinguish different roles (head, tail, and context nodes). This subgraph-centric reasoning aligns naturally with anomaly detection, as enclosing subgraphs provide structural context for evaluating triple plausibility. Nevertheless, extracting enclosing subgraphs is computationally costly at scale, and purely structural features may fail to distinguish rare but correct patterns from genuine noise, raising the risk of false positives in refinement.

Zhu et al. proposed **Neural Bellman-Ford Networks (NBFNet)**, which model path-based reasoning as a differentiable dynamic program [92]. NBFNet adapts the Bellman-Ford algorithm, which iteratively updates shortest path distances, into a neural architecture that propagates relation-aware messages along paths. The network learns to aggregate path scores through differentiable recurrence, enabling robust multi-hop reasoning. For noise detection, triples with weak or inconsistent path evidence are flagged as anomalous. However, training stable path aggregators requires careful tuning, and hub nodes may dominate flow computations, reducing robustness in noisy graphs.

The MERIT framework was introduced as a multi-level refinement model for large-scale graphs [17]. Instead of operating directly on the full KG, MERIT performs *graph coarsening* to cluster structurally or semantically related entities into super-nodes, thereby reducing the computational burden in subsequent stages. On the compressed graph, it generates embeddings through scalable representation learning, and then progressively refines the KG by propagating corrections back from the coarse level to the original fine-grained structure. This hierarchical design enables efficient processing of millions of triples and alleviates the scalability bottlenecks faced by traditional refinement pipelines. However, the limitation is the potential loss of fine-grained semantics during coarsening, as entities grouped into a single cluster may differ in local relational patterns. Moreover, refinement decisions made at the coarse level may not always transfer accurately to the original graph, especially when subtle contextual distinctions are essential for disambiguation.

Zhang and Chen introduced **SEAL**, which encodes k-hop enclosing subgraphs using GNNs for link prediction [86]. SEAL first extracts enclosing subgraphs around candidate triples, then applies node labeling (distance from

head/tail) to preserve role information before feeding the subgraph to a GNN encoder. This approach captures rich local motifs and structural dependencies, making it effective at detecting links with atypical local structures and identifying anomalous triples. However, scalability remains a barrier for large hop sizes, and motif rarity does not always equate to factual error, raising the risk of false positives during refinement.

GNN-based methods offer strong modeling capacity for noise detection by leveraging structural context, multi-hop reasoning, and neighborhood information. They provide richer contextual signals than embedding-only methods and can implicitly act as denoisers through attention or path aggregation. For refinement, they enable scoring of alternative candidates with improved structural awareness. Nonetheless, they face notable challenges: noise can propagate through message passing, interpretability remains limited, and computational costs grow rapidly with graph size and relation complexity. As such, while GNN-based approaches are powerful tools for KG noise handling, they are most effective when complemented with symbolic or probabilistic mechanisms that provide explicit interpretability and noise robustness.

2.7 Probabilistic and Statistical Relational Models

Probabilistic and statistical relational models offer a complementary line of work for noise detection and refinement in KGs. Unlike deterministic rule-based or embedding-based methods, these approaches explicitly model uncertainty in facts and relational dependencies. By assigning probabilistic semantics to triples or rules, they provide a principled way to capture noise, incompleteness, and inconsistency, making them particularly well-suited for knowledge graphs constructed from heterogeneous sources.

A foundational example is **Markov Logic Networks (MLNs)** [56], which combine first-order logic with Markov random fields. In MLNs, a set of weighted first-order logic formulas defines a probability distribution over possible KG worlds:

$$P(X) = \frac{1}{Z} \exp\left(\sum_i w_i n_i(X)\right), \tag{2.10}$$

where w_i is the weight of rule i, $n_i(X)$ is the number of satisfied groundings in world X, and Z is the normalization constant. For noise detection, triples that consistently violate high-weight rules obtain low probabilities. For refinement, MLNs can suggest alternative assignments that maximize the joint

likelihood. However, inference in MLNs is computationally expensive and often intractable for large KGs, limiting scalability.

Probabilistic Soft Logic (PSL) provides a scalable alternative by relaxing discrete logic into continuous truth values in the interval $[0, 1]$ [5]. PSL defines rules as weighted linear constraints and infers soft truth assignments through convex optimization:

$$\min_{Y \in [0,1]^n} \sum_{r \in \mathcal{R}} w_r \, \phi_r(Y), \tag{2.11}$$

where $\phi_r(Y)$ is the hinge-loss potential of rule r. This formulation allows efficient inference while tolerating noisy or uncertain triples. For noise detection, triples with consistently low soft truth values are flagged as implausible, while refinement can be guided by adjusting assignments toward satisfying the most consistent rules. Nevertheless, PSL relies on careful rule specification, and performance degrades if critical rules are missing or misweighted.

Bayesian and graphical-model approaches such as relational dependency networks and Markov random fields [34] have also been applied to noisy KGs. These models define conditional probability distributions over triples and estimate them from data. In detection, triples with very low posterior probability are considered noise, while refinement involves sampling or inferring more likely replacements. Although statistically principled, these models are computationally heavy and face difficulties with high-dimensional relational structures.

Uncertainty-aware embedding models explicitly combine representation learning with probabilistic semantics. For example, KG2E [28] represents entities and relations as Gaussian distributions instead of point vectors. The plausibility of a triple is measured by the KL-divergence between head–relation and tail distributions as follows:

$$f(h, r, t) = D_{\mathrm{KL}}(\mathcal{N}(\mu_h + \mu_r, \Sigma_h + \Sigma_r) \parallel \mathcal{N}(\mu_t, \Sigma_t)) \, . \tag{2.12}$$

High divergence values indicate anomalous triples (detection), while refinement can be guided by selecting replacements that minimize divergence. Such probabilistic embeddings are more robust to noise and incompleteness, but incur higher computational costs and offer limited interpretability for uncertainty estimates.

Probabilistic and statistical relational models provide a principled framework for handling uncertainty in KGs. They enable both noise detection, by assigning low probabilities to implausible triples, and noise refinement, by suggesting alternative configurations that maximize joint likelihood. Their strengths include the ability to model structured dependencies and to incorporate prior knowledge in a probabilistic setting. However, scalability

remains a central challenge, as inference is often costly, and the interpretability of probabilistic scores is limited. Consequently, these models are often combined with embeddings or symbolic rules to achieve a better trade-off between scalability, robustness, and explainability in real-world noisy KGs.

2.8 Characteristics of this study

This dissertation is characterized by several distinctive aspects that collectively advance the state of research on noisy KGs. Unlike prior work that has typically focused on isolated components or relied on idealized settings, our study proposes two independent frameworks, *LLM_sim* and *LLM_rule*, that systematically address both noise detection and refinement in a realistic and interpretable manner. The key characteristic of this work is LLM-augmented semantic and symbolic reasoning, which is outlined as follows.

Existing KG refinement methods largely rely on embeddings or manually defined symbolic rules. While embeddings provide scalability, they lack interpretability and struggle with heterogeneous noise. On the other hand, symbolic rule systems demand substantial human curation and are brittle in sparse graph regions. Our study addresses both limitations by leveraging LLMs in two ways: *LLM_sim* uses LLMs to generate candidate triples and evaluates them through semantic similarity with KG context, while *LLM_rule* prompts LLMs to generate symbolic rules tailored to relational patterns. To our knowledge, this is the first work that exploits LLMs for *automatic rule generation for noise detection and refinement*, thereby reducing manual effort and enhancing the flexibility of rule-based refinement. By grounding refinements either in similarity scores or in explicit rules, our frameworks balance scalability and interpretability.

Through the contributions, this dissertation advances the methodological, experimental, and interpretability dimensions of KG noise handling, thereby positioning itself as a comprehensive study in the field.

Chapter 3

Proposed Method

3.1 Overview

Let us suppose that a KG is formally represented as a set of triples (h, r, t), where h and t denote the head and tail entities, respectively, and r denotes the relation linking them. The central objective of this study is to transform a noisy KG into a renewed version KG' by systematically identifying and correcting erroneous triplets, thereby improving its reliability and usability for downstream tasks.

To achieve this objective, we propose two methods, namely *LLM_sim* and *LLM_rule*, both of which leverage the reasoning capabilities of LLMs for KG refinement. Despite their methodological differences, these models share a common two-stage processing pipeline: **Noise Detection** followed by **Noise Refinement**.

In the first stage, **Noise Detection**, the model evaluates each triplet in the input KG to determine its correctness. Based on this classification, the triplets are partitioned into two disjoint subsets: a set of valid triplets, denoted as KG_v, and a set of noisy or erroneous triplets, denoted as KG_n. This separation provides a foundation for subsequent refinement, ensuring that only potentially erroneous information undergoes modification.

In the second stage, **Noise Refinement**, each noisy triplet $(h, r, t) \in KG_n$ is systematically corrected. The refinement process is realized by replacing either the head or the tail entity with a more plausible alternative. Formally, the corrected triplet is represented as either (h', r, t), where the head entity is substituted, or (h, r, t'), where the tail entity is substituted. The collection of such corrected triplets forms the refined KG, denoted as KG_r.

Finally, the renewed KG, KG', is reconstructed by concatenating the

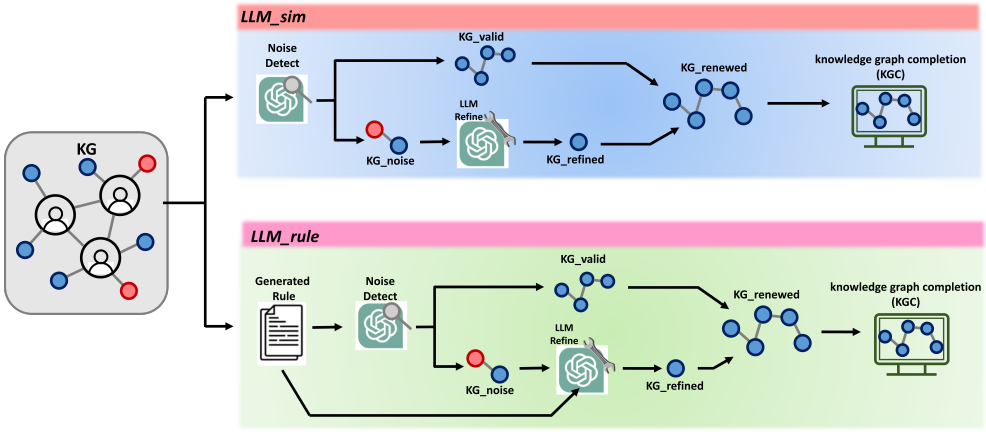

Figure 3.1: Overview of the proposed frameworks *LLM_sim* and *LLM_rule*.

valid and refined triplets, i.e.,

$$KG' = KG_v \cup KG_r. \tag{3.1}$$

This unified representation ensures that the renewed graph preserves the trustworthy information from the original KG while simultaneously repairing noisy triplets through LLM-driven refinement.

An overview of the two proposed models is illustrated in Figure 3.1. In this figure, the blue circles represent correct entities, while the red circles indicate erroneous entities. The upper panel depicts the *LLM_sim* model, which directly employs the LLM for both noise detection and refinement: noisy triplets identified by the LLM are refined by replacing either the head or tail entity, and the refined triplets are merged with valid ones to form KG'. The lower panel illustrates the *LLM_rule* model, which first generates rules via the LLM and subsequently uses these rules to guide noise detection and refinement. The refined triplets obtained through this rule-guided process are then combined with the valid triplets to produce the renewed graph. The main distinction between the two models thus lies in their refinement strategies: *LLM_sim* relies on direct similarity-based reasoning from the LLM, whereas *LLM_rule* incorporates rule generation to enhance interpretability and consistency in the refinement process.

The following subsections provide detailed expositions of the two proposed models, *LLM_sim* and *LLM_rule*, including their design principles, operational mechanisms, and contributions to the overall KG refinement pipeline.

To support both the semantic-driven and rule-based components of our framework, this study adopts Llama-3-8B-Instruct as the backbone model

for all LLM-based operations. This choice is motivated by several practical and methodological considerations.

First and foremost, the primary objective of this study is to validate the feasibility of the proposed framework for improving KG quality, rather than to benchmark the upper limits of current LLMs. While larger commercial models (e.g., GPT-4) might yield superior absolute performance, employing Llama-3-8B allows us to demonstrate that the proposed semantic and rule-based refinement strategies are effective even under resource-constrained settings. Proving the framework's effectiveness on an 8B parameter model suggests that the methodology is robust and scalable, implying that results would likely further improve with more capable models.

Second, at the time of conducting this research, Llama-3-8B-Instruct represented one of the most capable open-source instruction-tuned models. Unlike proprietary APIs where model versions may change implicitly, the open-source nature of Llama-3 ensures full transparency, controllability, and reproducibility. These properties are essential for scientific evaluation, allowing independent replication of our results without the "black box" uncertainty associated with commercial models.

Finally, the 8-billion-parameter scale provides a favorable balance between computational feasibility and reasoning quality, allowing all experiments to be conducted within standard academic computing environments without incurring the substantial financial cost associated with commercial API usage.

3.2 Semantic-Driven Approach (LLM_sim)

3.2.1 Noise Detection

Figure 3.2 shows an overview of *LLM_sim*, The green circles represent the refined entity, and the yellow circles represent the reference from the original *KG*. The first stage of *LLM_sim* focuses on identifying noisy triples in a principled and context-aware manner. To enhance the evaluation of LLM with relevant evidence from KG, we adopt a soft matching mechanism to retrieve semantically similar triples from the original KG and provide them to LLM as *ADDITIONAL_CONTEXT*. This retrieved context consists of a triple that is structurally and semantically proximate to the target triple $\langle h, r, t \rangle$, enabling the LLM to ground its judgment in the existing graph structure rather than relying solely on parametric knowledge.

As illustrated in the challenges of using PLMs, they often struggle to capture implicit or long-range relations that require reasoning beyond surface-

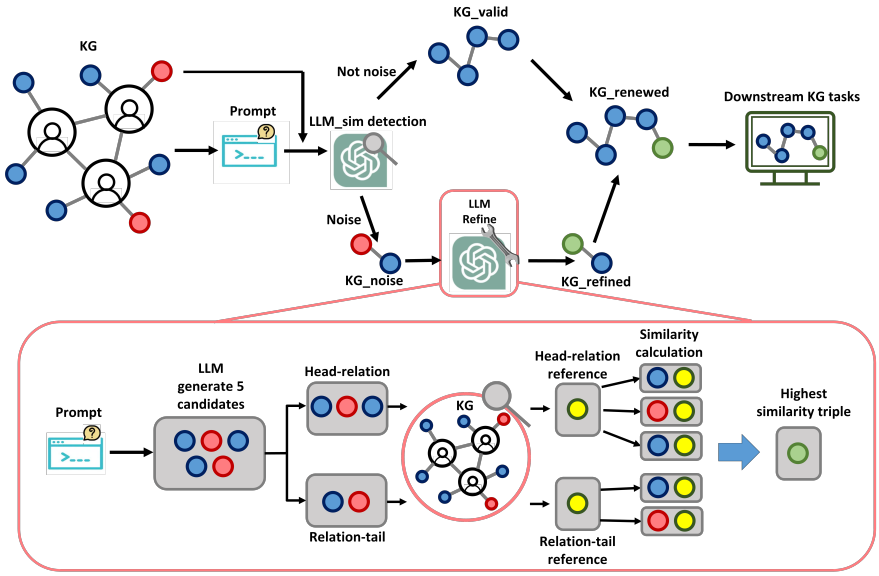

Figure 3.2: Overview of *LLM_sim*.

level text. Furthermore, they lack grounding in the specific schema and scope of the target KG, which can lead to judgments that are factually fluent but structurally inconsistent with the graph's ontology. To address these limitations, we adopt an *ADDITIONAL_CONTEXT (AC)* mechanism. The motivation is twofold: (1) Grounding: Retrieved triples from the KG serve as anchors, forcing the LLM to align its judgment with the existing graph structure rather than relying solely on parametric memory; (2) Disambiguation: Contextual neighbors help distinguish entities with similar surface forms but different semantics. This process transforms the task from simple zero-shot classification into a retrieval-augmented verification task. Thereby helping the model distinguish between genuine noise and valid but implicit relational facts.

The detection process is divided into five steps, as detailed below, and ultimately, the input KG is divided into valid triplets (*KG_valid*) and noisy triplets (*KG_noise*).

Query Vector Construction:

We encode the target triple as a query vector $\mathbf{q}$ that captures its semantic information:

$$\mathbf{q} = f(\text{realization}(h, r, t)), \tag{3.2}$$

where $f(\cdot)$ is a function based on the Sentence Transformer model *all-MiniLM-L6-v2*[1], mapping a sentence to its corresponding embedding. The function realization() converts a triple into a succinct textual statement "h r t", which is then embedded by $f(\cdot)$.

Soft Matching Mechanism.

Given the query vector $\mathbf{q}$, we aim to identify semantically proximate triples in the KG that can serve as contextual evidence. To achieve this, we employ a two-stage retrieval strategy.

We first restrict the search space by generating a candidate pool of triples that are structurally or lexically similar to the target triple. Specifically, we consider triples that share either the head–relation (E_1, R) or relation–tail (R, E_2) pattern with the query, combined with lexical retrieval and lightweight schema constraints.

Each candidate $\tau \in \mathcal{C}$ is converted into a textual realization T_τ and encoded into a dense embedding $\mathbf{t} = f(T_\tau)$, using the same embedding function $f(\cdot)$ defined in Equation (3.2). We then compute semantic similarity between $\mathbf{q}$ and $\mathbf{t}$ using cosine similarity:

$$\cos(\mathbf{q}, \mathbf{t}) = \frac{\mathbf{q} \cdot \mathbf{t}}{\|\mathbf{q}\| \cdot \|\mathbf{t}\|}. \tag{3.3}$$

From a theoretical perspective, this soft matching process leverages the self-attention mechanism inherent in the Transformer-based encoder $f(\cdot)$. The computed cosine similarity acts as a proxy for semantic attention, ensuring that the retrieved context is not just lexically overlapping but semantically attended to by the model's embedding space.

Selection of Similar Triples:

To avoid distracting the LLM with heterogeneous contexts, we select a single triple $\langle h', r', t' \rangle$ that is most similar to $\mathbf{q}$ according to Equation (3.3). The textual realization of this triple is supplied to the LLM as *ADDITIONAL_CONTEXT*. This design empirically stabilizes LLM judgments.

Prompt Design:

The prompt used for noise detection is shown below. We adopt a structured, instruction-following prompt that compels step-by-step reasoning while enforcing a strict output schema. The prompt explicitly references the retrieved

[1] `https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2`

ADDITIONAL_CONTEXT and asks whether the relationship between the target head and tail holds. This prompt reduces ambiguity, improves calibrations of `yes/no` predictions, and simplifies downstream parsing.

Prompt for Noise Detection in *LLM_sim*

Based on all your knowledge and the given context {ADDITIONAL_-CONTEXT}. Determine if the {head} has a {relation} with the {tail}. Answer the question by reasoning step-by-step, and provide your final answer within 'yes' or 'no'.

