LLMs as Knowledge Graph Refiners: Mitigating Factual Inconsistencies in Generative Knowledge Extraction

Donghyun Kim[1] **Hyeongjun Yang**[1] **Seokju Hwang**[1] **Kyong-Ho Lee**[1]* **Chanhee Lee**[2]
[1]Department of Computer Science, Yonsei University
[2]Samsung Securities
{dhkim92,edbm95,hsjtjrwn,khlee89}@yonsei.ac.kr
chanhee88.lee@samsung.com

Abstract

Knowledge graphs (KGs) provide a structured representation of real-world facts as triples consisting of entities and their relationships. With the rapid progress of large language models (LLMs), recent studies increasingly explore LLMs for end-to-end KG construction from text. In particular, generative knowledge extraction (GKE) builds KGs by directly generating structured triples from documents. However, generation errors are inevitable, and the resulting KGs often contain triples that do not align with the facts expressed in the source text. To address these issues, we propose GraphRefine, a framework that performs triple-level refinement on KGs constructed via GKE. We first analyze factual inconsistencies that arise in GKE and categorize their types based on a human evaluation. We then construct training data reflecting these types and fine-tune an LLM as a KG refiner. Given a draft KG, the fine-tuned refiner selects a refinement operation for each triple and, if needed, deletes, edits, or rewrites it to reduce factual inconsistencies. Extensive experiments demonstrate that GraphRefine goes beyond deletion-only approaches and improves KG quality from diverse perspectives.

1 Introduction

Knowledge graphs (KGs) represent real-world entities and their relations in a structured form, serving as a key resource for knowledge-driven downstream tasks such as question answering, recommendation, and fact-checking (Huang et al., 2019; Yang et al., 2022; Kim et al., 2023). Traditionally, KGs have been constructed through information extraction pipelines (Hogan et al., 2021). More recently, advances in large language models (LLMs) have spurred increasing efforts to automate end-to-end KG construction using LLMs (Han et al., 2024; Chen et al., 2024; Zhang and Soh, 2024; Niu et al., 2025; Lu and Wang, 2025; Mo et al., 2025). Among these LLM-driven approaches, a representative paradigm is generative knowledge extraction (GKE), which constructs a KG by directly generating structured triples from natural language documents, rather than progressively identifying entities and relations through a multi-stage pipeline (Zhang et al., 2025). This paradigm offers strong flexibility and scalability, substantially reducing the cost of KG construction across diverse scenarios.

However, errors during generation are inevitable, and consequently, the resulting KGs often contain triples that do not align with the real-world facts expressed in the source documents (Wang et al., 2021; Xue and Zou, 2022). In this work, we refer to such cases as **factual inconsistencies** in KGs. Existing studies on KG refinement and validation have largely focused on identifying and filtering incorrect triples using rule-based heuristics, schema constraints, or KG embedding methods (Paulheim, 2016; Huaman et al., 2020). However, KGs constructed via GKE exhibit a broader and often amplified range of errors, stemming from the generative nature of LLM-based extraction (Zhang et al., 2024; Kamoi et al., 2024). These errors include entity recognition failures, relational semantic distortions, unsupported hallucinations, and representation inconsistencies that reduce the clarity and usability of extracted triples. Together, these error patterns differ fundamentally from the assumptions of prior refinement methods, making them difficult to be fully addressed in practice. Accordingly, mitigating these inconsistencies requires careful analysis of their characteristics and underlying causes to facilitate effective refinement for GKE.

In this context, to improve the correctness of LLM-based KG construction, GraphJudge (Huang et al., 2025) employs an LLM as a judge to assess the validity of triples and delete incorrect ones. While this approach can eliminate some factual inconsistencies in GKE, such as unsupported triples, refinement by deletion has fundamental limitations.