Answer in this format:
Final Answer: [yes/no]

Filtering of Erroneous Triples:

The LLM's final decision is parsed from the templated output. Triples predicted as `yes` are retained as valid and assigned to KG_valid, whereas those predicted as `no` are flagged as erroneous and assigned to KG_noise.

By systematically executing these five steps, query construction, soft matching mechanism, precise context selection, structured prompting, and filtering, *LLM_sim* produces a principled, context-grounded partition of the input KG into KG_valid and KG_noise, which we employ in the next stage for targeted refinement.

Confidence Measurement via Token Probability:

It is worth noting that the LLM's decision-making process is inherently probabilistic. Although our framework primarily utilizes the discrete output ("Yes" or "No") for hard filtering, the probability distribution over the generated tokens provides a theoretical basis for confidence. Specifically, the likelihood $P(token|context)$ associated with the generated answer implicitly reflects the model's certainty in its detection. This probabilistic nature ensures that the generated classification corresponds to the hypothesis with the maximum likelihood given the context. The use of the token probability for noise detection remains future work.

3.2.2 Noise Refinement

The second stage of our framework is dedicated to refining the noisy triples that have been identified in the previous step. As illustrated in the bottom

half of Figure 3.2, this stage leverages the reasoning capability of LLMs to improve the reliability of the KG. Starting from the initially detected noisy triples, we note that, even when enriched with contextual information, directly correcting erroneous triples remains a non-trivial challenge for LLMs due to their susceptibility to hallucinations and limited grounding in structured knowledge. To address this issue and to strike a balance between computational efficiency and refinement accuracy, we adopt a candidate generation strategy in which the LLM produces five alternative triples for each noisy instance.

These generated candidates are then systematically organized according to two grouping strategies: *Head–Relation* grouping, which considers the semantic consistency between the head entity and relation, and *Relation–Tail* grouping, which emphasizes the alignment between the relation and tail entity. Once the candidate triples are grouped, we further integrate the structural knowledge already embedded in the KG by computing similarity scores between each candidate triple and the reference triples within the graph. This similarity-based evaluation not only anchors the LLM-generated candidates to existing graph semantics but also provides a robust criterion for selecting the most plausible correction.

Finally, among the candidate set, the triple with the highest similarity score is selected as the refined replacement for the corresponding noisy triple. This refinement process ensures that the corrected triples are not only linguistically coherent but also semantically consistent with the broader graph structure, thereby substantially improving the overall quality of the KG. Compared to the naive approach of directly prompting the LLM for corrections, our method demonstrates superior accuracy and reliability.

Candidate Generation

To address the noisy triples, we designed a prompt that allows the LLM to automatically refine the mismatches. We designed the prompt as follows:

Prompt for Noise Refinement in *LLM_sim*

The entities in the given triple do not correctly correspond to each other. Based on your knowledge, please rectify the triple and generate five correct triples, ensuring that the original {relation} remains unchanged. Each refined triple should include either {head} or {tail} from the given triple.
Please output the refined triples in the following format:
1. (entity, relation, entity)
2. (entity, relation, entity)
3. (entity, relation, entity)
4. (entity, relation, entity)
5. (entity, relation, entity)

This prompt explicitly informs the model that the given triples are incorrect and requires it to generate refined versions. Specifically, the LLM is instructed to preserve the original relation while generating five alternative triples, each of which must include either the head or the tail entity from the original input. By constraining the generation process in this manner, we substantially reduce the likelihood of the LLM producing fabricated or semantically irrelevant content.

The rationale behind this design is twofold. First, by explicitly indicating that an input triple is incorrect, the model is guided to make corrections rather than simply reproducing noisy content. Second, by modifying only one entity while preserving explicit constraints on relations, the generation process is grounded in the existing KG structure. This foundation not only enforces factual consistency but also improves semantic coherence with surrounding triples in the graph. This allows the refinement process to leverage the strengths of LLM while mitigating its well-known tendency toward information hallucination.

Formally, each triple can be represented as (E_1, r, E_2), where E_1 and E_2 denote the head and tail entities, respectively, and r denotes the relation. When a triple is corrupted, it is often ambiguous whether the inconsistency stems from the head entity E_1 or the tail entity E_2. To address this uncertainty, our prompt instructs the LLM to fix one entity together with the relation, and to predict the other entity accordingly. This design systematically narrows the error space, thereby reducing noise propagation and improving refinement accuracy. Moreover, by focusing the correction process on a single entity at a time, the approach improves computational efficiency while maintaining semantic fidelity to the KG.

Grouping Strategy

To address the inherent uncertainty regarding whether the noise within a triple originates from the head or the tail entity, we introduce a grouping strategy designed to improve the reliability of LLM-based refinements. The central idea of this strategy is to disaggregate the triple into smaller, semantically coherent components, thereby allowing a more precise evaluation of candidate corrections. Specifically, the generated candidate triples are organized according to two complementary grouping criteria: (i) the *Head–Relation* pair (E_1, r), which groups together all triples that share the same head entity and relation, and (ii) the *Relation–Tail* pair (r, E_2), which groups triples based on their relation and tail entity. This organization facilitates a more targeted assessment of entity–relation alignments and effectively mitigates the noise introduced by erroneous or permuted entities.

Once grouped, candidate triples are compared against reference structures drawn from the original KG. Importantly, rather than attempting to match the entire triple (E_1, r, E_2), which may lead to spurious alignments with unrelated entries, we instead search for partial matches at the level of either (E_1, r) or (r, E_2). By doing so, the refinement process leverages the partial consistency of the noisy triple with the KG, enabling the model to anchor its predictions on verified entity–relation combinations. This approach ensures that semantically similar triples are compared more effectively, even in cases where one entity is corrupted.

The grouping strategy thus serves as a crucial mechanism for improving both the accuracy and robustness of noise refinement. Without such grouping, direct matching of entire triples often leads to incorrect alignments, as the model may erroneously associate mismatched entities with plausible but unrelated KG entries. By contrast, grouping enables a more fine-grained and role-sensitive comparison, focusing independently on the contributions of head and tail entities. This not only reduces the propagation of noise but also enhances the overall fidelity of the refinement process, thereby ensuring that the resulting triples remain semantically consistent with the structure of the underlying KG.

Similarity Calculation

Both the original noisy triple and its generated candidates are embedded into a dense vector space by converting them into textual representations. For any given triple (E_1, r, E_2), we construct its text representation by concatenating the head entity, relation, and tail entity into a single sequence:

$$T = \text{“}E_1 \ r \ E_2\text{”}. \tag{3.4}$$

To represent these triples in a vector space, we employ the Sentence Transformer model, all-MiniLM-L6-v2 variant. The model encodes each triple's textual representation into a dense vector:

$$v_T = f(T), \tag{3.5}$$

where $f(\cdot)$ denotes the SentenceTransformer function defined in Equation (3.2). Thus, for the sequence T of the given triple, the corresponding embedding vector $\mathbf{v}_T$ is obtained by encoding its textual representation.

After generating embeddings for candidate and reference triples, we measure their similarity using cosine similarity. For a pair of triples (T_1, T_2), the cosine similarity between their embeddings $\mathbf{v}_{T_1}$ and $\mathbf{v}_{T_2}$ is calculated as:

$$\cos\_\text{sim}(\mathbf{v}_{T_1}, \mathbf{v}_{T_2}) = \frac{\mathbf{v}_{T_1} \cdot \mathbf{v}_{T_2}}{\|\mathbf{v}_{T_1}\| \|\mathbf{v}_{T_2}\|}. \tag{3.6}$$

For each group, we select the candidate triple with the highest cosine similarity to any reference triple, a triple in KG. A higher cosine similarity indicates that the textual representations of the two triples are more semantically consistent, suggesting that the entity relationships they express are more closely aligned. Let G_j represent a candidate triple and F_i represent a reference triple from the dataset. We aim to find the candidate triple G_j^* that maximizes the cosine similarity to any reference triple within the group:

$$G_j^* = \arg \max_{G_j \in \text{gen}} \left(\max_{F_i \in \text{ref}} \cos\_\text{sim}(\mathbf{v}_j, \mathbf{v}_i) \right), \tag{3.7}$$

- “gen” represents the set for all generated candidate triples.
- “ref” represents the set for all reference triples in the dataset.

Despite the presence of noise in the dataset, the majority of triples remain accurate. Leveraging this fact, our method addresses noise effectively by calculating cosine similarity between generated triples and reference triples. This similarity-based approach allows us to isolate noise from correct data with high accuracy, thereby enhancing the dataset's reliability.

This three-step refinement process, including candidate generation, grouping, and similarity-based selection, ensures that noisy triples are corrected with high precision. Despite the presence of noise, most KG triples remain correct; thus, comparing candidates against the existing KG structure provides a reliable criterion for refinement. The final refined set ($KG\_renewed$), combined with valid triples (KG_valid), constitutes the renewed KG.

3.2.3 Illustrative Example of Detection and Refinement

To demonstrate the proposed workflow, we consider the noisy triple:

(Albert Einstein, *works_at*, NASA).

Noise Detection

The triple is realized as the sentence $s =$ *"Albert Einstein works at NASA"* and encoded into a query vector $\mathbf{q} = f(s)$. Using Equation (3.3), we retrieve the top-K nearest neighbors from the KG as shown in Table 3.1.

Table 3.1: Illustrative example of top-K retrieved triples (from the KG) for the query "Albert Einstein works at NASA". The highest-scoring triple $\tau^\star (= \tau 1)$ is used as *Additional Context*.

Candidate Triple τ_i & Its Realization T_i	$\cos(\mathbf{q}, \mathbf{v}_\tau)$
τ_1: (Albert Einstein, *works_at*, Institute for Advanced Study) T_1: *Albert Einstein works at Institute for Advanced Study*	**0.86**
τ_2: (Sean Duffy, *works_at*, NASA) T_2: *Sean Duffy works at NASA*	0.83
τ_3: (Albert Einstein, *works_at*, ETH Zurich) T_3: *Albert Einstein works at ETH Zurich*	0.72

Here, τ_1 achieves the highest similarity and is selected as $\tau^\star$, which provides the contextual evidence for the following prompt for noise detection that shows the original triple is likely noisy.

Prompt Example for Noise Detection in *LLM_sim*

Based on all your knowledge and the given context {(Albert Einstein, works_at, Institute for Advanced Study)}. Determine if the {Albert Einstein} has a {works_at} with the {NASA}. Answer the question by reasoning step-by-step, and provide your final answer within 'yes' or 'no'.
Answer in this format: Final Answer: [yes/no]

Having evaluated the target triple against the retrieved contextual evidence, the LLM concludes that the original assertion(Albert Einstein, *works_at*, NASA) is not true. Accordingly, this triple is classified as noisy. In the

subsequent step, we proceed to the *Refinement* stage, where the goal is to generate and evaluate candidate replacements in order to restore semantic correctness and align the triple with factual knowledge.

Noise Refinement

Once the triple is flagged as noisy, the LLM is prompted to generate alternative triples while preserving the relation *works_at*.

To apply the grouping strategy introduced in subsection 3.2.2, the five candidate triples are divided into two groups based on which entity remains unchanged from the original triple. Specifically, one group preserves the head entity "Albert Einstein", while the other preserves the tail entity "NASA". The model produces five candidates:

Group 1: Head entity preserved (Albert Einstein)

- (Albert Einstein, works_at, Princeton University)
- (Albert Einstein, works_at, FBI)

Group 2: Tail entity preserved (NASA)

- (Ruth Amundsen, works_at, NASA)
- (Stephen Hawking, works_at, NASA)
- (Carl Sagan, works_at, NASA)

We then apply the ranking strategy using Equation (3.6) to compare these candidates with KG references. The highest-scoring candidate is selected as the refined replacement:

$$(\text{Albert Einstein}, \textit{works\_at}, \text{Princeton University}).$$

This example demonstrates how the framework first identifies a noisy triple through context-based detection, and then corrects it via LLM-generated candidates combined with similarity-based refinement.

3.3 Rule-Based Approach (LLM_rule)

Figure 3.3 provides an overview of the proposed *LLM_rule* framework. The central idea of this approach is to integrate rule-based semantic constraints

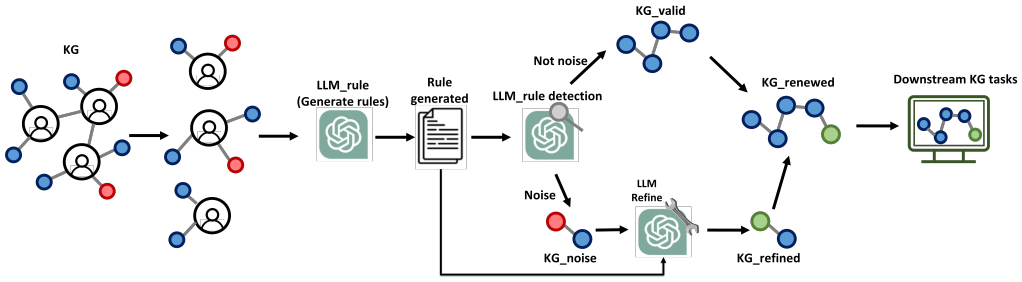

Figure 3.3: Overview of *LLM_rule*.

into the process of noisy triple detection and refinement, thereby complementing the reasoning capacity of LLMs with explicit and structured knowledge. In particular, rules that capture semantic dependencies between entities and relations are automatically generated by an LLM, and these rules are subsequently leveraged as auxiliary guidance in both the noise detection and refinement stages. While LLMs are capable of producing semantically rich representations, they often lack explicit grounding in domain-specific constraints. As a result, their predictions may suffer from factual inconsistencies or logical invalidity, especially when handling complex KG structures. By incorporating rules as an additional layer of supervision, the model gains access to symbolic constraints that explicitly encode permissible or impermissible entity–relation configurations. This not only improves the precision of noise detection by highlighting triples that violate known constraints, but also enhances the refinement process by guiding the generation of corrected triples toward semantically consistent outcomes. *LLM_rule* extends the standard LLM-based framework by embedding rule-based constraints into the prompt design. Through this integration, the LLM is able to reason not only over contextual semantics but also over explicit structural knowledge of the *KG*, leading to more reliable and accurate identification and correction of noisy triples.

3.3.1 Rule Generation

To incorporate explicit semantic knowledge into the noise detection and refinement pipeline, we employ an LLM to automatically generate rules that capture relational constraints. For each relation r in the KG, we first extract all triples associated with r from the original *KG*. The set of these triples is denoted as $T_r = \{(h_i, r_i, t_i) \mid r_i = r\}$. To construct representative examples for rule induction, we randomly sample ten triples from this set, ensuring that $|T_r| = 10$. The head and tail entities extracted from T_r serve as input

exemplars to guide the LLM in producing rules that generalize the underlying semantics of relation r.

For each relation, we instruct the LLM to generate five rules, denoted as $\{\rho_1, \rho_2, \ldots, \rho_5\}$. These rules are designed to be complementary and non-redundant, capturing distinct relational properties from different perspectives, such as type constraints, cardinality restrictions, temporal validity, external knowledge base alignment, and semantic plausibility checks. The prompt used to guide the LLM in generating such rules is presented below.

Prompt for Rule Generation

Given the relation "{relation}" with the following entity pair examples:
Head entities: {", ".join(head_examples)}
Tail entities: {", ".join(tail_examples)}
You are an expert in Knowledge Graph validation. Your task is to generate rules for detecting incorrect triples involving a given relation. Based on the relationship of these examples, generate a rule that can be used to determine whether a triple with this relation is valid.
According to commonsense knowledge in realistic scenarios, please generate 5 logical rules in natural language to describe the premises of the given conclusion. Each rule should:
1. Describe a unique pattern from different aspects (Entity Type Constraint, One-to-Many Relationship Constraint, Temporal Consistency Check, Trusted Knowledge Base Validation, Semantic Contradiction Detection)
2. Use placeholders X/Y for entities
3. Specify clear validation conditions
4. Reflect patterns observed in the examples
5. Avoid overlapping between rules
Output the rules in Natural language ONLY the numbered rules without explanations.

Here, the placeholders "head_examples" and "tail_examples" correspond to the head and tail entities extracted from T_r, respectively. By incorporating these examples into the prompt, the LLM is guided to derive rules that are not only contextually relevant to the specific relation but also generalizable to unseen triples involving the same relation.

We fixed the number of generated rules at $k = 5$ to balance semantic coverage and the cognitive limitations of the LLM. Empirically, a smaller set of rules often fails to capture the multi-faceted nature of complex relations. Conversely, increasing the number of rules (e.g., to 10 or more) exacerbates

the context window limitation of the LLM during the subsequent detection phase. Feeding an excessive number of rules into the prompt creates information overload, which dilutes the model's attention and leads to hallucinations or redundant constraints. Thus, five rules represent an optimal trade-off between strict constraint coverage and reasoning efficiency.

Table 3.2 illustrates examples of rules generated for the relation */location/country/form_of_government*. It is found that the induced rules encode diverse semantic constraints. For instance, ρ_1 enforces entity type compatibility by requiring the head entity to be a country; ρ_2 introduces one-to-many consistency by prohibiting conflicting government forms for the same entity; ρ_3 incorporates temporal validity by aligning the government form with historical eras; ρ_4 leverages trusted external knowledge bases for validation; and ρ_5 ensures semantic plausibility by rejecting logically inconsistent assignments. Together, these rules provide a structured mechanism to capture the essential relational properties of r, thereby enhancing the LLM's ability to detect and refine noisy triples with higher precision.

Table 3.2: Examples of rules

Relation: `/location/country/form_of_government`
Rule ρ_1: Entity Type Constraint X must be a country to have a form of government. – If the head is not a country (e.g., a city or region), the triple is likely invalid.
Rule ρ_2: One-to-Many Constraint A country should not simultaneously have multiple conflicting forms of government. – If X has both monarchy and democracy labels, the triple is questionable.
Rule ρ_3: Temporal Consistency The form of government must align with historical periods. – A mismatch between Y and the political regime during the relevant era indicates noise.
Rule ρ_4: External KB Validation The assigned government form must match trusted sources like Wikidata. – Contradictions with reliable KBs suggest incorrect information.
Rule ρ_5: Semantic Contradiction Detection The form of government must be plausible given the country's political background. – Labeling the UK as a Communist state is semantically implausible.

3.3.2 Rule Quality Validation

While the initial rule generation process aims to capture diverse semantic constraints for each relation, the rules produced by the LLM may occasionally contain inconsistencies, redundancies, or ill-formed conditions. To ensure that only logically sound and semantically reliable rules are retained, we introduce a dedicated validation stage, termed *rule validation*. This stage functions as a Self-Correction mechanism, where the model is tasked with reviewing and critiquing its own generated output to filter out logical errors or hallucinations. In this stage, the LLM is prompted to critically examine each candidate rule and provide an explicit validity judgment. The assessment follows a binary classification scheme, where each rule is labeled as either "valid" or "invalid," accompanied by a brief explanation that outlines the rationale behind the judgment. This design guarantees not only a systematic filtering of low-quality rules but also an interpretable validation process that allows for transparent quality control.

The following prompt is designed so that the LLM provides binary validity judgments along with concise explanations.

Prompt for Rule Validation

Please evaluate the following relation and its five associated rules.
The relation is: {relation}. The rules are: {rules_text}.
For each rule, verify the following aspects:
1. Whether the rule is logically consistent (free from internal contradiction).
2. Whether the rule is semantically aligned with the intended meaning of the relation {relation}.
3. Whether the rule is non-redundant, i.e., it does not duplicate the content of another rule in the set.
If a rule is valid, output 'Valid'. If a rule is invalid, output 'Invalid' along with specific repair suggestions.
Your final answer must follow the original order and contain exactly the following lines (do not add extra commentary):
Relation: {relation} Rule 1: [Valid/Invalid] - Explanation (include repair suggestions if invalid)
Rule 2: [Valid/Invalid] - Explanation (include repair suggestions if invalid)
Rule 3: [Valid/Invalid] - Explanation (include repair suggestions if invalid)
Rule 4: [Valid/Invalid] - Explanation (include repair suggestions if invalid)
Rule 5: [Valid/Invalid] - Explanation (include repair suggestions if invalid)

Formally, for each relation r, let $\hat{\mathcal{R}}_r$ denote the set of rules initially generated by the LLM. Each rule $\rho \in \hat{\mathcal{R}}_r$ is independently evaluated using the validation prompt, which instructs the LLM to verify whether ρ is (i) logically consistent, (ii) semantically aligned with the relation r, and (iii) non-redundant with respect to other rules. Rules that fail to satisfy these criteria are classified as invalid and are consequently discarded. The subset of rules that pass validation constitutes the final rule set $\mathcal{R}_r$, which is subsequently used in noise detection and refinement tasks. This step ensures that $\mathcal{R}_r$ encodes constraints that are both contextually relevant and robust across different entity instantiations.

To further enhance the accuracy of the classification process, the validation prompt also instructs the LLM to repair rules deemed invalid by suggesting corrected versions. Although the repaired rules are not incorporated into $\mathcal{R}_r$, this repair-oriented instruction has an important auxiliary role: it encourages the LLM to engage in deeper semantic reasoning and

to reconsider the underlying rationale for its validity judgments. In our experiments, we found that the inclusion of repair instructions substantially improved the precision of rule classification, as the model became less prone to false positives when evaluating borderline cases. Thus, even though the repaired outputs are not directly utilized, they play a crucial role in shaping the overall reliability of the validation mechanism.

3.3.3 Noise Detection

Once a high-quality rule set $\mathcal{R}_r$ has been validated and finalized for a relation r, it can be leveraged to assess the reliability of individual triples in the original KG. The fundamental idea is to determine whether a given triple (h, r, t) adheres to the semantic constraints encoded by the rules in $\mathcal{R}_r$. Specifically, for each triple, the LLM evaluates its consistency with respect to the rules. If the triple satisfies the majority of the rules in $\mathcal{R}_r$, it is classified as *valid*; otherwise, it is flagged as *invalid* and considered as noise. This majority-based criterion ensures robustness against occasional ambiguities or overly restrictive rules, while still maintaining high precision in noise detection.

Formally, let $\mathcal{R}_r = \{\rho_1, \rho_2, \ldots, \rho_m\}$ denote the set of validated rules for relation r. For a candidate triple (h, r, t), we define a binary scoring function $\sigma(\rho_j, (h, r, t)) \in \{0, 1\}$, which evaluates whether the triple satisfies rule ρ_j. The overall validity score of the triple is then given by the aggregation:

$$S(h, r, t) = \sum_{j=1}^{m} \sigma(\rho_j, (h, r, t)). \tag{3.8}$$

A triple (h, r, t) is classified as valid if $S(h, r, t) > \frac{m}{2}$, i.e., if it is consistent with the majority of rules. Otherwise, the triple is labeled invalid. This formulation allows the system to tolerate minor inconsistencies while still identifying triples that substantially deviate from the expected semantic patterns.

In addition we further investigate three distinct aggregation strategies to determine the validity of a triple given the rule set $\mathcal{R}_r$. The first strategy, *LLM_rule_one*, adopts a permissive approach in which a triple is considered valid if it satisfies at least one rule. The second strategy, *LLM_rule_maj*, corresponds to the majority-based criterion described above, requiring that more than half of the rules be satisfied. The third strategy, *LLM_rule_all*, enforces the strictest condition, necessitating that the triple must satisfy all applicable rules to be accepted. These variants enable a systematic analysis of how different aggregation policies influence both detection accuracy and refinement quality.

To operationalize this process, the LLM is employed as a reasoning engine that applies the rules in natural language to the triple under consideration. The prompt is designed as follows:

Prompt for Noise Detection in *LLM_rule*

You are an expert in knowledge graphs and logical reasoning. Your task is to determine whether a given triple follows the specified rules.
Rule:{rule}
Triple:
Head Entity: "{head_text}"
Relation: "{relation}"
Tail Entity: "{tail_text}"
Task: 1. Analyze whether the head and tail entities satisfy the given rule.
2. If more than half of the rules apply to this triple, return "Valid".
3. If more than half of the rules do not apply, return "Invalid".
4. Justify your reasoning step by step for each rule.
5. Please also consider the context step by step of whether the relation and entities are true or false in the real world.
Output Format:
Valid or Invalid: {validity}
Justification: {reasoning}

Concretely, the rules $\mathcal{R}_r$ are explicitly provided in the LLM prompt, together with the candidate triple, to serve as structured background knowledge. By incorporating these rules, the LLM is guided to perform a context-aware evaluation rather than relying solely on statistical plausibility derived from its pre-trained parameters. This integration of explicit constraints enables the LLM to achieve higher accuracy in noise detection, especially in cases where entity semantics are subtle or context-dependent.

The corresponding prompt is designed to be both precise and interpretable: it enumerates the relevant rules and asks the LLM to determine whether the candidate triple conforms to them. The inclusion of rules as explicit prior knowledge effectively enhances the transparency of the reasoning process, as the decision can be traced back to concrete semantic conditions. In this way, noise detection benefits from the complementary strengths of symbolic rules and the LLM's natural language inference capabilities, resulting in a more reliable and interpretable framework for assessing triple validity.

3.3.4 Noise Refinement

In addition to detecting noisy triples, the rule set $\mathcal{R}_r$ is further employed to guide the refinement of invalid triples, thereby improving the overall quality of the KGs. The central idea is that rules encode essential semantic constraints governing the relation r, and thus can serve as corrective signals when a triple (h, r, t) is flagged as inconsistent. Instead of discarding such triples entirely, our approach leverages the LLM to generate semantically valid replacements that are consistent with the given relation and the rules. This strategy not only mitigates information loss but also enables the recovery of structurally valid triples, thus enriching the renewed KG.

Concretely, when a triple (h, r, t) is identified as potentially incorrect, the LLM is prompted to refine it under the guidance of a specific rule $\rho \in \mathcal{R}_r$ that the triple violates. The refinement task is deliberately constrained: the relation r is fixed, and either the head or the tail entity must remain unchanged, while the other entity is replaced to ensure consistency with the rule. This controlled modification design serves two purposes: (1) it significantly reduces the risk of hallucination by the LLM, since the model is not required to generate arbitrary new triples; and (2) it preserves the structural alignment of the KG by maintaining partial continuity with the original triple.

The refinement process is implemented through the following prompt, which provides the LLM with the candidate triple, the violated rule, and explicit step-by-step instructions for correction:

Prompt for Noise Refinement in *LLM_rule*

You are an expert in knowledge graphs and logical reasoning. The following triple is potentially incorrect based on the logic rule of its relation.
Rule:{rule}
Possibly Incorrect Triple:
Head Entity: "{head_text}"
Relation: "{relation}"
Tail Entity: "{tail_text}"
Task:
1. Analyze why the triple might be incorrect based on the rule.
2. Without changing the relation and one of the entities, generate 5 candidate that would make the triple valid according to the rule.
3. Provide only the corrected triples in the following format strictly:
(correct triples: - {head entity} {relation} {tail entity})

This prompt is designed with several considerations. First, by explicitly presenting the violated rule alongside the triple, the LLM is provided with structured background knowledge that grounds its reasoning process. Second, the instruction to retain either the head or the tail entity while keeping the relation fixed enforces strong semantic and structural constraints, thereby avoiding arbitrary modifications that could compromise KG consistency. Third, the requirement to generate multiple (five) alternative candidates increases the likelihood that at least one corrected triple is both rule-consistent and factually accurate, offering a broader refinement pool for subsequent filtering. Finally, the request for outputs in a strict and uniform format ensures compatibility with downstream processing pipelines, where candidate triples are compared against reference KGs for validation.

We adopt a simple yet effective strategy by directly selecting the first candidate generated by the LLM. Generating multiple candidates allows the model to explore diverse refinement possibilities, thereby increasing the overall chance of including the correct triple among them. However, since modern LLMs internally rank their outputs by likelihood, the first candidate typically represents the model's highest-confidence prediction, the one it deems most semantically coherent and factually plausible. This is particularly suitable for KG refinement, where each triple usually has a single correct completion. Hence, the top-ranked candidate often aligns with the ground truth. Empirically, we observe that in most cases, the first candidate already corresponds to the correct refinement. This strategy leverages the model's inherent confidence ordering without requiring additional scoring or filtering. Although more elaborate selection mechanisms can be applied on top of this baseline, we find that even this simple heuristic consistently yields competitive performance in practice.

In this way, the rule-guided refinement process integrates symbolic constraints with generative reasoning, enabling the recovery of high-quality triples from noisy data. Together with rule-based detection, this refinement stage ensures that the renewed KG not only minimizes noise but also retains and restores semantically valid knowledge.

It is important to note that this refinement strategy differs fundamentally from the methodology employed by *LLM_sim*. The two frameworks are built upon mutually exclusive validation anchors.

LLM_sim is a context-driven model that validates each candidate's semantic coherence against the existing KG_{valid} context. This validation is essential because its "weak-constraint" prompt often generates multiple factually plausible candidates, and the ranker must select the one most consistent with the KG's scope.

In contrast, *LLM_rule* is a logic-driven model anchored not in the KG

context but in the logical rule itself. This "strong-constraint" prompt sharply narrows the hypothesis space, allowing the Top-1 generation to represent a high-confidence, rule-entailing solution. Additional contextual ranking would contradict its design assumption by reintroducing semantic biases that logical reasoning explicitly avoids.

This design difference results in an inherent incompatibility: applying the *LLM_sim* ranker to rule-generated triples would systematically down-rank logically correct yet unseen knowledge, as it lacks existing semantic anchors. Empirical evidence in Chapter 4, high accuracy but low Restoration Ratio (RR), supports this behavior.

Therefore, *LLM_rule* adopts a direct Top-1 refinement strategy, preserving its ability to perform pure logical inference and discover novel, rule-consistent knowledge. Together, the two frameworks form complementary paradigms—semantic alignment versus logical discovery—within our broader exploration of LLM-based refinement.

Chapter 4

Evaluation

4.1 Datasets

To evaluate the effectiveness of the proposed methods, we conduct experiments on three widely adopted benchmark datasets for KGC: **WN18RR**, **FB15k-237**, and **NELL**. These datasets are representative of different domains and characteristics, allowing for a comprehensive assessment of both KGC performance and the robustness of noise detection and refinement strategies.

Table 4.1 summarizes the statistics of the datasets.

Table 4.1: Statistics of Datasets

Dataset	WN18RR	FB15k-237	s-NELL
Entity	40,943	14,541	745
Relation	11	237	301
Training Triples	86,835	272,115	500
Validation Triples	3,034	17,535	500
Test Triples	3,134	20,466	500

4.1.1 Standard Benchmarks and Synthetic Noise Generation

We utilize two existing public datasets:

- **WN18RR**[1] is a subset of WordNet [44], a large lexical database of English where words are grouped into sets of synonyms (synsets) and

[1] `https://huggingface.co/datasets/VLyb/WN18RR`

linked by various semantic relations. WN18RR focuses on a reduced set of 11 relations and 40,943 entities, and is designed to address test leakage issues identified in its predecessor WN18 by removing reversible relations.

FB15k-237[2] is derived from Freebase [10], a large-scale collaborative knowledge base. This dataset contains 14,541 entities and 237 relations. It is constructed by eliminating inverse relation pairs to reduce redundancy and improve the rigor of KGC evaluation.

Creation of Pseudo Noisy Triplets:

Since these benchmarks originally contain only correct facts, we established the ground truth for noise detection through a controlled injection protocol. For every positive triple (h, r, t) in the training set, we generated a corresponding noisy triple (h', r, t) or (h, r, t') by randomly replacing the head or tail entity with a different entity e_{random} sampled from the entity set E.

The precise definitions of the valid and noisy triples are as follows:

1. The original triples from the benchmark are labeled as **Valid (Positive)**.

2. The synthetically corrupted triples are labeled as **Noisy (Negative)**.

We generated datasets with noise ratios of 10%, 20%, and 30%. This synthetic approach allows us to have a clear, indisputable ground truth for calculating precision and recall.

We focused on entity corruption for two primary reasons. First, empirical surveys on KG quality indicate that entity-related errors, such as incorrect entity linking and disambiguation failures, constitute the predominant source of noise in automatically constructed KGs [50]. Second, in the context of KGC, the standard link prediction task typically frames the problem as inferring a missing entity given a relation (i.e., $(h, r, ?)$). Therefore, modeling noise through entity corruption serves as a representative and rigorous simulation of real-world error patterns impacting downstream applications.

4.1.2 Real Noisy Dataset: s-NELL

We further assess our methods on a real-world noisy KG, namely the **Never-Ending Language Learning (NELL)** dataset[3] [30]. NELL is an automat-

[2] `https://huggingface.co/datasets/VLyb/FB15k-237`

[3] `http://rtw.ml.cmu.edu/rtw/`

ically constructed knowledge base built through continuous web-scale information extraction. While it contains a large volume of factual knowledge, the automatic extraction process inevitably introduces a significant amount of noise, making it a realistic testbed for evaluating noise detection and refinement methods.

Construction Protocol:

For the purpose of this study, we curate a small-scale version manually, referred to as **s-NELL**. Unlike the synthetic datasets, the ground truth here relies on human verification. The construction process is as follows:

1. **Sampling:** We randomly sampled 500 triples from the original NELL database for training.
2. **Manual Annotation (Ground Truth):** We manually inspected the training partition against trusted external sources.
3. **Labeling:** Triples confirmed as factually correct were labeled **Valid** ($N = 398$), and those containing extraction errors were labeled **Noisy** ($N = 102$).

This process yielded a dataset with a natural noise ratio of 20.4%, providing a realistic testbed where the "Noisy" ground truth comes from actual algorithm failures rather than artificial injection.

Our experimental setup leverages both synthetic and naturally noisy datasets. WN18RR and FB15k-237 provide well-established, clean benchmarks with controlled synthetic noise injection, while s-NELL offers a naturally noisy environment closer to real-world scenarios. This combination allows for a comprehensive evaluation of the proposed framework across varying degrees and types of noise, thereby ensuring both robustness and generalizability of the results. It is worth noting that while WN18RR and FB15k-237 are general-domain benchmarks, s-NELL introduces real-world extraction noise and long-tail entities, serving as a challenging proxy for the complexity often found in domain-specific applications.

4.2 Evaluation of Noise Detection

4.2.1 Experimental Settings of Noise Detection

To comprehensively evaluate the effectiveness of our proposed methods, namely *LLM_sim* and *LLM_rule*, we compare them against a broad set of representative baselines and ablated variants. These comparison methods are carefully

selected to cover diverse paradigms of noise detection, including embedding-based approaches, rule-based methods, state-of-the-art contrastive frameworks, and pre-trained language models. The details of these baselines are explained as follows.

(1) Embedding-based KGE Models. We employ three classical and widely recognized KGE methods as baselines: **TransE**, **RotatE**, and **ExpressivE**. These models have been extensively studied in the literature and are representative of different families of embedding techniques. Their primary objective is to learn continuous vector representations of entities and relations in a KG, such that the structural properties of triples are preserved in the embedding space. Once KGE is trained, these embeddings can be leveraged to assess the plausibility of any given triples, thereby enabling the detection of noise.

In particular, for a triple (h, r, t), where h and t denote the head and tail entities and r represents the relation, a scoring function is used to quantify its plausibility. It is important to note that KGE models are inherently designed for ranking rather than classification. To adapt them for noise detection, we employ a **threshold-based strategy**. For each triple, the model computes an energy score $E(h, r, t)$ (e.g., $||h + r - t||$ for TransE), where lower values indicate higher plausibility. We determine an optimal decision threshold γ by maximizing the classification accuracy on the validation set. Triples with energy scores exceeding γ are flagged as noisy, while those below are considered valid. Based on this optimization, the thresholds were set to $\gamma = 0.1$ for TransE and RotatE, and $\gamma = 0.2$ for ExpressivE.

(2) Handcrafted Rule-based Method (H-Rule). We further design a handcrafted rule-based baseline, denoted as **H-Rule**, which leverages prior linguistic knowledge and entity type constraints to detect noise in KGs. This baseline provides a transparent comparison against our proposed LLM-based approaches, although it is inherently limited by its reliance on manually crafted rules.

For the WN18RR dataset, the relations are derived from WordNet and inherently associated with lexical semantic properties. Since WordNet explicitly provides part-of-speech (POS) annotations for entities, we directly leverage this information to define admissible POS constraints for each relation. A triplet is classified as noisy if the POS of either its head or tail entity does not satisfy the predefined constraint of the relation. For instance, relations such as synonymy or hypernymy are expected to connect entities within specific lexical categories (e.g., noun–noun or verb–verb), and any deviation

from these patterns is considered an indicator of noise.

In the case of FB15k-237, the major challenge arises from the considerably larger number of relations (237 in total), which renders the manual specification of constraints for each relation infeasible. To address this, relations are first clustered into semantically coherent groups to reduce complexity. Within each group, we analyze the head and tail entities of every relation using Named Entity Recognition (NER). The recognized entity types are aggregated to determine the dominant or majority type patterns associated with the relation. These majority entity types are then defined as the expected type constraints for the head and tail positions. A triplet is flagged as noisy if either its head or tail entity exhibits a type inconsistent with these majority-defined constraints. This grouping-and-majority strategy allows for the scalable construction of interpretable rules, while still maintaining sufficient granularity to capture relation-specific patterns.

An example of H-Rule for FB15k-237

Relation: /people/person/place_of_birth,
Most Common Head Type: PERSON,
Most Common Tail Type: GPE

Head Type Counts:
PERSON 90
UNKNOWN 4
GPE 2
ORG 2
WORK_OF_ART 1

Tail Type Counts:
GPE 73
PERSON 14
UNKNOWN 8
ORG 2
FAC 1
ORDINAL 1

To illustrate how the prompt operates in practice, consider the relation */people/person/place_of_birth*. As described earlier, the majority type of the head entities for this relation is *PERSON* with the highest 90 instances, while the dominant tail type is *GPE* with the highest 73 instances. Accordingly, the expected type constraint derived for this relation is $PERSON \rightarrow GPE$.

For the s-NELL dataset, which is automatically constructed and widely acknowledged to contain a substantial amount of noise, we adopt an entity type–based strategy similar to that used for FB15k-237. However, unlike FB15k-237, where relations can be grouped into semantically coherent categories, the relations in s-NELL are often less clearly defined and exhibit higher variability. To address this challenge, we determine the plausible types of head and tail entities that a given relation is expected to connect, based on the observed type distributions in the data. If the type of either the head or the tail entity deviates from these expected categories, the corresponding triplet is classified as noisy. This type-driven rule design is particularly suitable for s-NELL, as it leverages the availability of entity type information while accounting for the dataset's inherently uncertain and noisy relation definitions.