*Corresponding Author

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 29358–29378

First, deleting triples may discard partially correct facts that could otherwise be corrected through targeted edits, thereby unnecessarily reducing KG coverage. Second, such inconsistencies in GKE are not always captured by a binary judgment of whether a triple is valid or invalid. In particular, beyond correctness-level errors, GKE also produces representation-level inconsistencies, such as imprecise entity spans, that degrade triple clarity and usability without necessarily making them incorrect. Therefore, correctness-only filtering is insufficient to improve overall KG quality, motivating refinement methods that edit and normalize triples rather than simply removing them.

In this paper, we propose **GraphRefine**, a post-hoc refinement framework for KGs constructed via GKE. Given a draft KG and its source document, GraphRefine selects a document-grounded refinement operation for each triple and edits or rewrites it when needed to mitigate factual inconsistencies. To this end, we first analyze and systematically define factual inconsistency types that frequently arise in GKE settings, and then train an LLM as a KG refiner to perform triple-level refinement operations. In particular, we fine-tune the refiner on training data constructed to cover diverse inconsistency types, enabling it to learn a principled refinement strategy that goes beyond simple filtering and supports deletion, editing, and rewriting. We also evaluate KG quality beyond correctness alone using multi-faceted metrics such as GenRES (Jiang et al., 2024), measuring not only factual accuracy but also representation quality. Our experiments show that GraphRefine improves KG quality across different dimensions by reducing factual inconsistencies and enhancing triple clarity and consistency, rather than merely removing incorrect triples.

Our main contributions are as follows:

- We analyze and systematically define factual inconsistency types that arise in GKE, thereby formalizing the refinement problem for LLM-based KG construction.
- We propose GraphRefine, a post-hoc refinement method for KGs constructed via GKE, which fine-tunes an LLM as a KG refiner to perform triple-level refinement operations.
- We conduct multi-faceted evaluations under diverse metrics and demonstrate GraphRefine's effectiveness through extensive experiments across datasets and base extractors.

2 Background & Related Work

KG construction involves identifying entities and relations from raw text and representing them as structured triples. Traditional approaches have typically relied on modular pipelines, which decompose this process into subtasks such as named entity recognition, relation extraction, and event extraction (Hogan et al., 2021). Recent progress in LLMs has introduced a generative paradigm that replaces such procedural pipelines with end-to-end generation (Wan et al., 2023; Wang et al., 2025). In particular, generative knowledge extraction (GKE) leverages the contextual reasoning and language generation capabilities of LLMs to directly generate structured knowledge from natural language, enabling a more holistic and flexible approach to KG construction that addresses the limitations of conventional methods (Zhang et al., 2025).

Building on this generative paradigm, a growing body of work has proposed advanced frameworks for KG construction. SAC-KG (Chen et al., 2024) adopts an LLM-driven framework for automated domain-specific KG construction. Its generator retrieves relevant information from domain corpora and DBpedia to compose model inputs, while the verifier performs rule-based correction and the pruner removes irrelevant entities using a classification model. EDC (Zhang and Soh, 2024) proposes a framework that decomposes KG construction into extraction, definition, and canonicalization to reduce schema dependence and redundancy. This approach utilizes LLM-generated definitions and similarity-based canonicalization to construct structured knowledge without predefined ontologies. Tree-KG (Niu et al., 2025) constructs an explicit tree-structured KG by leveraging the hierarchical organization of knowledge-intensive documents. It expands this structure using LLM-based operators, enabling efficient integration and expansion of knowledge while preserving structural consistency. KGGen (Mo et al., 2025) utilizes an LLM-driven pipeline that performs entity and relation extraction, integration, and deduplication to construct a KG from plain text. It combines embedding clustering with LLM-based normalization to minimize semantic redundancy and enhance the density and coherence of the resulting KG.

However, GKE remains prone to errors, including incorrect entity recognition, relation distortion, and hallucination. These errors often lead to factually inconsistent triples that degrade the quality

of the constructed KG while also being difficult to fix during extraction, making post-hoc validation and refinement essential. GraphJudge (Huang et al., 2025) filters out noisy or hallucinated triples by employing an LLM as a graph judge to assess the correctness of extracted triples. Although such methods can improve accuracy, deletion-only filtering may ultimately reduce KG coverage. Moreover, factual inconsistencies may go beyond correctness-level errors, limiting the effectiveness of filtering-based approaches. To address these limitations, we employ an LLM as a KG refiner to mitigate diverse factual inconsistencies arising in GKE.