To further demonstrate the applicability of this strategy to other knowledge graphs, consider the relation *aquariumincity* from the NELL dataset. After applying NER to the head and tail entities associated with this relation, we observe that the most frequent head type is *ORG*, which appears in 3 instances, whereas the remaining heads are labeled as *UNKNOWN* (2 instances). For the tail position, the dominant type is *PERSON* with 2 instances, followed by *GPE* (2 instances) and *UNKNOWN* (1 instance). On the basis of these distributions, the inferred constraint specifies that the head entity of *aquariumincity* should predominantly be of type *ORG*, while the tail entity is expected to be of type *PERSON*. Any triple whose head or tail deviates from these majority types is thus treated as a potential noise candidate. This example illustrates that the majority-type inference mechanism remains effective even when the distribution of entity types is relatively small or partially uncertain, as is often the case in NELL.

An example of H-Rule for s-NELL

relationship:aquariumincity,
Most Common Head Type: ORG,
Most Common Tail Type: PERSON
Head Type Counts:
ORG 3
UNKNOWN 2

Tail Type Counts:
PERSON 2
GPE 2
UNKNOWN 1

H-Rule baseline embodies a deterministic and interpretable method for noise detection. While it is inherently constrained by the manual effort required for rule construction and the limited coverage of handcrafted constraints, it serves as a valuable benchmark to highlight the advantages of more flexible LLM-based methods.

(3) Automated Rule Mining (AMIE). Distinct from the handcrafted constraints in H-Rule, we include **AMIE** [24], a representative system for automated rule mining in KGs. AMIE discovers logical Horn rules (e.g., $r_1(x, y) \wedge r_2(y, z) \Rightarrow r_3(x, z)$) by calculating statistical metrics such as PCA confidence and support directly from the graph structure. In our experiments, we utilize AMIE to mine high-confidence rules from the noisy KG. A triple is considered valid if it is entailed or supported by these mined rules. Including AMIE serves as a critical benchmark to contrast the robustness of our LLM-based inference against traditional statistical rule mining, particularly in scenarios where explicit structural evidence is sparse or corrupted.

(4) Contrastive Noise Filtering. We further include **CAGED** [87], a state-of-the-art method specifically designed for detecting noisy triples in KGs. Unlike earlier approaches that primarily rely on handcrafted logical rules or negative sampling strategies, CAGED addresses the inherent difficulty of distinguishing realistic yet incorrect triples (e.g., *"Bruce_Lee, place_of_birth, China"*) that often lack explicit supervision signals. To this end, CAGED introduces a novel contrastive learning framework that leverages multiple representational views of triples and jointly optimizes embedding-based and contrastive objectives. The core idea of CAGED is to construct *hyper-views*, where each triple is represented as a node and encoded from different perspectives. By enforcing consistency of triple representations across these views, the model learns to capture more fine-grained relational semantics. In addition, CAGED integrates a dual training objective: the standard embedding loss ensures structural fidelity in the KG, while the contrastive loss explicitly encourages alignment between different views of the same triple. This joint optimization enables the model to evaluate the credibility of triples based on two complementary signals: (i) the cross-view consistency of representations, and (ii) the intrinsic self-compatibility of the triple itself. For noise detection, CAGED evaluates each triple by measuring its representation consistency across views and its internal plausibility within the embedding space. Triples that exhibit low cross-view agreement or weak self-compatibility are assigned lower credibility scores and are thus flagged as noise. This mechanism enables the model to quantify the reliability of a

triple directly.

Its ability to model subtle distinctions in relational structure and to identify inconsistencies at scale makes it not only a strong but also a particularly challenging baseline for evaluating our proposed framework.

(5) Pre-trained Language Model. To further benchmark our approaches against contemporary advances in generative AI, we include methods based on large-scale pre-trained language models. In this setup, a pre-trained LLM is directly prompted to classify whether a given triple is correct or incorrect, without any additional task-specific fine-tuning. We employ **GPT-2 XL** as the representative backbone in this experiment. As a model pre-trained on massive text corpora, GPT-2 XL possesses strong general language understanding capabilities, allowing it to infer semantic plausibility and relational coherence between entities, even in the absence of explicit KG-specific training. This makes it a practical baseline to evaluate the effectiveness of our noise detection methods.

Specifically, GPT-2 XL detects noise by reasoning over triples using natural language prompts. An example prompt used in this study is:

Prompt for GPT2-XL in *LLM_rule*

Is ⟨Entity1⟩ a ⟨relationship⟩ of ⟨Entity2⟩? Answer the question with "yes" or "no".

Note that this prompt differs from the one used for Llama3, as described in subsection 3.2.1. Initially, identical prompts were applied to both GPT-2 XL and Llama3. However, prompts optimized for GPT-2 XL did not elicit appropriate responses from Llama3 due to differences in model architecture and training objectives, and vice versa. By tailoring prompts to the strengths of each model, we are able to leverage the knowledge pre-trained in GPT-2 XL more effectively, thereby improving the accuracy of noise detection in KG triples.

(6) Variants of the Proposed Methods. To gain a more comprehensive understanding of the contributions of individual components in our proposed framework, we conduct an ablation study by evaluating several variants of the methods.

For *LLM_sim*, we evaluate a variant denoted as **LLM_sim wo AC**. Crucially, this variant serves as the Standard Zero-Shot Baseline for our study. Unlike a naive zero-shot approach (which simply asks "Is this triple correct?"

without reasoning), this variant utilizes our optimized CoT prompt but relies solely on the LLM's parametric memory without external retrieval. We prioritize this strong baseline to demonstrate that the performance gains are strictly attributable to the Additional Context rather than just prompt engineering. Evaluating *LLM_sim wo AC* allows us to systematically measure the contribution of contextual information to the model's effectiveness, highlighting the extent to which additional knowledge in the prompt enhances the accuracy and reliability of noise detection and triple refinement in KGs.

For *LLM_rule*, we evaluate the following three strategies.

- ***LLM_rule_one*** adopts a permissive policy, classifying a triple as valid if it satisfies at least one rule.
- ***LLM_rule_maj*** follows the majority-based criterion, where a triple is considered valid if more than half of the rules support it.
- ***LLM_rule_all*** applies the strictest criterion, requiring all rules to be satisfied simultaneously.

In addition, to evaluate the effect of the rule validation step, we examine ablated versions of these methods, indicated by the suffix "wo VAL," in which the validation of generated rules is deliberately omitted. It is noteworthy that, for the WN18RR dataset, this validation step is always excluded. Empirical observations suggest that the rules derived for WN18RR are sufficiently reliable, and further validation does not contribute to improve the performance.

These ablation studies provides detailed insights into the contributions of prompt context, rule aggregation, and rule validation in enhancing the robustness and accuracy of the proposed KG noise detection and refinement framework.

4.2.2 Results of Noise Detection

Tables 4.2 4.3 and 4.4 report the results under different noise ratios 10%, 20%, and 30%, respectively. Table 4.5 show the results of noise detection on s-NELL datasets. It is important to note that AMIE is omitted from the s-NELL results because it failed to extract any valid rules from this dataset. As a statistical rule mining system, AMIE requires a minimum level of pattern recurrence3 to establish rules. However, the s-NELL dataset is extremely sparse, featuring only 500 triples distributed across 301 relations. This lack of data density prevented the algorithm from satisfying the necessary support constraints, resulting in an empty rule set.

Table 4.2: Results of noise detection on WN18RR and FB15k-237 with 10% noise ratio. A, P, R, and F stand for accuracy, precision, recall, and F-measure, respectively.

Model	WN18RR				FB15k-237			
	A	P	R	F	A	P	R	F
TransE	.473	.896	.468	.615	.411	.884	.398	.549
RotatE	.511	.900	.509	.650	.464	.895	.459	.606
ExpressivE	.603	.902	.585	.709	.527	.890	.521	.660
H-Rule	.672	.900	.715	.797	.681	**1.00**	.645	.784
AMIE	.385	**.999**	.316	.481	.441	.874	.224	.356
CAGED	.853	.963	.856	.906	.839	.850	.857	.900
GPT-2 XL	.823	.562	.468	.511	.787	.899	.860	.879
LLM_sim	.911	.934	.969	.951	.806	.962	.816	.883
LLM_sim wo AC	.868	.980	.871	.922	.806	.969	.810	.883
LLM_rule_one	–	–	–	–	.828	.909	.898	.904
LLM_rule_one wo VAL	.884	.939	.932	.935	.825	.907	.898	.903
LLM_rule_maj	–	–	–	–	.820	.928	.868	.897
LLM_rule_maj wo VAL	**.926**	.939	**.980**	**.960**	**.894**	.933	**.950**	**.942**
LLM_rule_all	–	–	–	–	.818	.912	.883	.897
LLM_rule_all wo VAL	.876	.933	.929	.931	.833	.915	.898	.906

Overall, the variants of *LLM_sim* and *LLM_rule* consistently outperform the other baselines across most metrics, particularly in terms of accuracy and F1-measure. This clearly demonstrates the effectiveness and robustness of our LLM-based approaches for detecting noisy triples in KGs.

Comparing the baseline methods, the embedding-based KGE models (TransE, RotatE, and ExpressivE) generally exhibit limited effectiveness in noise detection, particularly as the noise ratio increases. This performance degradation underscores their inherent limitation: while these models are designed to capture latent structural patterns within KGs, they lack specialized mechanisms to explicitly identify and filter erroneous triples. Consequently, their embeddings become increasingly unreliable in noisy environments, leading to diminished accuracy and F1 scores. The handcrafted rule-based baseline (H-Rule) presents a contrasting behavior. It consistently achieves high precision, reflecting its conservative decision-making process and its capacity to generate accurate predictions for the subset of noisy triples. However, this comes at the expense of recall, since the handcrafted rules are unable to capture the full diversity of noise patterns present in large-scale and heterogeneous KGs. This limitation results in many noisy triples being overlooked, thereby restricting its overall effectiveness. Furthermore, to contextualize the per-

Table 4.3: Results of noise detection on WN18RR and FB15k-237 with 20% noise ratio. A, P, R, and F stand for accuracy, precision, recall, and F-measure, respectively.

Model	WN18RR				FB15k-237			
	A	P	R	F	A	P	R	F
TransE	.335	.708	.305	.426	.353	.724	.325	.449
RotatE	.423	.774	.407	.533	.438	.782	.425	.551
ExpressivE	.593	.810	.558	.661	.501	.811	.501	.619
H-Rule	.634	.811	.715	.760	.719	**1.00**	.653	.790
AMIE	.419	**.999**	.282	.440	.442	.944	.368	.531
CAGED	.804	.933	.814	.869	.712	.694	.927	.794
GPT-2 XL	.786	.811	.960	.829	.775	.809	.945	.872
LLM_sim	.883	.894	.971	.930	.784	.918	.806	.858
LLM_sim wo AC	.869	.960	.875	.915	.798	.935	.807	.866
LLM_rule_one	–	–	–	–	.813	.815	.987	.897
LLM_rule_one wo VAL	.868	.880	.969	.923	.841	.844	.988	.911
LLM_rule_maj	–	–	–	–	.816	.820	.994	.898
LLM_rule_maj wo VAL	**.891**	.896	**.979**	**.936**	**.878**	.848	**.988**	**.929**
LLM_rule_all	–	–	–	–	.822	.859	.934	.895
LLM_rule_all wo VAL	.863	.872	.973	.920	.818	.859	.929	.983

formance of LLM-based methods against traditional symbolic approaches, we evaluated **AMIE**, a representative automated rule mining system. As shown in Tables 4.2, 4.3, and 4.4, AMIE exhibits a distinct performance profile characteristic of statistical rule mining: it achieves near-perfect precision (> 0.99 across datasets) but suffers from prohibitively low recall (ranging from 0.25 to 0.46). This confirms that while statistical rule mining is highly reliable for frequent and well-supported patterns, it fails to cover the sparse, implicit, or long-tail relations that effectively evade statistical detection. In contrast, our LLM-based methods maintain high precision while significantly improving recall (> 0.80), demonstrating the value of semantic reasoning in handling data sparsity. In comparison, GPT-2 XL leverages its pre-trained language understanding to achieve moderate performance in this task. Its ability to encode general semantic and syntactic regularities allows it to detect certain types of implausible triples. Nonethelessarchitectural advances, its performance remains inferior to that of the vanilla Llama3 model (*LLM_sim wo AC*), which highlights two important insights. First, architectural advances in LLMs contribute significantly to improved reasoning over KG triples. Second, GPT-2 XL is more sensitive to prompt formulation. Since it lacks strong instruction-following ability, it cannot fully utilize prompts

Table 4.4: Results of noise detection on WN18RR and FB15k-237 with 30% noise ratio. A, P, R, and F stand for accuracy, precision, recall, and F-measure, respectively.

Model	WN18RR				FB15k-237			
	A	P	R	F	A	P	R	F
TransE	.310	.553	.278	.370	.322	.568	.291	.385
RotatE	.377	.630	.352	.452	.403	.655	.383	.483
ExpressivE	.504	.728	.511	.600	.486	.720	.483	.578
H-Rule	.600	.730	.716	.723	.749	**1.00**	.656	.792
AMIE	.453	**.999**	.250	.399	.461	.879	.410	.578
CAGED	.734	.895	.703	.788	.732	.702	.892	.785
GPT-2 XL	.712	.730	.960	.829	.717	.737	.953	.831
LLM_sim	.853	.850	.970	.906	.777	.842	.855	.848
LLM_sim wo AC	.865	.934	.875	.904	.798	.903	.809	.854
LLM_rule_one	–	–	–	–	.828	.865	.909	.886
LLM_rule_one wo VAL	.844	.839	**.973**	.901	.824	.870	.894	.882
LLM_rule_maj	–	–	–	–	.833	.836	.962	.895
LLM_rule_maj wo VAL	**.875**	.896	.939	**.917**	**.879**	.863	**.994**	**.924**
LLM_rule_all	–	–	–	–	.815	.936	.932	.881
LLM_rule_all wo VAL	.831	.840	.950	.892	.818	.937	.935	.883

designed for more advanced models like Llama3. This gap indicates that prompt effectiveness is model-dependent, and that noise detection performance relies not only on pre-training and model size, but also on how well the prompt aligns with the model's instruction-following capability.

Focusing on the ablation study of *LLM_sim*, the impact of *AC* exhibits a nuanced pattern depending on the noise density and domain. In the s-NELL dataset and WN18RR at a lower noise ratio (10%), removing context (*LLM_sim wo AC*) leads to a decline in performance, confirming the effectiveness of AC in grounding the model with external evidence. However, as shown in Tables 4.3 and 4.4, at higher noise ratios (20% and 30%) on WN18RR, *LLM_sim wo AC* performs comparably to, or even slightly better than, the context-aware *LLM_sim*. We attribute this phenomenon to **Context Pollution**: as the KG becomes densely populated with noise, the retrieval mechanism is more likely to fetch erroneous triples as context, which can mislead the semantic reasoning of the LLM. In such high-noise scenarios for general domains, the model's internal parametric knowledge (utilized in wo AC) proves more robust than the polluted external context.

By providing supplementary context, the LLM can disambiguate entities with similar surface forms, recognize plausible relations more accurately, and

distinguish between correct and noisy triples with greater reliability.

Furthermore, comparing *LLM_sim* with *LLM_sim wo AC* in Tables 4.2, 4.3, and 4.4 on the FB15k-237 dataset, the performance gap is minimal, and in some metrics, the variant without context performs comparably. We attribute this to the nature of the dataset. FB15k-237 contains general-world knowledge (e.g., geography, movies) that is heavily represented in the pre-training corpora of LLMs. Consequently, the model relies on its strong parametric memory to validate these triples. In such cases, retrieved additional context may become redundant or even distracting. In contrast, for domains like s-NELL or specific relations in WN18RR where the LLM lacks sufficient internal knowledge, the Additional Context proves crucial by providing necessary anchors for reasoning. This divergence confirms that AC is highly effective for domain-specific or long-tail knowledge but optional for common-sense facts grounded in the model's parameters.

For *LLM_rule*, the evaluation results demonstrate that *LLM_rule_maj* consistently achieves the most favorable balance between precision and recall across all datasets. The variant that validates a triplet using only a single supporting rule (*LLM_rule_one*) tends to be overly permissive. While this approach often leads to higher recall, it simultaneously compromises precision, as noisy triples are more likely to be misclassified as correct if they happen to satisfy just one rule, even when the majority of evidence suggests otherwise. In contrast, the strict requirement that all rules must be satisfied (*LLM_rule_all*) enforces a highly conservative decision criterion. Although this yields strong precision, it substantially reduces recall, as many valid triples may be incorrectly rejected due to the absence of unanimous rule support. The majority-vote strategy employed in *LLM_rule_maj* provides a principled middle ground between these two extremes. Compared with *"wo VAL"*, we found that judging whether the generated rule is valid has little impact on the result. We believe that this is because the generated rules are independent of each other, so even if the rule is not valid, it will not have much impact on our judgment result in *LLM_rule_maj* and *LLM_rule_all*.

A direct comparison between *LLM_sim* and *LLM_rule* reveals that the incorporation of rule-augmented reasoning consistently yields superior detection performance across all three datasets. This finding underscores the importance of complementing purely semantic inference from LLMs with explicit symbolic reasoning. By leveraging logical rules as an additional layer of validation, *LLM_rule* not only improves predictive accuracy but also mitigates over-reliance on implicit statistical correlations learned during pre-training. Equally important, the integration of rules enhances the interpretability and reliability of the detection process. Unlike purely neural approaches, the decisions made by *LLM_rule* can be traced back to concrete

Table 4.5: Results of noise detection on s-NELL

Model	A	P	R	F
TransE	.564	.734	.709	.721
RotatE	.670	.767	.842	.802
ExpressivE	.694	.773	.872	.819
H-Rule	.728	**1.00**	.658	.794
CAGED	.849	**1.00**	.678	.808
GPT-2 XL	.772	.955	.798	.870
LLM_sim	.840	.868	.942	.904
LLM_sim wo AC	.804	.839	.932	.883
LLM_rule_one	.848	.978	.856	.915
LLM_rule_one wo VAL	.864	.975	.870	.919
LLM_rule_maj	.858	.979	.861	.917
LLM_rule_maj wo VAL	**.884**	.985	.883	**.931**
LLM_rule_all	.860	.979	.863	.918
LLM_rule_all wo VAL	.794	.989	.799	.884

rules or constraints, providing a transparent rationale for each classification outcome. Such traceability is particularly valuable in knowledge-centric applications, where explainability and accountability are essential. Overall, these results highlight that combining LLM inference with structured rule-based reasoning offers a balanced paradigm that achieves both strong empirical performance and improved interpretability.

Impact of Rule Source: LLM vs. AMIE

To further validate the quality of the rules generated by our framework, we conducted a substitution experiment. Specifically, we replaced the semantic rules generated by *LLM_rule* with logical rules mined by the traditional AMIE algorithm and evaluated the noise detection performance on WN18RR and the original FB15k datasets under a 10% noise setting. The results are reported in Table 4.6.