3 Problem Definition

We first describe generative knowledge extraction (GKE) for LLM-based KG construction and then introduce the KG refinement task.

3.1 Generative Knowledge Extraction

We define GKE as the task of extracting a KG $\mathcal{G}_\mathcal{D}$ from a source document $\mathcal{D}$ with an LLM, where the KG consists of structured triples $\tau = (h, r, t)$. The constructed $\mathcal{G}_\mathcal{D}$ is defined as:

$$\mathcal{G}_\mathcal{D} = \{(h, r, t) \mid h, t \in \mathcal{E},\ r \in \mathcal{R}\}, \quad (1)$$

where $\mathcal{E}$ denotes the set of entities and $\mathcal{R}$ denotes the set of relations, respectively. During the extraction process, an LLM generates an output sequence conditioned on the source document $\mathcal{D}$. This process can be modeled autoregressively as:

$$P(x_i \mid x_1, x_2, ..., x_{<i}, \mathcal{D}), \quad (2)$$

where x_i represents the i-th token of the output sequence. From the generated output, a set of triples $[\tau_1, \tau_2, ...]$ is derived, yielding $\mathcal{G}_\mathcal{D}$. The objective of this task is to extract a KG that covers as much relevant factual information as possible while remaining consistent with the facts contained in the source document.

3.2 Knowledge Graph Refinement

The KG $\mathcal{G}_\mathcal{D}$ constructed through GKE may contain factual inconsistencies. The goal of KG refinement is to transform $\mathcal{G}_\mathcal{D}$ into a refined graph $\hat{\mathcal{G}}_\mathcal{D}$ that is more faithful to the underlying document $\mathcal{D}$. To formalize this process, we introduce an LLM-based refinement operator $\mathcal{F}_\theta$ as follows:

$$\hat{\mathcal{G}}_\mathcal{D} = \mathcal{F}_\theta(\mathcal{D}, \mathcal{G}_\mathcal{D}). \quad (3)$$

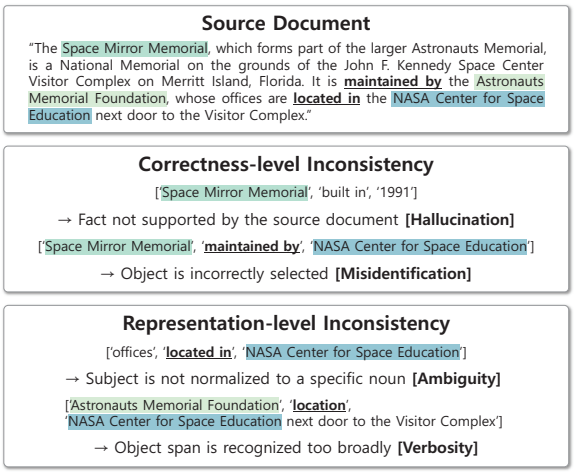


Figure 1: Representative examples of factual inconsistencies observed in generative knowledge extraction.

Here, $\mathcal{F}_\theta$ takes the source document $\mathcal{D}$ and the initial graph $\mathcal{G}_\mathcal{D}$ as inputs and enhances the factual consistency of the KG by identifying and correcting errors. This refinement mitigates factual inconsistencies introduced during GKE, yielding a KG that more faithfully reflects the original content of the source document.

4 Method

In this section, we propose GraphRefine for LLM-based KG refinement. We first categorize factual inconsistencies in GKE, and then describe a strategy for fine-tuning an LLM as a KG refiner.