As observed, replacing LLM-generated rules with AMIE-mined rules leads to a catastrophic degradation in performance. While Precision remains relatively high, indicating that the few rules AMIE finds are generally correct, the accuracy and recall are critically low. This confirms the limitations of traditional rule mining in noisy datasets: statistical methods like AMIE require high data fidelity to establish support for rules. In the presence of noise, valid patterns are obscured, leading to a sparse rule set that fails to cover the

Table 4.6: Comparison of noise detection performance between AMIE-mined rules and LLM-generated rules (10% Noise Ratio).

Dataset	Rule Source	Accuracy	Precision	Recall	F1 Score
WN18RR	AMIE	0.193	0.894	0.190	0.304
	LLM_rule	**0.926**	**0.939**	**0.980**	**0.960**
FB15k*	AMIE	0.176	0.796	0.170	0.281
	LLM_rule	**0.929**	**1.000**	**0.929**	**0.963**

*Results on the original FB15k dataset (aligned with the substitution experiment settings).

majority of the graph. Furthermore, the results suggest that unreasonable or incomplete rules mined from noise can confuse the model, introducing spurious constraints that actively hinder the identification of valid triples, thereby severely compromising the overall model quality compared to the robust, semantic-grounded rules generated by our approach.

4.3 Evaluation of Noise Refinement

4.3.1 Experimental Settings of Noise Refinement

Due to the lack of ground-truth annotations for noise refinement on the WN18RR and FB15k-237 datasets, it is not feasible to directly measure refinement performance against a fully verified reference set. To address this limitation, we employ two complementary evaluation criteria that jointly capture different aspects of refinement quality.

The first criterion is the *restoration ratio* (RR), which is defined as the proportion of cases where the refinement process successfully recovers the original triplets from the clean KG (before the artificial injection of noise) to the total number of triplets identified as noisy. This metric serves as a quantitative indicator of the system's ability to restore factual correctness and recover the underlying structure of the KG. A higher RR suggests that the refinement method replaces a noisy triple with the most plausible and contextually accurate triplets. The second criterion is the *accuracy* (ACC), which relies on human evaluation to account for semantic fidelity beyond synthetic restoration. Specifically, we randomly sample 100 refined triplets and manually verify whether the outputs correspond to valid and semantically meaningful KG statements. This criterion ensures that the evaluation is not limited to reproducing pre-injected ground truth but also reflects the

broader capacity of the refinement process to generate statements that are logically sound, contextually coherent, and factually reliable. By incorporating manual inspection, ACC provides an additional layer of robustness to the evaluation, complementing the automated measure of RR.

For the s-NELL dataset, ground-truth annotations are directly available through manual curation, which provides a reliable benchmark for assessing refinement quality. As a result, only the ACC metric is employed for evaluation in this setting, since it directly reflects the semantic correctness of the refined triplets against human-validated labels. This approach avoids the need for proxy measures such as restoration ratio, thereby ensuring that the evaluation remains closely aligned with the actual ground-truth semantics of the dataset.

Across all experiments, we systematically report the refinement performance of both *LLM_sim* and *LLM_rule*, applied exclusively to the set of triplets identified as noise by their respective detection modules. For *LLM_rule*, we specifically employ the *LLM_rule_maj wo VAL* variant for noise detection, as prior evaluations consistently demonstrated that this majority-voting strategy offers the most favorable balance between precision and recall. By grounding the refinement stage on the outputs of *LLM_rule_maj wo VAL*, we ensure that the refinement module is tested under the most effective and representative detection scenario, thereby providing a fair and rigorous assessment of its true capability.

4.3.2 Results of Noise Refinement

Table 4.7 reports the refinement results across the three benchmark datasets. The results clearly demonstrate that the proposed methods consistently achieve high levels of accuracy under diverse settings, ranging from 0.783 to 0.940 across varying datasets and noise ratios. Such consistently strong performance highlights not only the robustness of the proposed framework but also its adaptability to heterogeneous KG structures and noise characteristics. In particular, the ability to maintain high accuracy under different noise levels suggests that the refinement process is effective in both low- and high-noise scenarios, thereby underscoring the generalizability of the approach. A closer inspection of the results further reveals that *LLM_rule* consistently outperforms *LLM_sim* in terms of accuracy across all evaluation settings. This advantage indicates that the incorporation of explicit logical rules yields a more reliable mechanism for correcting noisy triples than strategies that rely solely on semantic similarity. By leveraging rule-based reasoning, the refinement process can capture structural dependencies and relational regularities that are intrinsic to the KG, which semantic similarity methods often over-

look. As a result, *LLM_rule* demonstrates a stronger capability to preserve semantic coherence and enforce logical consistency, leading to superior overall refinement performance.

An important distinction between the two refinement strategies lies in their underlying mechanisms. Specifically, the refinement process of *LLM_rule* operates independently of explicit comparisons with the original KG. This design enables the model to generate novel triples that are semantically coherent and contextually valid, even if such triples are not present in the gold-standard dataset. While this capability enhances the semantic completeness and expressiveness of the refined KG, it also introduces a methodological limitation in terms of evaluation. In particular, the restoration ratio (RR) metric exclusively measures the proportion of refined triples that align with the original clean KG prior to noise injection. As a result, semantically valid but novel triples produced by *LLM_rule* are not credited, leading to comparatively lower RR scores despite the method achieving consistently higher accuracy. This observation underscores an inherent trade-off between the two evaluation criteria. While the RR metric mainly rewards methods that simply recover the original triples, the ACC metric better captures improvements in semantic validity and factual correctness introduced by refinement approaches such as *LLM_rule*. Taken together, these complementary perspectives highlight the importance of employing multiple evaluation metrics to fully capture the strengths and limitations of different refinement strategies.

To further validate the necessity of our structured refinement framework, we analyzed the performance of the *LLM_zero_shot* baseline. As shown in Table 4.7, this naive approach yields critically low performance, with Accuracy hovering between 0.08 and 0.27 and a Restoration Ratio (RR) of 0.00 across all datasets. This complete failure to restore ground-truth triples highlights a fundamental limitation of generative LLMs in KG tasks: without explicit grounding (such as our Grouping and Similarity strategies in *LLM_sim* or logical constraints in *LLM_rule*), the model tends to hallucinate plausible-sounding but factually incorrect entities, or generate entities that do not exist in the target KG schema. The sharp contrast between the near-failure of *LLM_zero_shot* and the high accuracy (> 0.80) of our proposed methods empirically proves that unconstrained generation is insufficient for KG refinement, and that our proposed constraints are essential for aligning LLM outputs with graph structures.

Table 4.7: Results of noise refinement. RR and ACC stand for the restoration ratio and accuracy, respectively. We include LLM_zero_shot results to highlight the necessity of our structured refinement strategy.

NR	Model	WN18RR		FB15k-237		s-NELL
		RR	ACC	RR	ACC	ACC
10%	*LLM_sim*	.824	.890	.872	.920	–
	LLM_rule	.150	.900	.154	.920	–
	LLM_zero_shot	.000	.210	.000	.270	–
20%	*LLM_sim*	.808	.870	.853	.880	–
	LLM_rule	.211	.890	.231	.940	–
	LLM_zero_shot	.000	.100	.000	.130	–
30%	*LLM_sim*	.784	.820	.834	.830	–
	LLM_rule	.280	.850	.218	.910	–
	LLM_zero_shot	.000	.120	.000	.080	–
—	*LLM_sim*	–	–	–	–	.783
	LLM_rule	–	–	–	–	.848
	LLM_zero_shot	–	–	–	–	.211

4.4 Evaluation of KGC

4.4.1 Experimental Settings of KGC

KGC is a fundamental and widely studied task in the field of knowledge representation and reasoning. Its primary objective is to predict missing relations between entities within a KG, thereby improving the completeness, consistency, and overall utility of the KG for downstream applications such as question answering, recommendation systems, and semantic search. Accurate KGC not only fills gaps in knowledge but also enhances the reliability of inference and decision-making processes built upon the KG. While the direct evaluation metrics (Accuracy and Restoration Ratio) in Section 4.3 assess the *semantic correctness* of individual triples, they do not quantify the *structural integrity* of the graph as a whole. We employ KGC as a downstream extrinsic evaluation to measure the practical utility of the refined KG. The rationale is that noise in KGs disrupts the latent feature space learned by embedding models. If our refinement method successfully restores valid connections and removes inconsistencies, the global structure of the KG should be improved, allowing embedding models to learn more accurate representations. Thus,

improvements in KGC metrics (MRR, Hits@k) serve as a functional proxy for validating that the refined graph is not only textually correct but also structurally robust for downstream reasoning. In this experiment, we adopt the ExpressivE model as the representative KGC framework due to its capability to capture complex relational patterns and its demonstrated effectiveness on benchmark datasets. The model is trained on a KG that has been preprocessed and refined either by our proposed LLM-based methods or by established baseline approaches. Its predictive performance is then systematically evaluated on a held-out test set to ensure unbiased assessment. To mitigate overfitting and ensure robust model generalization, the validation set is employed for early stopping and hyperparameter optimization.

For quantitative evaluation, we utilize two widely accepted metrics in KGC research. The first is the *Mean Reciprocal Rank* (MRR), which calculates the average of the reciprocal ranks of the correctly predicted relations across all test queries. MRR provides a nuanced measure of ranking quality, emphasizing the model's ability to assign higher ranks to the true relations, thereby reflecting its precision in ranking plausible candidates. The second metric is *Hit@k* (with $k = 1, 3, 10$), which measures the proportion of test instances in which the correct relation appears within the top-k predictions. This metric offers an interpretable indication of the model's success in capturing the most relevant relations and its practical utility in downstream applications where top-ranked predictions are typically prioritized.

For baseline comparisons, we include both **CAGED**, a state-of-the-art KGC method specifically designed to maintain robust performance under noisy conditions, and **GPT-2 XL**, which serves as a pre-trained language model baseline capable of providing semantic priors without task-specific fine-tuning. These baselines establish a reference for evaluating the effectiveness of our proposed approaches.

To provide a comprehensive assessment of our methods, we evaluate multiple variants of the proposed framework. Specifically, we consider *LLM_sim*, which performs similarity-based noise detection and refinement; *LLM_sim wo AC*, which excludes the additional contextual information in the prompt to measure the impact of context on refinement quality; and *LLM_rule_maj* (hereafter referred to simply as *LLM_rule* in this section), which incorporates rule-augmented reasoning with a majority-vote strategy to guide the refinement process. These variants are consistent with the definitions provided in Section 4.2 and 4.3, ensuring comparability across experiments. In addition, we include two ablated settings, *LLM_sim wo ref* and *LLM_rule wo ref*, in which noisy triples are detected but not refined, and are instead simply removed from the KG. These ablation experiments are critical for disentangling the relative contributions of noise detection and noise refine-

ment: by comparing performance with and without the refinement step, we can quantify the added value of the refinement module in enhancing KGC performance. The comparison among the aforementioned systems allows us to rigorously isolate the effects of contextual augmentation, rule-based reasoning, and refinement strategies on the overall predictive capability of the KGC model.

For reference, we include two reference settings to contextualize the performance impact of noise and refinement. The first is the "Original KG", which corresponds to the clean KG before any noise is injected. It serves as an upper bound for downstream KGC performance. The second is the "No Refinement" setting, which uses the noisy KG directly without applying any detection or correction.

4.4.2 Results of KGC

Tables 4.8 and 4.9 present the experimental outcomes of the KGC task across three widely used benchmark datasets. A direct comparison between the "Original KG" (0% noise ratio) and the "No Refinement" configurations in Table 4.8 clearly illustrates that as the proportion of noise in the KG increases, the performance of the KGC model deteriorates in a pronounced and consistent manner. This trend provides strong empirical evidence that the predictive capacity of KGC models is inherently sensitive to the structural and semantic quality of the underlying KG. In other words, even modest levels of noise can substantially compromise the reliability of link prediction, thereby reducing the completeness and usability of the KG for downstream applications. These findings highlight the critical importance of deploying robust noise detection and refinement strategies as indispensable preprocessing steps to safeguard and enhance the effectiveness of KGC.

A strong and consistent positive correlation is observed between the accuracy of noise detection and the downstream effectiveness of KGC. In particular, our proposed approaches, *LLM_sim* and *LLM_rule*, achieve superior performance across all benchmark settings when compared to established baselines. This consistent advantage demonstrates that enhancing the quality of the KG through precise noise identification and refinement directly translates into significant improvements in predictive accuracy for KGC models. More broadly, these findings provide compelling empirical evidence that noise mitigation should not be regarded as a peripheral or optional preprocessing step. Instead, it constitutes a central determinant of the overall success of KGC, with direct implications for the reliability, robustness, and utility of knowledge-driven systems in downstream applications.

When comparing the two proposed refinement strategies, *LLM_rule* gen-

erally demonstrates superior effectiveness over *LLM_sim*, with the sole exception occurring under the 30% noise ratio on the WN18RR dataset. This exception underscores an important nuance: rule-based reasoning appears particularly well suited to domains where relations are predominantly fact-oriented and structurally grounded, such as FB15k-237 and NELL. In these settings, explicit logical rules derived by the LLM can effectively capture relational regularities, leading to more accurate noise refinement. By contrast, in linguistically oriented datasets such as WN18RR, where relations often rely on subtle semantic associations rather than strict factual regularities, semantic similarity plays a comparatively greater role in supporting accurate inference. These observations highlight the complementary strengths of semantic similarity and rule-based reasoning, suggesting that their integration may offer a more robust framework for handling the diverse relational structures encountered across different KGs. The integration of *LLM_sim* and *LLM_rule* remains in future work.

The results of the ablation studies, reported in Tables 4.8 and 4.9, offer deeper insights into the functional contributions of individual system components. A direct comparison between *LLM_sim* and its ablated variant *LLM_sim wo AC* underscores the critical role of contextual augmentation: the removal of additional contextual triplets leads to a measurable and consistent decline in performance. This finding demonstrates that enriching LLM prompts with semantically related evidence substantially enhances the model's ability to detect and refine noisy triples. Furthermore, the ablated settings *LLM_sim wo ref* and *LLM_rule wo ref*, which perform noise detection but eliminate noise refinement process, consistently yield inferior results relative to their fully refined counterparts. This degradation illustrates that the refinement stage is not merely an auxiliary step but a vital mechanism for restoring missing or corrupted factual knowledge, thereby strengthening the structural integrity and semantic validity of the KG. Taken together, these findings confirm that both detection and refinement modules play indispensable and complementary roles, and that their joint operation is essential for realizing the full potential of LLM-based noise mitigation in KGC tasks.

Performance on Original FB15k Dataset

To further investigate the robustness of our framework, we applied the best-performing model, *LLM_rule*, to the original FB15k-237 dataset. Table 4.10 summarizes the performance across all three stages: noise detection, refinement, and downstream KGC. The results indicate that *LLM_rule* achieves exceptional precision and accuracy in identifying and refining noisy triples, benefiting from the dataset's dense relational patterns. However, these im-

provements translate into only limited gains in the downstream KGC task (MRR 0.213). This discrepancy suggests that while rule-based refinement successfully enforces logical consistency, the redundancy inherent in FB15k-237 limits the marginal benefit of these corrections for embedding-based link prediction models.

Table 4.8: Results of KGC task on WN18RR and FB15k-237

	WN18RR				FB15k-237			
	MRR	Hit@1	Hit@3	Hit@10	MRR	Hit@1	Hit@3	Hit@10
Original KG	.506	.459	.519	.597	.212	.148	.235	.339
10% Noise Ratio								
No refinement	.423	.375	.448	.508	.191	.128	.214	.319
CAGED	.400	.343	.429	.505	.176	.117	.195	.298
GPT2 XL	.412	.359	.433	.467	.143	.106	.156	.216
LLM_sim wo ref	.434	.380	.460	.533	.173	.116	.190	.292
LLM_sim wo AC	.402	.343	.430	.509	.174	.116	.193	.293
LLM_sim	.404	.346	.431	.510	.183	.122	.206	.302
LLM_rule wo ref	**.441**	**.388**	**.464**	**.539**	.184	.123	.206	.310
LLM_rule	.427	.313	.400	.527	**.244**	**.158**	**.264**	**.419**
20% Noise Ratio								
No refinement	.363	.317	.396	.435	.174	.116	.192	.293
CAGED	.351	.295	.381	.450	.168	.115	.185	.276
GPT2 XL	.338	**.396**	.370	.404	.171	.115	.188	.286
LLM_sim wo ref	.390	.333	.421	**.487**	.170	.115	.187	.284
LLM_sim wo AC	.370	.312	.401	.473	.175	.117	.196	.294
LLM_sim	.376	.318	.407	.481	.178	.120	.201	.306
LLM_rule wo ref	**.391**	.337	**.421**	.487	.177	.117	.198	.301
LLM_rule	.382	.315	.407	.485	**.237**	**.148**	**.259**	**.416**
30% Noise Ratio								
No refinement	.290	.246	.324	.358	.141	.109	.152	.201
CAGED	.294	.241	.327	.382	.167	.116	.185	.273
GPT2 XL	.321	.271	.356	.401	.167	.114	.184	.278
LLM_sim wo ref	.343	.289	.375	.434	.166	.113	.185	.277
LLM_sim wo AC	.335	.280	.367	.429	.169	.115	.188	.281
LLM_sim	**.370**	**.312**	**.391**	**.463**	.177	.119	.199	.295
LLM_rule wo ref	.344	.289	.375	.438	.172	.115	.191	.281
LLM_rule	.344	.295	.384	.436	**.229**	**.141**	**.252**	**.409**

Table 4.9: Results of KGC task on s-NELL

Model	MRR	Hit@1	Hit@3	Hit@10
No refinement	.231	.200	.200	.300
CAGED	.086	.037	.081	.108
GPT2 XL	.068	.102	.102	.178
LLM_sim wo ref	.133	.105	.105	184
LLM_sim wo AC	.139	.111	.111	.222
LLM_sim	.144	.089	.178	.200
LLM_rule wo ref	.159	.108	.162	.216
LLM_rule	**.284**	**.214**	**.357**	**.357**

Table 4.10: Results of LLM_rule on the original FB15k dataset across detection, refinement, and KGC tasks.

Task	**Metrics**			
Noise Detection	Accuracy	Precision	Recall	F1 Score
	0.929	1.000	0.929	0.963
Noise Refinement	ACC	RR	-	-
	0.950	0.312	-	-
KGC	MRR	Hit@1	Hit@3	Hit@10
	0.213	0.149	0.235	0.341

Chapter 5

Error Analysis on Noise Detection

While quantitative metrics demonstrate overall performance, they cannot fully reveal the specific strengths and weaknesses of each approach. To address this, we complement our experimental results with a detailed qualitative analysis of failure cases. To ensure reliability, we first conducted a manual inspection of the rationales generated for a random sample of 50 triples. This verification confirmed that the models' decisions were grounded in logical reasoning rather than spurious correlations. Crucially, this analysis highlighted a distinct cognitive boundary between the two methods: failures in *LLM_sim* are primarily driven by the nature of knowledge (e.g., linguistic nuances vs. factual memory), whereas failures in *LLM_rule* are structurally driven by graph characteristics (e.g., graph density and sparsity). These insights form the basis of the error pattern categorization detailed below.

5.1 Error Patterns in *LLM_sim*

Although *LLM_sim* performs well in detecting and refining noisy triples, its decisions rely heavily on semantic similarity signals derived from contextual embeddings. Unlike rule-based or type-constrained approaches, *LLM_sim* does not explicitly enforce domain-scope compatibility, nor ground relational semantics in the structural properties of the KG. As a result, certain types of errors tend to recur across different datasets, reflecting the model's limitations in interpreting relational meanings, identifying entity roles, and handling superficial associations.

5.1.1 Categorization of Error Types

To support a consistent and non-redundant analysis, we organize all observed model errors into three categories: *type errors*, *semantic misalignment*, and *hallucinations*. They capture the most recurrent patterns of failure made by *LLM_sim* and are applicable uniformly to WN18RR, FB15k-237, and NELL.

1. **Type Error**
   A type error occurs when the head or tail entity in a triple violates the implicit or explicit domain–range constraints of the relation. In other words, even if a triple appears lexically or contextually plausible, it is invalid because the entity belongs to an incompatible semantic class. This kind of error does not stem from misunderstanding the relational meaning, but rather from the model's inability to incorporate type-aware validation. For example, assigning a *Location* entity to a relation whose range is restricted to *Person* constitutes a type error, regardless of surface-level fluency.

2. **Semantic Misalignment**
   Semantic misinterpretation occurs when the model preserves the correct relation but assigns it to a head-tail pair that does not satisfy the relation's intended semantic scope. In contrast to type errors, the involved entities may belong to admissible classes, yet the predicate is applied in an overly broad, overly narrow, or contextually inappropriate manner. Such errors typically arise when the model relies on surface-level association or distributional proximity without fully capturing the relational constraints encoded in the KG. The result is not a violation of entity type, but a mismatch between the true semantics of the relation and the way it is instantiated in the triple.

3. **Hallucination**
   Hallucination refers to errors caused by superficial lexical cues, embedding-based associations, or frequency biases that override factual correctness. In contrast to type errors and semantic misinterpretation, hallucinations do not necessarily violate domain–range constraints or relational semantics; instead, the model fabricates or accepts a triple based on spurious correlations or uninformed guesses. This phenomenon commonly appears when the model overgeneralizes from co-occurrence patterns or prioritizes linguistic plausibility over KG grounding, leading to factually unsupported predictions.

To gain a deeper understanding of these limitations, we analyzed incorrect predictions made by *LLM_sim* on WN18RR, FB15k-237, and NELL. Ta-

ble 5.1 summarizes one example from each category across the three datasets. In the following subsections, we analyze each error type in detail, explaining why it occurs and why *LLM_sim* is particularly prone to these errors.

5.1.2 Type errors in *LLM_sim*

Table 5.1: Representative Examples of LLM_sim Errors Across Datasets

Error Category	**Dataset**	**Example**
Type Error	WN18RR	(*nuclear_family.n.01*, _hypernym, *nutriment.n.01*)
	FB15k-237	(*M._C._Gainey*, /people/person/gender, *Brooklyn*)
	NELL	(*Bangladesh*, cityliesonriver, *Ganges*)
Semantic Misinterpretation	WN18RR	(*united_kingdom.n.01*, _derivationally_related_form, *stream.n.04*)
	FB15k-237	(*Edward_Fox*, /film/performance/film, *The_Sting*)
	NELL	(*Breakfast*, bakedgoodservedwithbeverage, *Tea*)
Hallucination	WN18RR	(*aeolis.n.01*, _instance_hypernym, *governor's_plum.n.01*)
	FB15k-237	(*Hollyoaks*, /tv/tv_program/genre, *Vegetarianism*)
	NELL	(*Sherlock_Holmes*, bookwriter, *Arthur_Conan_Doyle*)

Type errors arise when *LLM_sim* retains or accepts triples whose head or tail entities violate the implicit domain-range constraints of a relation.

A representative example comes from WN18RR: *(nuclear_family.n.01*, _hypernym, *nutriment.n.01)*. The _hypernym relation in WordNet expresses hierarchical semantic inclusion between concepts of the same class (e.g., *dog.n.01* $\rightarrow$ *canine.n.02*). However, "nuclear family" denotes a social unit, whereas "nutriment" refers to an edible substance. Despite the categorical mismatch, *LLM_sim* assigns plausibility to the triple because the surface-level semantic embedding of both nouns places them in proximate regions of the vector space. The model's similarity-based heuristic implicitly treats co-occurrence and topical association as sufficient indicators of compatibility, failing to enforce the relational constraint that hypernym pairs should belong to the same ontological domain.

A similar pattern is evident in FB15k-237. In the triple *(M._C._Gainey*, /people/person/gender, *Brooklyn*), the head entity is correctly recognized as a person. However, the tail entity "Brooklyn" is a location rather than a

valid gender label. Because *LLM_sim* does not incorporate an explicit range restriction for the relation `/people/person/gender`, it does not reject the triple based on type incompatibility. Instead, it relies on semantic proximity in pre-trained embeddings, where named entities referring to people and places are often closely clustered due to shared lexical contexts.

In the NELL dataset, the triple *(Bangladesh*, cityliesonriver, *Ganges)* provides another clear instance of a type error. Although "Bangladesh" and "Ganges" are geographically related, the relation *cityliesonriver* presupposes that the head entity is of type *City*. Because *LLM_sim* lacks access to explicit type hierarchies or KG schemas, it relies on contextual similarity between location-based entities. As a result, a country is incorrectly treated as a plausible head for a city-specific relation.

Across all three datasets, these errors demonstrate that *LLM_sim* does not inherently distinguish between semantically related entities that differ in type. When distributional similarity outweighs structural constraints, triples with invalid domain–range assignments are falsely accepted or insufficiently penalized. This limitation leads to systematic false negatives and highlights the absence of type-aware judgment in similarity-driven refinement.

5.1.3 Semantic Misinterpretation in *LLM_sim*

A second class of errors arises when *LLM_sim* preserves the correct relation but assigns it to a head-tail pair that does not satisfy its intended semantics.

In WN18RR, this problem is exemplified by the triple *(united_kingdom.n.01, _derivationally_related_form, stream.n.04)*. Both arguments are nouns and therefore type-compatible with lexical relations in WordNet. However, the predicate _derivationally_related_form is restricted to words that share morphological roots (e.g., "govern" and "government"). The model incorrectly treats topical or co-occurrence-based association as sufficient evidence for a derivational link, indicating that it fails to capture the precise semantic constraint embedded in the relation.

A similar misalignment occurs in FB15k-237. In the triple *(Edward_Fox, /film/performance/film, The_Sting)*, the actor and film entities are correctly typed, but the underlying KG schema encodes participation through an intermediate performance node rather than a direct binary relation. Since *LLM_sim* does not account for multi-hop structural requirements or relation-specific instantiation patterns, it treats the predicate as a direct link and retains the triple based on surface semantic compatibility.

The NELL dataset illustrates another manifestation of semantic misinterpretation with the example *(Breakfast, bakedgoodservedwithbeverage, Tea)*. "Breakfast" is conceptually related to food and beverages, but the predi-

cate *bakedgoodservedwithbeverage* presupposes that the head entity denotes a baked item. The model broadens the relation's scope to any edible concept co-occurring with drinks, prioritizing thematic consistency over predicate-specific constraints. This reveals that the model does not enforce the narrower relational semantics required by the KG.

Across all three datasets, semantic misinterpretation reflects a tendency of *LLM_sim* to conflate contextual association with relational correctness. Rather than violating type requirements, the model applies the relation in contexts where its meaning is loosely but not accurately satisfied. These errors highlight a limitation of similarity-based reasoning, in which predicate intent is approximated through proximity rather than grounded in the structural semantics of the KG.

5.1.4 Hallucination in *LLM_sim*

A third source of errors in *LLM_sim* stems from hallucination or surface-level bias. In these cases, the model does not merely broaden the semantics of a relation but generates or accepts triples based on lexical association, token-level proximity, or frequency-driven priors. The resulting predictions are not grounded in KG-specific semantics. They reflect overgeneralization from language modeling signals that lack factual alignment with the underlying graph.

In WN18RR, this phenomenon is illustrated by the triple *(aeolis.n.01, _instance_hypernym, governor's_plum.n.01)*. Both entities refer to uncommon lexical senses, and their embeddings may occupy sparse or noisy regions of semantic space. Rather than rejecting the relation due to a lack of conceptual hierarchy, *LLM_sim* infers plausibility from faint topical co-occurrences. The model effectively "hallucinates" a hypernymic connection in the absence of any legitimate hierarchical link.

A comparable error appears in FB15k-237 with *(Hollyoaks, /tv/tv_program/genre, Vegetarianism)*. Here, the head entity is a television show, and the predicate expects a genre label. However, *LLM_sim* treats "Vegetarianism" as semantically related to lifestyle-oriented or thematic categories that co-occur with entertainment content. The model does not fabricate the entities themselves but hallucinates a genre relationship based on surface-level thematic relevance rather than factual correctness.

In the NELL dataset, the triple *(Sherlock_Holmes, bookwriter, Arthur_Conan_Doyle)* demonstrates a different form of surface bias. Since the two entities frequently appear together in textual corpora, *LLM_sim* implicitly assumes a symmetric authorship or role-based relation. The tail entity is valid in the KG, yet the predicate *bookwriter* implies that the head is the

author, not the subject, of the written work. This inversion is not a misunderstanding of type or relation scope, but a hallucinated link supported by distributional association.

5.2 Error Patterns in *LLM_rule*

Compared to *LLM_sim*, which relies on distributional semantics to judge the plausibility of triples, *LLM_rule* makes decisions through rule-based inference. This fundamental difference in reasoning leads to a distinct error landscape. Since the model does not depend on embedding similarity or lexical proximity, it is less prone to hallucinations or type confusion. However, rule-driven inference introduces its own sources of failure, particularly when the induced rules are incomplete, overly generalized, or applied in contexts that do not reflect their intended scope.

In practice, we observe that *LLM_rule* seldom generates errors caused by entity-type mismatches or superficial association. Instead, its mistakes emerge from the intrinsic limitations of the rule induction and application process. If a correct rule is missing, a noisy triple may be incorrectly retained. If a rule is too permissive, it may validate triples that do not truly conform to its semantics. If a rule fails to account for legitimate exceptions, valid triples may be wrongly rejected.

To capture these distinctive failure modes, Table 5.2 organizes the errors of *LLM_rule* into three categories: *absence of applicable rules*, *overgeneralized rules*, and *exception handling failures*. The following subsections examine how these issues manifest across WN18RR, FB15k-237, and NELL, and how they differ from the error patterns identified in similarity-based reasoning.

5.2.1 Absence of Applicable Rules in *LLM_rule*

The first major source of error in *LLM_rule* arises when no applicable rule covers a given triple. Unlike *LLM_sim*, which defaults to semantic plausibility when making judgments, the rule-based model operates under a closed-world assumption: a triple can only be validated or rejected if it is explicitly covered by an induced rule. When no rule is triggered, the model neither infers alternative constraints nor leverages semantic cues, resulting in false negatives, not because the triple appears correct, but because the system lacks the structural means to challenge it. This limitation often stems from the inherent weakness of LLM prompting or induction algorithms when dealing with sparse or long-tail relations: the model may fail to generalize from a limited number of positive examples and thus cannot distill sufficiently rigor-

Table 5.2: Representative *LLM_rule* Errors Across Datasets

Error Category	**Dataset**	**Example**
Absence of applicable rule	WN18RR	(*ontology.n.01*, _hypernym, *knock_cold.v.01*)
	FB15k-237	(*World_War_II*, /film/film_subject/films, *United_States_of_America*)
	NELL	(*world*, *aquariumincity*, *vegas*)
Overgeneralized Rules	WN18RR	(*annex.n.01*, _derivationally_related_form, *explication.n.02*)
	FB15k-237	(*St._John's*, /music/genre/parent_genre, *World_music*)
	NELL	(*chicken*, *animaleatfood*, *fontina_cheese*)
Exception Handling Failures	WN18RR	(*british_cabinet.n.01*, _derivationally_related_form, *vulnerable.a.01*)
	FB15k-237	(*Nick_Cave*, /people/person/spouse_s./people/marriage/type_of_union, *Seattle_Storm*)
	NELL	(*master*, *hasfamilymember*, *lord_jesus*)

ous and broadly applicable rules. Consequently, valid triples associated with underrepresented relations remain outside the coverage of the induced rule set, exposing a key vulnerability of rule-based refinement under data sparsity conditions.

A representative case in WN18RR is *(ontology.n.01, _hypernym, knock_cold.v.01)*. The relation _hypernym enforces a noun–noun hypernymic hierarchy in the original WordNet schema. However, *LLM_rule* does not learn or apply any rule that restricts _hypernym to same-part-of-speech pairs. Since no constraint forbids linking a noun head to a verb tail, the model leaves the triple unchallenged. The issue is not misinterpretation of the predicate itself, but the absence of any rule encoding the expected linguistic structure.

A similar phenomenon occurs in FB15k-237 with *(World_War_II, /film/film_subject/films, United_States_of_America)*. Since the FB15k-237 dataset is one of the most representative and semantically diverse benchmarks, we present only the rule generated for this dataset as an illustrative example. According to the generated rule set shown in Table 5.3, several constraints were defined, including entity-type validity, one-to-many consistency, temporal coherence, and knowledge-base alignment. However, because the induced rules were derived from limited positive instances, none of these constraints were triggered in this case. The model's closed-world assumption further exacerbates the issue: *LLM_rule* can only validate or reject triples that are covered by its induced rule space. When no applicable rule is activated, it neither infers alternative constraints nor consults semantic cues, leading to the erroneous triple being preserved. This failure illustrates a classic Absence of Applicable Rules error, where incomplete rule coverage and lack of semantic generalization jointly cause false negatives. This case underscores

Table 5.3: Generated Rule Example in FB15k-237 for Relation */film/film_-subject/films*

Generated Rule
Entity Type Constraint: Entity_Type_A must be a film subject to have films. If the head entity is not a film subject (e.g., a genre, actor, or director) but instead an unrelated entity, the triple is likely incorrect. **One-to-Many Relationship Constraint:** A film subject should have only one set of films. If a film subject is assigned multiple conflicting sets of films, such as being both a producer of two movies with different genres, the data may be incorrect. **Temporal Consistency Check:** The films should be historically accurate and consistent with the film subject's career timeline. If a film subject is associated with films that are not consistent with their actual filmography, the information is likely incorrect. **Trusted Knowledge Base Validation:** The films should match trusted knowledge sources, such as IMDb or film databases. If a film subject's films contradict reliable knowledge bases, it should be flagged as potentially incorrect. **Semantic Contradiction Detection:** A film subject's films should not contradict their genre or style. If a film subject is associated with films that are not consistent with their typical genre or style, the data is likely incorrect.

the need for integrating rule-based reasoning with semantic validation mechanisms that can generalize beyond observed instances. Such integration would allow the model to recognize type-inconsistent triples even when no explicit rule has been induced. Although the schema of this relation implicitly expects a subject related to film content, *LLM_rule* only operates over rules it has induced from observed instances. If the extraction process does not capture rules that constrain the head to a film and the tail to a film entity, the model has no mechanism to reject mismatched combinations. As a result, unrelated entities can co-occur under a relation without being flagged.

In NELL, the absence of applicable rules is prevalent due to the openness and variability of relations. The triple *(world, aquariumincity, vegas)* remains unfiltered because no rule specifies that the head of *aquariumincity* must denote an entity of type *Aquarium*. Even when the intended semantics are clear to humans, *LLM_rule* does not infer constraints beyond its induced rule inventory. Without explicit coverage, the model effectively treats the triple as unverifiable rather than incorrect.

5.2.2 Overgeneralized Rules in *LLM_rule*

Overgeneralized rules reflect the opposite issue: a rule is induced, but its scope is too broad to enforce meaningful discrimination. During rule extraction, the system prioritizes high support or frequent co-occurrence patterns, which can lead to rules that capture superficial or incidental regularities

rather than the precise relational structure. This problem is often exacerbated by the LLM's tendency to rely heavily on the statistical frequencies of its training data, causing it to favor rules that generalize across numerous co-occurrence patterns instead of those grounded in genuine logical entailment. As a result, once such an overly inclusive rule is established, any triple matching its loose condition may be incorrectly accepted, even if the entities do not instantiate the intended semantics of the relation.

An example from WN18RR *(annex.n.01, _derivationally_related_form, explication.n.02)* illustrates this phenomenon. Here, the rule derived for _derivationally_related_form may rely on shallow lexical overlap or co-occurrence statistics instead of morphological constraints. As a result, the model treats two unrelated nouns as if they shared a derivational root. The problem is not that the rule is absent, but that its learned condition is insufficiently specific, inappropriately supporting an incorrect triple.

Table 5.4: Generated Rule Example in FB15k-237 for Relation */music/genre/parent_genre*

Generated Rule
Entity Type Constraint: Entity_Type_A must be a music genre and Entity_Type_B must also be a music genre to have a parent-genre relationship. If either Entity_Type_A or Entity_Type_B is not a music genre, the triple is likely incorrect. **One-to-One Relationship Constraint:** Each music genre can have only one parent genre at a given time. If a music genre is assigned multiple conflicting parent genres, the data may be incorrect. **Hierarchy Consistency Check:** The parent genre should be a higher-level music genre than the child genre in a hierarchical structure. If the parent genre is not a higher-level music genre than the child genre, the information is likely incorrect. **Trusted Music Database Validation:** The parent genre should match trusted music databases such as AllMusic or MusicBrainz. If a music genre's parent genre contradicts reliable music databases, it should be flagged as potentially incorrect. **Semantic Contradiction Detection:** A music genre's parent genre should not contradict its stylistic characteristics. If a music genre is labeled with a parent genre that contradicts its known stylistic characteristics, the data is likely incorrect.

A similar pattern occurs in FB15k-237 with triples such as *St._John's, /music/genre/parent_genre, World_music*. According to the rule set shown in Table 5.4, the relation */music/genre/parent_genre* requires both the head and tail entities to denote valid music genres, with the parent genre representing a broader stylistic category. However, due to the overgeneralization of induced rules, *LLM_rule* incorrectly classifies this triple as valid. During rule induction, the system favors rules with high co-occurrence frequency or broad coverage across training instances. As a result, it may induce a rule of the form "if the relation is */music/genre/parent_genre*, then any pair of enti-

ties associated with music or containing musical terms can be linked." Such a rule captures superficial lexical correlations rather than genuine hierarchical semantics. Because "St._John's" occasionally co-occurs with music-related entities (e.g., concerts or choirs) in textual data, it satisfies the loose lexical condition of the overgeneralized rule, even though it is not a music genre. Consequently, the model accepts the triple not because it is semantically valid, but because the induced rule's condition is too permissive to enforce meaningful type constraints. The tail entity "World_music" is indeed a legitimate music genre, but the head entity "St._John's" violates the entity-type constraint defined in the correct relational schema. This mismatch goes undetected because the overgeneralized rule fails to differentiate between entities that merely co-occur in musical contexts and those that truly instantiate hierarchical genre relations. This case illustrates how overgeneralization undermines the discriminative capacity of *LLM_rule*. While the system successfully induces frequent patterns, it loses precision in distinguishing relational semantics. To mitigate such errors, future refinement should incorporate stricter type alignment and semantic filtering criteria during rule induction to avoid validating coincidental associations.