4.1 Factual Inconsistencies in GKE

By analyzing KGs constructed by GKE, we identify two prominent types of factual inconsistencies: (i) correctness-level and (ii) representation-level. Moreover, these inconsistencies are observed across datasets and extraction models.[1] In the following, we describe each inconsistency type and discuss how it can be addressed in the context of KG refinement.

Correctness-level Inconsistencies. Correctness-level inconsistencies occur when individual triples are factually incorrect. As illustrated in Figure 1, this category includes hallucinated triples that are not supported by the source document, as well as cases where the meanings of entities or relations are distorted, leading to incorrect factual statements. Such inconsistencies directly undermine the factual accuracy of the constructed KG and mislead

[1] Detailed statistics and analyses of factual inconsistencies are provided in Appendix D.

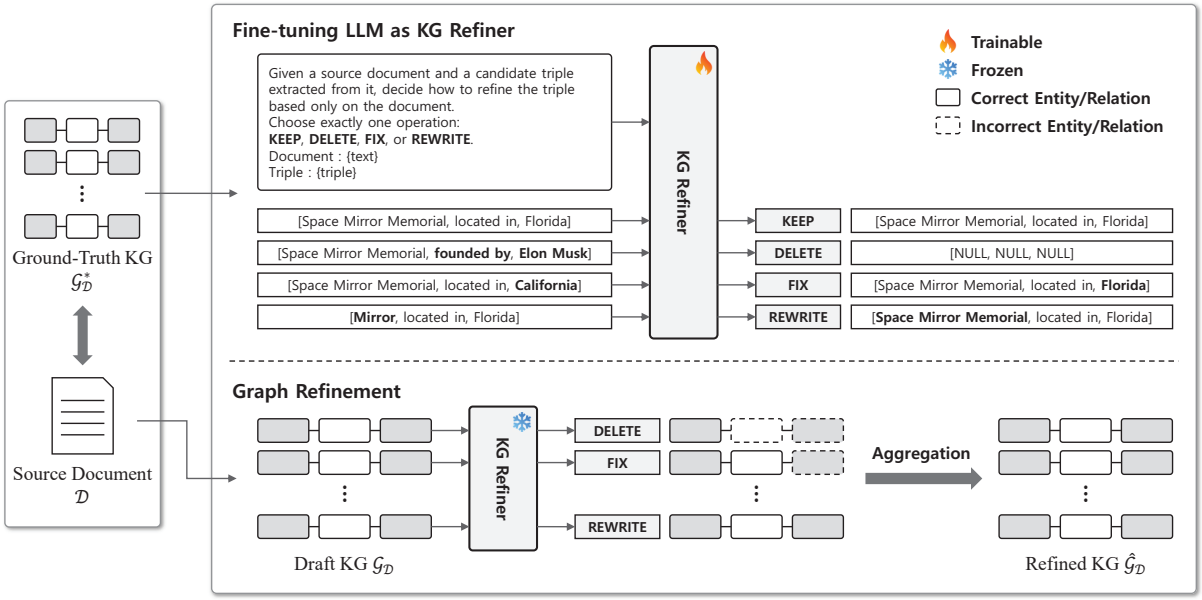


Figure 2: Overview of the proposed GraphRefine framework. We synthesize operation-specific supervision to model factual inconsistency types arising in GKE and fine-tune an LLM as a KG refiner. The refiner then refines a draft KG at the triple level, and the outputs are aggregated to construct the final refined KG. This design enables GraphRefine to be model-agnostic and applied post-hoc to draft KGs produced by diverse GKE systems.

downstream reasoning and applications. They are often removed, but this can reduce the coverage of the resulting KG. When supporting evidence exists in the source document, rewriting a triple to correct its semantics is often more effective than simply removing it, as it preserves information coverage while improving factual accuracy.

Representation-level Inconsistencies. This type of inconsistency encompasses cases where triples are not strictly incorrect but nonetheless degrade KG quality due to inconsistent or imprecise representations. Figure 1 provides representative examples, such as ambiguous span boundaries and unnecessarily verbose expressions. As a result, the KG may fail to express the facts in the source document precisely and unambiguously, introducing factual inconsistencies at the representation level even when the underlying facts are correct. Therefore, we mitigate these representation-level inconsistencies via meaning-preserving rewriting, reducing ambiguity and surface-form variation while enhancing clarity and consistency.