In NELL, overgeneralization often appears in broad relational templates. For instance, *(chicken, animaleatfood, fontina_cheese)* is accepted when the model inductively associates "animal eats food" as a general pattern, without distinguishing plausible or empirically valid instances from accidental or culturally irrelevant ones. Although the relation might apply to many genuine animal–food pairs, the absence of finer-grained constraints allows idiosyncratic matches to slip through.

5.2.3 Exception Handling Failures in *LLM_rule*

A third form of error in *LLM_rule* arises when an induced rule is broadly valid but insufficiently refined to account for exceptions, special cases, or asymmetrical semantic roles. Unlike the absence of applicable rules, where the rule inventory provides no guidance, or overgeneralized rules, where the rule itself is fundamentally too broad and captures superficial or incidental regularities, exception handling failures occur when a rule is partially correct yet fails to capture necessary boundary conditions. Consequently, the model incorrectly accepts triples that violate nuanced semantic or relational constraints (false negatives), or it may wrongly reject valid but infrequent structures (false positives). This error stems not from the rule being too expansive, but from its lack of sensitivity to the edge cases that define the true scope of the relation.

In WN18RR, this problem is reflected in triples such as *(british_cabi-*

net.n.01, _derivationally_related_form, vulnerable.a.01). The model likely induces a rule associating organizational nouns with derived adjectives, a pattern that is broadly valid for many legitimate lexical pairs. However, this rule lacks the necessary constraint refinement to ensure that the two words share etymological roots or conceptual lineage. Unlike an overgeneralized rule, which would accept the triple based on a superficial statistical association, the rule is partially correct. It correctly identifies the Noun → Adjective pattern but fails to incorporate the subtle lexical exceptions required to prevent this specific mismatched derivation. Thus, the error is not due to the rule being fundamentally too expansive (overextension), but rather its insufficient sensitivity to the boundary conditions (edge cases) that define the precise linguistic scope of the relation `_derivationally_related_form`.

Table 5.5: Generated Rule Example in FB15k-237 for Relation */people/person/spouse_s./people/marriage/type_of_union*

Generated Rule
Entity Type Constraint: The head (X) and tail (Y) must both be individuals (people) for them to have a type of union. If either X or Y is not a person, the triple is likely incorrect. **Marriage Consistency Check:** A person should not be married to themselves. If a person is listed as married to themselves, the data is likely incorrect. **Spousal Relationship Integrity:** A person can only have one spouse at a given time with a specific type of union. If a person is listed as having multiple spouses with different types of union, the data may be incorrect. **Historical Accuracy Check:** The type of union should be historically accurate for the marriage. If a person's marriage type contradicts historical records or reliable sources, the information is likely incorrect. **Semantic Contradiction Detection:** A person's marriage type should not contradict their known relationships or biographical information. If a person is listed as married with a type of union that contradicts their known relationships or biographical information, the data is likely incorrect.

A representative example of this error category is the triple *(Nick_Cave, /people/person/spouse_s./people/marriage/type_of_union, Seattle_Storm)*. According to the rule set shown in Table 5.5, the relation */people/person/spouse_s./people/marriage/type_of_union* presupposes that both the head and tail entities denote individuals who can participate in a marital union. However, while this rule structure is broadly valid, it lacks the ability to handle exceptions or detect asymmetrical semantic roles. Although the first rule for the relation explicitly requires both the head (X) and tail (Y) entities to be human individuals, *LLM_rule* still misclassifies the triple as correct. This misjudgment does not stem from the absence of an appropriate rule, but rather from the model's inability to accurately apply it. In prac-

tice, *LLM_rule* relies on co-occurrence and surface-level type cues extracted from training data rather than explicit entity typing from external knowledge bases. Because "Seattle_Storm" frequently appears in human-related contexts (e.g., "Seattle Storm player" or "member of Seattle Storm"), the model erroneously interprets it as an individual entity capable of participating in a marital relation. Consequently, the type constraint rule is formally present but semantically untriggered. The model activates it based on distributional similarity rather than ontological correctness. This leads to an exception handling failure: the rule functions properly for typical cases but fails to recognize boundary conditions where an entity superficially fits the pattern but violates its semantic intent. The error reveals a key limitation of *LLM_rule*: even when correct rules exist, their effectiveness depends on accurate type recognition and exception filtering, which are currently beyond the model's structural inference capabilities. During rule induction, the system correctly learns that entities appearing in this relation are typically of type *person*, but it does not refine the rule to exclude organizational or collective entities that occasionally co-occur in similar relational contexts. The tail entity "Seattle_Storm" refers to a professional basketball team rather than an individual person. Nevertheless, because the induced rule checks only for generic relational patterns, the head has appeared with "spouse_s." relations before, it fails to account for the exception that the tail is not a human entity.

In NELL, the triple *(master, hasfamilymember, lord_jesus)* demonstrates how exception handling breaks down when relations intersect with culturally or contextually nuanced expressions. The underlying rule induced for hasfamilymember is broadly correct as it captures legitimate family-member links. However, it fails to account for exceptions involving metaphorical, religious, or idiomatic uses that fall outside the relation's literal, intended scope. A rule would be considered overgeneralized if it accepted the triple based on superficial statistical correlation (e.g., any common noun followed by any proper noun is a valid link), capturing incidental regularities. In contrast, the failure here is that the partially correct rule is insufficiently refined to enforce the strict semantic boundary conditions, specifically, excluding figurative or context-dependent mentions. In the absence of exception-aware constraints, the model treats this case as a valid relational assertion rather than recognizing it as a violation of the rule's specific semantic intent.

5.3 Summary of Error Analysis

The errors observed in *LLM_sim* and *LLM_rule* do not merely reflect isolated failures, but reveal two fundamentally different mechanisms of reason-

ing and their corresponding limitations. To explicitly clarify the operational differences between the two models, we investigated instances where their judgments diverged.

- **Case 1: *LLM_sim* Correct / *LLM_rule* Incorrect.** This scenario typically arises in sparse regions of the KG (e.g., long-tail relations in s-NELL). Here, *LLM_rule* often fails to induce a valid rule due to insufficient evidence (Absence of Applicable Rule) and thus cannot validate the triple. In contrast, *LLM_sim* successfully leverages the semantic proximity of retrieved context to correctly identify the plausible relationship, demonstrating its superiority in handling implicit or low-resource facts.

- **Case 2: *LLM_rule* Correct / *LLM_sim* Incorrect.** This divergence is prevalent in relations requiring strict ontological constraints (e.g., *hypernym* in WN18RR). *LLM_sim* may be misled by the high distributional similarity between two entities (e.g., distinct but related nouns) and incorrectly flag them as valid. Conversely, *LLM_rule* applies explicit logic – such as "Head must be a superclass of Tail" – to strictly reject the mismatch, showcasing its robustness against superficial semantic associations.

Similarity-based inference is vulnerable to surface correlations, semantic drift, and the absence of type-sensitive grounding, whereas rule-based inference fails when no rule is induced, when rules overgeneralize, or when legitimate exceptions fall outside the learned constraints. Rather than converging to the same error space, the two methods exhibit complementary weaknesses across all three KGs.

This complementarity is especially evident in how each model handles sparsity, ambiguity, and structural variation. *LLM_sim* can capture implicit relational cues that rule-based methods overlook, but it lacks mechanisms to enforce ontological constraints. Conversely, *LLM_rule* can enforce structural consistency where rules exist, but collapses in the absence of applicable rules or exception-aware refinement. These differences explain the performance variations reported in Chapter 4 and highlight why neither paradigm is sufficient in isolation.

More importantly, the analysis suggests actionable directions rather than merely cataloging failure. The identified error modes indicate where type constraints, semantic calibration, or hybrid integration could most effectively improve robustness. In particular, the non-overlapping weaknesses of the two approaches create opportunities for complementary noise detection strategies, confidence-based filtering, and joint reasoning pipelines.

Chapter 6

Conclusion

6.1 Summary

This dissertation has addressed the critical challenge of noise in KGs, which undermines their reliability and diminishes the performance of downstream applications such as KGC. Motivated by the scientific inquiry into LLMs' potential for handling structured knowledge, and the practical need for scalable noise mitigation, we proposed a novel framework that leverages LLMs to detect and refine noisy triples in KGs. The framework integrates two strategies: (*i*) a semantic similarity-based approach (*LLM_sim*), which exploits contextualized representations of triples to identify and correct inconsistencies, and (*ii*) a rule-based reasoning approach (*LLM_rule*), which derives symbolic rules from LLMs to guide more structured refinement. These methods were further enhanced by incorporating contextual augmentation.

Comprehensive experiments were conducted on three widely used benchmark datasets, WN18RR, FB15k-237, and s-NELL, under varying noise conditions. For noise refinement, we introduced two complementary evaluation metrics, namely RR and ACC, to capture both factual restoration and semantic validity. The results demonstrated that our proposed methods consistently achieved high accuracy across datasets. Crucially, we identify ***LLM_-rule*** as the best-performing model overall. It consistently outperformed *LLM_sim* in terms of detection accuracy and refinement validity, particularly in domains characterized by complex factual relations (such as FB15k-237 and s-NELL). This superiority is attributed to its ability to leverage explicit symbolic constraints, which effectively mitigate the risk of LLM hallucinations common in purely semantic approaches. While the RR of *LLM_rule* appeared relatively lower due to its capacity to generate novel, semantically valid triples not present in the gold-standard dataset, its consistently high

accuracy underscored the broader validity of the refinement process. In addition, the ablation studies revealed the effectiveness of the individual modules in our proposed framework.

Beyond refinement quality, we evaluated the downstream task, KGC using the ExpressivE model as the representative framework. The experimental results revealed a strong correlation between noise mitigation and KGC performance, demonstrating that improving the quality of KGs directly enhances predictive accuracy. Our methods significantly outperformed established baselines such as CAGED and GPT-2 XL, confirming the effectiveness of LLM-based refinement. Ablation studies further validated the necessity of both detection and refinement modules. The contextual augmentation proved essential for effective noise detection, while refinement played a crucial role in restoring factual completeness and improving KG reliability.

6.2 Answer for Research Question

As described in Section 1.3, our major research question is: *"How can LLMs be effectively leveraged to detect and refine noise in KGs, while maintaining both accuracy and interpretability?"* This overarching question was further decomposed into three sub-research questions (RQ1, RQ2, and RQ3) to guide the investigation in a systematic manner. We provide answers to these research questions as follows.

RQ1: *How can LLMs be adapted to effectively detect noisy triples in KGs, distinguishing between valid and erroneous facts?*

To address RQ1, we investigated how LLMs can be adapted to reliably detect noisy triples in heterogeneous KGs. We proposed two strategies:

- **Semantic-based detection (*LLM_sim*):** This approach identifies implausible triples by comparing their contextual embeddings with semantically aligned triples retrieved from the KG. Instead of relying solely on plausibility scores, we use similarity metrics between candidate triples and their surrounding graph context. This enables the model to detect errors caused by semantic mismatch, outdated information, or weak relational coherence.

- **Rule-based detection (*LLM_rule*):** This method induces logical consistency rules via LLMs and applies them to detect violations in the KG. Rather than manually designing constraints, the LLM infers domain and range rules, symmetry or inverse dependencies, and other relation-specific patterns. These rules provide a symbolic lens for identifying logically inconsistent triples.

Extensive experiments on WN18RR, FB15k-237, and NELL demonstrate that both strategies outperform embedding-based baselines in accuracy, precision, and recall. The semantic-based method excels in contexts with rich neighborhood information, while the rule-based method captures structural inconsistencies beyond local embeddings. These findings confirm that LLMs can be effectively adapted for noise detection with both contextual and logical reasoning capabilities.

RQ2: *How can LLM be used to improve or refine detected noisy triples to ensure that the corrected knowledge graph maintains high factual accuracy?*

To answer RQ2, we designed a refinement pipeline that enables LLMs to propose accurate corrections. The refinement process varies between the two approaches:

- **Refinement in *LLM_sim*:** Once a triple is flagged as noisy, the model generates multiple candidate triples by replacing either the head or tail entity. These candidates are divided using head–relation and relation–tail groups. We then compute similarity scores between each candidate and reference triples in the KG, and select the candidate with the highest score as the refined replacement.

- **Refinement in *LLM_rule*:** When a triple violates an induced rule, the LLM proposes alternatives that satisfy the constraint. For example, if a triple conflicts with domain or range rules, the head or tail entity is adjusted to restore logical validity. This ensures that the updated triple is both semantically coherent and structurally consistent.

Experimental results confirm that both strategies significantly improve factual accuracy compared to leaving noisy triples unchanged or relying on direct LLM rewriting. Moreover, the refined KGs enhance downstream KGC performance, reflected in higher MRR and Hits@K scores. It demonstrated that the corrections preserve and strengthen global graph consistency.

RQ3: *How can LLM be integrated into the noise detection and improvement process to maintain or enhance explainability and thus enable transparent reasoning about the decisions made?*

To address RQ3, we explicitly designed both approaches to preserve explainability throughout detection and refinement:

- **Interpretability in *LLM_sim*:** Decisions are grounded in explicit similarity scores and retrieved KG context. Users can inspect which reference triples were used for comparison and why a particular candidate was selected, offering transparency beyond black-box plausibility scoring.

- **Interpretability in *LLM_rule*:** Logical rules are human-readable and directly linked to each detection and correction decision. Users can trace which rule a triple violates and how the refinement satisfies that constraint, enabling traceable and auditable reasoning.

Both approaches allow users to visualize noisy triples before and after refinement, inspect propagation effects, and perform manual verification when necessary. Unlike opaque neural detectors, our framework integrates LLMs in a way that enhances interpretability rather than sacrificing it. Furthermore, downstream KGC evaluations show that transparency does not compromise performance. Instead, it supports more reliable error correction and informed knowledge curation.

6.3 Future Work

While this dissertation has demonstrated the effectiveness of *LLM_sim* and *LLM_rule* as independent frameworks, a natural and compelling question arises: why were these two methods not integrated into a unified, hybrid model? This separation was a deliberate methodological choice, deemed necessary to rigorously disentangle and independently quantify the distinct contributions of semantic-based refinement versus symbolic-based refinement. A premature integration would have obscured which mechanism, the semantic grounding of *LLM_sim* or the logical inference of *LLM_rule*, was responsible for performance gains.

Furthermore, our analysis reveals that a simple sequential combination of these methods introduces non-trivial technical challenges, suggesting that a sophisticated hybrid framework is, in itself, a significant future research endeavor.

- **The *LLM_rule* $\rightarrow$ *LLM_sim* Challenge (Context Conflict):** If we first apply *LLM_rule*, it may generate novel, logically correct triples that do not exist in the original KG. However, the *LLM_sim* mechanism is fundamentally dependent on *semantic grounding* against the context of the *original* KG (as detailed in subsection 3.2.2). These newly generated triples, lacking pre-existing semantic anchors, would fail the `LLM_sim` validation, causing the high-precision filter to reject logically sound corrections incorrectly.

- **The *LLM_sim* $\rightarrow$ *LLM_rule* Challenge (Inefficiency and Opportunity Loss):** Conversely, applying *LLM_sim* first introduces two prohibitive flaws.

First, this sequence is computationally inefficient. The two methods have vastly different costs: *LLM_rule* is an efficient logical filter, while *LLM_sim* is computationally expensive, requiring LLM candidate generation and large-scale vector similarity computations. In any sequential pipeline, efficiency is maximized by applying the cheapest, high-recall filter first. Placing the "heavy" *LLM_sim* filter first would force *all* triples to undergo this expensive process, creating an unacceptable bottleneck. The optimal path (R→S) would use the efficient *LLM_rule* to clear the majority of structural errors, passing only the most difficult semantic cases to the expensive *LLM_sim*. Second, this sequence causes irreversible opportunity loss. As demonstrated by its high-precision, lower-recall characteristic (Chapter 4), *LLM_sim* acts as a highly conservative filter. It would prematurely reject or fail to repair many triples for which it lacks strong semantic anchors. This wastes the LLM generation step and permanently discards triples that the efficient, high-recall *LLM_rule* component could have easily and correctly repaired.

Therefore, having established the independent value and complementary error patterns of both approaches, the development of a sophisticated hybrid model that resolves these integration challenges remains a critical and promising direction for future research. The following points outline the key pillars required to build this future framework and otherwise extend this work.

- **Development of a Dynamic Hybrid Framework:** To resolve the integration challenges discussed above, particularly the issue of context conflict, we propose a theoretical "Dynamic Context-Aware Architecture" for future work. Unlike a naive sequential pipeline, this framework introduces a mechanism to update the retrieval context dynamically.

    1. **Stage 1 (Structural Filter via *LLM_rule*):** Utilize the computationally efficient *LLM_rule* as a first-pass filter to eliminate structurally invalid triples (e.g., Entity Type Errors). This addresses the scalability issue by reducing the volume of data requiring expensive semantic analysis.
    2. **Stage 2 (Semantic Validator with Dynamic Context Injection):** Apply *LLM_sim* to the structurally valid candidates. Crucially, the triples validated or generated by Stage 1 are temporarily injected into the retrieval context as provisional ground

truth. This allows *LLM_sim* to validate semantic plausibility conditioned on the updated structural rules, preventing the incorrect rejection of novel, rule-consistent triples.

This proposed integration aims to resolve the trade-off between the high precision of semantic reasoning and the logical consistency of rule-based constraints.

- **Enhancing Scalability for the Hybrid Framework:** To ensure the practical deployment of the frameworks presented in this dissertation, addressing scalability remains fundamental. Both the independent frameworks and the integrated solution require efficient strategies for large-scale KGs. Future work must focus on techniques such as distributed inference or scalable retrieval-augmented prompting to enhance computational efficiency and reduce latency.

- **Cross-Model Validation Strategy:** Currently, our framework utilizes a self-correction mechanism where the same model instance validates its own outputs. A promising direction for future research is to implement a "Cross-Model Verification" strategy. This involves employing a distinct, potentially more capable LLM (e.g., GPT-4 or proprietary models) as an independent judge to evaluate the rules and refinements generated by the smaller backbone model. This approach would help mitigate the risk of self-reinforcing biases and ensure a more robust quality control process.

- **Expansion to Domain-Specific KGs**: We acknowledge that standard benchmarks like WN18RR and FB15k-237 primarily represent general-world knowledge. To address the need for specialized validation, future work will extend this framework to high-stakes vertical domains, such as **Biomedical KGs** (e.g., UMLS) or **Legal KGs**. In these fields, the scarcity of labeled data and the complexity of relational logic make the "expert" reasoning capabilities of LLMs—particularly the rule induction in LLM_rule—even more critical for ensuring data reliability.

- **Improving Robustness against Complex Noise Patterns:** In parallel, improving the robustness and generalization of the noise detection methods presented is an important avenue. In this dissertation, noise was either synthetically injected or identified from curated datasets. However, real-world noise often exhibits more diverse and complex patterns, including subtle semantic shifts, contextual ambiguities, and incomplete evidence. Extending the frameworks to handle

such heterogeneous noise types, possibly through multi-modal integration (e.g., combining text, images, and structured signals), would greatly enhance the practical applicability of LLM-based refinement.

- **Adaptability and Efficiency via PEFT:** To further enhance the adaptability and reusability of the proposed frameworks, future research may explore parameter-efficient fine-tuning (PEFT) techniques. PEFT methods such as LoRA, Prefix Tuning, or Adapter Tuning enable LLMs to adapt to new domains or relation distributions by updating only a small subset of parameters while keeping the base model frozen. In the context of LLM-based KG refinement, PEFT could allow the model to specialize in distinct noise patterns or relation types for `LLM_rule` or adapt to domain-specific semantics for `LLM_sim`, thereby improving their individual transferability and long-term maintainability.

- **Deepening Explainability and Uncertainty Quantification:** While this dissertation has demonstrated that *LLM_rule* improved interpretability, further work is needed across both frameworks. For instance, the semantic reasoning of *LLM_sim* could be enhanced with attention-visualization techniques to explain *why* it found a triple to be semantically distant. Furthermore, developing hybrid approaches that combine *LLM_rule*'s symbolic reasoning with uncertainty quantification could allow users to assess the confidence levels of refinements, which would be valuable in high-stakes domains such as healthcare or finance, where trustworthiness is critical.

- **Validation across Broader Downstream Applications:** Finally, the frameworks proposed herein have primarily been evaluated in the context of downstream KGC. A natural extension is to explore the impact of the refined KGs generated by *LLM_sim* and *LLM_rule* **independently**. This would provide stronger evidence of their **distinct utilities** (e.g., does the high-precision *LLM_sim* graph perform better in factual QA, while the high-recall *LLM_rule* graph performs better in recommendation?). By integrating these refined KGs into tasks such as question answering, dialogue systems, and recommendations, future work can provide stronger evidence of their utility and generalizability.

This dissertation began by addressing the critical problem of KG quality. The separation and analysis of semantic and symbolic refinement methods have provided the necessary foundation for the next generation of robust, knowledge-driven AI. The future development of an integrated framework, as

outlined above, promises a comprehensive and practically deployable solution to this challenge.

Publications

Journal:

- ***Na Dong***, Natthawut Kertkeidkachorn, Xin Liu, Kiyoaki Shirai. *Semantic-Driven and Rule-Based Noise Detection and Refinement in Knowledge Graphs using Large Language Models.* IEICE TRANSACTIONS on Electronics, 2025.
- Chen YA, ***Dong N***. *AI Capabilities and Export Performance: Moderating Effects of Market Development.* Int'l Journal of Emerging Markets, 2024.

Conference:

- ***Na Dong***, Natthawut Kertkeidkachorn, Xin Liu, Kiyoaki Shirai. *Refining Noisy Knowledge Graphs with Large Language Models.* Proceedings of the Workshop on Generative AI and Knowledge Graphs (GenAIK@COLING), 2025.
- ***Na Dong***, Kiyoaki Shirai. *A Study on Subjectivity-oriented Polarity Classification.* 29th Annual Conference of the Association for NLP, 2023.

Bibliography

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] Jumana Alsubhi, Abdulrahman Ahmed Gharawi, and Lakshmish Ramaswamy. Cost-aware tre-nd: Tri-embed noise detection for enhancing data quality of knowledge graph. In *ICAART (3)*, pages 1020–1027, 2024.

[3] Siddhant Arora, Srikanta Bedathur, Maya Ramanath, and Deepak Sharma. Iterefine: Iterative kg refinement embeddings using symbolic knowledge. *arXiv preprint arXiv:2006.04509*, 2020.

[4] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. Dbpedia: A nucleus for a web of open data. In *international semantic web conference*, pages 722–735. Springer, 2007.

[5] Stephen H Bach, Matthias Broecheler, Bert Huang, and Lise Getoor. Hinge-loss markov random fields and probabilistic soft logic. *Journal of Machine Learning Research*, 18(109):1–67, 2017.

[6] Xuefeng Bai, Song He, Yi Li, Yabo Xie, Xin Zhang, Wenli Du, and Jian-Rong Li. Construction of a knowledge graph for framework material enabled by large language models and its application. *npj Computational Materials*, 11(1):51, 2025.

[7] Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on freebase from question-answer pairs. In *Proceedings of EMNLP*, pages 1533–1544, 2013.

[8] Tyler Bikaun, Michael Stewart, and Wei Liu. Cleangraph: Human-in-the-loop knowledge graph refinement and completion. *arXiv preprint arXiv:2405.03932*, 2024.

[9] Olivier Bodenreider. The unified medical language system (umls): integrating biomedical terminology. *Nucleic acids research*, 32(suppl_1):D267–D270, 2004.

[10] 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, 2008.

[11] Rishi Bommasani. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

[12] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 26, 2013.

[13] Chenyang Bu, Xingchen Yu, Yan Hong, and Tingting Jiang. Low-quality error detection for noisy knowledge graphs. *Journal of Database Management (JDM)*, 32(4):48–64, 2021.

[14] Li Cai, Xin Mao, Yuhao Zhou, Zhaoguang Long, Changxu Wu, and Man Lan. A survey on temporal knowledge graph: Representation learning and applications. *arXiv preprint arXiv:2403.04782*, 2024.

[15] Linyue Cai, Chaojia Yu, Yongqi Kang, Yu Fu, Heng Zhang, and Yong Zhao. Practices, opportunities and challenges in the fusion of knowledge graphs and large language models. *Frontiers in Computer Science*, 7:1590632, 2025.

[16] Salvatore Carta, Alessandro Giuliani, Marco Manolo Manca, Leonardo Piano, Livio Pompianu, and Sandro Gabriele Tiddia. Towards knowledge graph refinement: Misdirected triple identification. In *Adjunct Proceedings of the 32nd ACM Conference on User Modeling, Adaptation and Personalization*, pages 460–466, 2024.

[17] Weishuai Che, Zhaowei Liu, Yingjie Wang, and Jinglei Liu. Merit: multi-level graph embedding refinement framework for large-scale graph. *Complex & Intelligent Systems*, 10(1):1303–1318, 2024.

[18] Chen Chen, Yufei Wang, Yang Zhang, Quan Z Sheng, and Kwok-Yan Lam. Separate-and-aggregate: A transformer-based patch refinement model for knowledge graph completion. In *International conference on advanced data mining and applications*, pages 62–77. Springer, 2023.

[19] Zhuo Chen, Yichi Zhang, Yin Fang, Yuxia Geng, Lingbing Guo, Xiang Chen, Qian Li, Wen Zhang, Jiaoyan Chen, Yushan Zhu, et al. Knowledge graphs meet multi-modal learning: A comprehensive survey. *arXiv preprint arXiv:2402.05391*, 2024.

[20] Rajarshi Das, Shehzaad Dhuliawala, Manzil Zaheer, Luke Vilnis, Ishan Durugkar, Akshay Krishnamurthy, Alex Smola, and Andrew McCallum. Go for a walk and arrive at the answer: Reasoning over paths in knowledge bases using reinforcement learning. In *International Conference on Learning Representations*, 2018.

[21] Randall Davis, Howard Shrobe, and Peter Szolovits. What is a knowledge representation? *AI magazine*, 14(1):17–17, 1993.

[22] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. 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 4171–4186, 2019.

[23] Na Dong, Natthawut Kertkeidkachorn, Xin Liu, and Kiyoaki Shirai. Refining noisy knowledge graph with large language models. In *Proceedings of the Workshop on Generative AI and Knowledge Graphs (GenAIK)*, pages 78–86, 2025.

[24] Luis Galárraga, Christina Teflioudi, Katja Hose, and Fabian M Suchanek. Fast rule mining in ontological knowledge bases with amie +. *The VLDB Journal*, 24(6):707–730, 2015.

[25] 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.

[26] Shu Guo, Quan Wang, Lihong Wang, Bin Wang, and Li Guo. Jointly embedding knowledge graphs and logical rules. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 192–202, 2016.

[27] Braden Hancock, Antoine Bordes, Pierre-Emmanuel Mazare, and Jason Weston. Learning from dialogue after deployment: Feed yourself, chatbot! In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3667–3684, 2019.

[28] Shizhu He, Kang Liu, Guoliang Ji, and Jun Zhao. Learning to represent knowledge graphs with gaussian embedding. In *Proceedings of the 24th ACM international on conference on information and knowledge management*, pages 623–632, 2015.

[29] Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard De Melo, Claudio Gutierrez, Sabrina Kirrane, José Emilio Labra Gayo, Roberto Navigli, Sebastian Neumaier, et al. Knowledge graphs. *ACM Computing Surveys (Csur)*, 54(4):1–37, 2021.

[30] Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S Yu. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE transactions on neural networks and learning systems*, 33(2):494–514, 2021.

[31] Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S Yu. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Transactions on Neural Networks and Learning Systems*, 33(2):494–514, 2022.

[32] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM computing surveys*, 55(12):1–38, 2023.

[33] Reagon Karki, Yojana Gadiya, Andrea Zaliani, Bishab Pokharel, Negin Sadat Babaiha, Marek Ostaszewski, Martin Hofmann-Apitius, and Philip Gribbon. Knowledge graph generator (kgg): A fully automated workflow for creating disease-specific knowledge graphs. *Bioinformatics*, page btaf383, 2025.

[34] Kristian Kersting and Luc De Raedt. 1 bayesian logic programming: theory and tool. *Statistical Relational Learning*, page 291, 2007.

[35] Juyeon Kim, Geon Lee, Taeuk Kim, and Kijung Shin. Kgmel: Knowledge graph-enhanced multimodal entity linking. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 3015–3019, 2025.

[36] Ni Lao and William W Cohen. Relational retrieval using a combination of path-constrained random walks. *Machine learning*, 81(1):53–67, 2010.

[37] Chunhua Li, Pengpeng Zhao, Victor S Sheng, Xuefeng Xian, Jian Wu, and Zhiming Cui. Refining automatically extracted knowledge bases

using crowdsourcing. *Computational Intelligence and Neuroscience*, 2017(1):4092135, 2017.

[38] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the AAAI conference on artificial intelligence*, volume 29, 2015.

[39] Bingchen Liu, Yuanyuan Fang, Naixing Xu, Shihao Hou, Xin Li, and Qian Li. Large language models for knowledge graph embedding: A survey. *Mathematics*, 13(14):2244, 2025.

[40] He Lyu, Ningyu Sha, Shuyang Qin, Ming Yan, Yuying Xie, and Rongrong Wang. Advances in neural information processing systems. *Advances in neural information processing systems*, 32, 2019.

[41] Arthur B Markman. *Knowledge representation*. Psychology Press, 2013.

[42] Christian Meilicke, Melisachew Wudage Chekol, Daniel Ruffinelli, and Heiner Stuckenschmidt. Anytime bottom-up rule learning for knowledge graph completion. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 3137–3143, 2019.

[43] Christian Meilicke, Melisachew Wudage Chekol, Daniel Ruffinelli, and Heiner Stuckenschmidt. An introduction to anyburl. In *Joint German/Austrian Conference on Artificial Intelligence (Künstliche Intelligenz)*, pages 244–248. Springer, 2019.

[44] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[45] Deepak Nathani, Jatin Chauhan, Charu Sharma, and Manohar Kaul. Learning attention-based embeddings for relation prediction in knowledge graphs. *arXiv preprint arXiv:1906.01195*, 2019.

[46] Maximilian Nickel, Volker Tresp, Hans-Peter Kriegel, et al. A three-way model for collective learning on multi-relational data. In *Icml*, volume 11, pages 3104482–3104584, 2011.

[47] Pouya Ghiasnezhad Omran, Kewen Wang, and Zhe Wang. Scalable rule learning via learning representation. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 2149–2155, 2018.

[48] Qihong Pan, Limin Yao, Guojiang Shen, Xiao Han, Yichuan Chen, and Xiangjie Kong. Leveraging temporal validity of rules via llms for enhanced temporal knowledge graph reasoning. *Knowledge-Based Systems*, page 114094, 2025.

[49] 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.

[50] Heiko Paulheim. Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic web*, 8(3):489–508, 2016.

[51] Aleksandar Pavlović and Emanuel Sallinger. Expressive: A spatio-functional embedding for knowledge graph completion. *arXiv preprint arXiv:2206.04192*, 2022.

[52] Ciyuan Peng, Feng Xia, Mehdi Naseriparsa, and Francesco Osborne. Knowledge graphs: Opportunities and challenges. *Artificial intelligence review*, 56(11):13071–13102, 2023.

[53] Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. Language models as knowledge bases? 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 2463–2473, 2019.

[54] LM Po. A brief history of llms: From transformers (2017) to deepseek-r1 (2025). `https://medium.com/@lmpo/a-brief-history-of-lmms-from-transformers-2017-to-deepseek-r1-2025-dae75dd3` 2025. Accessed: YYYY-MM-DD.

[55] Sumit Purohit, George Chin, Patrick S Mackey, and Joseph A Cottam. Graphaide: Advanced graph-assisted query and reasoning system. In *2024 IEEE International Conference on Big Data (BigData)*, pages 3485–3493. IEEE, 2024.

[56] Matthew Richardson and Pedro Domingos. Markov logic networks. *Machine learning*, 62(1):107–136, 2006.

[57] Peter N Robinson, Sebastian Köhler, Sebastian Bauer, Dominik Seelow, Denise Horn, and Stefan Mundlos. The human phenotype ontology:

a tool for annotating and analyzing human hereditary disease. *The American Journal of Human Genetics*, 83(5):610–615, 2008.

[58] Tim Rocktäschel, Sameer Singh, and Sebastian Riedel. Injecting logical background knowledge into embeddings for relation extraction. In *Proceedings of the 2015 conference of the north American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1119–1129, 2015.

[59] Ali Sadeghian, Mohammadreza Armandpour, Patrick Ding, and Daisy Zhe Wang. Drum: End-to-end differentiable rule mining on knowledge graphs. *Advances in neural information processing systems*, 32, 2019.

[60] Mohammad Javad Saeedizade, Najmeh Torabian, and Behrouz Minaei-Bidgoli. Kgrefiner: Knowledge graph refinement for improving accuracy of translational link prediction methods. *arXiv preprint arXiv:2106.14233*, 2021.

[61] 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.

[62] Chao Shang, Yun Tang, Jing Huang, Jinbo Bi, Xiaodong He, and Bowen Zhou. End-to-end structure-aware convolutional networks for knowledge base completion. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 3060–3067, 2019.

[63] Fatima Zohra Smaili, Xin Gao, and Robert Hoehndorf. Opa2vec: combining formal and informal content of biomedical ontologies to improve similarity-based prediction. *Bioinformatics*, 35(12):2133–2140, 2019.

[64] Barry Smith, Michael Ashburner, Cornelius Rosse, Jonathan Bard, William Bug, Werner Ceusters, Louis J Goldberg, Karen Eilbeck, Amelia Ireland, Christopher J Mungall, et al. The obo foundry: coordinated evolution of ontologies to support biomedical data integration. *Nature biotechnology*, 25(11):1251–1255, 2007.

[65] Virach Sornlertlamvanich. Enhancing large language models: alleviating knowledge deficiency with external knowledge and semantically aware reasoning (sar) v. sornlertlamvanich. *Knowledge and Information Systems*, pages 1–19, 2025.

[66] Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and policy considerations for modern deep learning research. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 13693–13696, 2020.

[67] Budhitama Subagdja, D Shanthoshigaa, Zhaoxia Wang, and Ah-Hwee Tan. Machine learning for refining knowledge graphs: A survey. *ACM Computing Surveys*, 56(6):1–38, 2024.

[68] Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, pages 697–706, 2007.

[69] Jiaqi Sun, Yujia Zheng, Xinshuai Dong, Haoyue Dai, and Kun Zhang. Type information-assisted self-supervised knowledge graph denoising. *arXiv preprint arXiv:2503.09916*, 2025.

[70] Komal Teru, Etienne Denis, and Will Hamilton. Inductive relation prediction by subgraph reasoning. In *International conference on machine learning*, pages 9448–9457. PMLR, 2020.

[71] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. In *International conference on ML*. PMLR, 2016.

[72] Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, and Partha Talukdar. Composition-based multi-relational graph convolutional networks. *arXiv preprint arXiv:1911.03082*, 2019.

[73] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[74] Denny Vrandečić and Markus Krötzsch. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85, 2014.

[75] Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. Kgat: Knowledge graph attention network for recommendation. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 950–958, 2019.

[76] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the AAAI conference on artificial intelligence*, volume 28, 2014.

[77] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022.

[78] Yanbin Wei, Qiushi Huang, Yu Zhang, and James Kwok. Kicgpt: Large language model with knowledge in context for knowledge graph completion. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 8667–8683, 2023.

[79] Yongkang Xiao and Rui Zhang. Hergc: Heterogeneous experts representation and generative completion for multimodal knowledge graphs. *arXiv preprint arXiv:2506.00826*, 2025.

[80] Ying-jie Xie and Guo-sun Zeng. A new way of search query like knowledge graph and its interpretability. *Knowledge and Information Systems*, 67(3):2745–2770, 2025.

[81] Derong Xu, Xinhang Li, Ziheng Zhang, Zhenxi Lin, Zhihong Zhu, Zhi Zheng, Xian Wu, Xiangyu Zhao, Tong Xu, and Enhong Chen. Harnessing large language models for knowledge graph question answering via adaptive multi-aspect retrieval-augmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 25570–25578, 2025.

[82] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2015.

[83] Fan Yang, Zhilin Yang, and William W Cohen. Differentiable learning of logical rules for knowledge base reasoning. *Advances in neural information processing systems*, 30, 2017.

[84] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Weizhu Ma. Collaborative knowledge base embedding for recommender systems. In *Proceedings of KDD*, pages 353–362, 2016.

[85] Jing Zhang, Bo Chen, Lingxi Zhang, Xirui Ke, and Haipeng Ding. Neural, symbolic and neural-symbolic reasoning on knowledge graphs. *AI Open*, 2:14–35, 2021.

[86] Muhan Zhang and Yixin Chen. Link prediction based on graph neural networks. *Advances in neural information processing systems*, 31, 2018.

[87] 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.

[88] Zhichun Zhang, Ruijie Cai, Xiangnan Zhang, and Lei Wang. Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic Web*, 9(6):889–916, 2018.

[89] Sendong Zhao, Bing Qin, Ting Liu, and Fei Wang. Biomedical knowledge graph refinement with embedding and logic rules. *arXiv preprint arXiv:2012.01031*, 2020.

[90] Xinkui Zhao, Haode Li, Yifan Zhang, Guanjie Cheng, and Yueshen Xu. Trail: Joint inference and refinement of knowledge graphs with large language models. *arXiv e-prints*, pages arXiv–2508, 2025.

[91] Lingfeng Zhong, Jia Wu, Qian Li, Hao Peng, and Xindong Wu. A comprehensive survey on automatic knowledge graph construction. *ACM Computing Surveys*, 56(4):1–62, 2023.

[92] Zhaocheng Zhu, Zuobai Zhang, Louis-Pascal Xhonneux, and Jian Tang. Neural bellman-ford networks: A general graph neural network framework for link prediction. *Advances in neural information processing systems*, 34:29476–29490, 2021.