4.2 Fine-tuning LLM as KG Refiner

To effectively mitigate the factual inconsistencies defined above, we fine-tune an LLM as a KG refiner. The KG refiner takes the source document and an extracted KG triple as input, and applies an appropriate refinement operation to each triple. Specifically, the KG refiner is designed to select one of the following operations for each triple:

- **KEEP**: If the triple is factually consistent with the source document and its representation is appropriate, it is retained as is.
- **DELETE**: If the triple is factually incorrect or not supported by the source document, it is removed to reduce noise.
- **FIX**: If supporting evidence exists but the meaning of the triple is incorrect, its semantics are corrected to match the evidence.
- **REWRITE**: If the meaning is preserved but the representation is inconsistent or suboptimal, the triple is rewritten.

4.2.1 Dataset Generation

We construct synthetic supervision for fine-tuning the KG refiner by defining operation-specific sets of triples. Let $\mathcal{S}_{\mathcal{G}_{\text{train}}}$ denote the set of ground-truth KGs in the training split, where each $\mathcal{G} \in \mathcal{S}_{\mathcal{G}_{\text{train}}}$ is a set of triples $\tau = (h, r, t)$ paired with a source document $\mathcal{D}$, and let $\mathcal{E}(\mathcal{D})$ denote the set of entities mentioned in $\mathcal{D}$, with $\mathcal{E}$ and $\mathcal{R}$ denoting the global sets of entities and relations, respectively.

KEEP. We retain ground-truth triples as is:

$$\mathcal{T}_{\text{KEEP}} = \bigcup_{\mathcal{G} \in \mathcal{S}_{\mathcal{G}_{\text{train}}}} \Big\{ (h, r, t) \;\Big|\; (h, r, t) \in \mathcal{G} \Big\}. \quad (4)$$

DELETE. We generate triples intended for removal by jointly corrupting the relation and tail, encouraging non-recoverable hallucinations. Each triple is further required to satisfy $\tau_{\texttt{DEL}} \notin \mathcal{G}$.

$$\mathcal{T}_{\texttt{DELETE}} = \bigcup_{\mathcal{G} \in \mathcal{S}_{\mathcal{G}_{\text{train}}}} \Big\{ \tau_{\texttt{DEL}} = (h, \tilde{r}, \tilde{t}) \;\Big|\; \tilde{r} \in \mathcal{R} \setminus \{r\},\ \tilde{t} \in \mathcal{E} \setminus \{t\} \Big\}. \tag{5}$$

FIX. We generate erroneous triples by substituting the tail entity with a different entity that does not appear in the source document. We further require that each generated triple satisfies $\tau_{\texttt{FIX}} \notin \mathcal{G}$. The same corruption is applied to the head entity in an analogous manner.

$$\mathcal{T}_{\texttt{FIX}} = \bigcup_{\mathcal{G} \in \mathcal{S}_{\mathcal{G}_{\text{train}}}} \Big\{ \tau_{\texttt{FIX}} = (h, r, \tilde{t}) \;\Big|\; \tilde{t} \notin \mathcal{E}(\mathcal{D}) \Big\}. \tag{6}$$

REWRITE. We generate representation-level inconsistencies by perturbing the entity mention span in the source document while preserving its underlying semantics. Let $\text{span}(e, \mathcal{D}) = (p, q)$ denote the word-level span of entity e in $\mathcal{D}$, where p and q are the start and end positions, respectively. We then obtain a perturbed span (p', q') as follows:

$$\begin{aligned} \text{Expansion:} \quad & (p', q') = (p - \delta_l,\ q + \delta_r), \\ \text{Shrinkage:} \quad & (p', q') = (p + \delta_l,\ q - \delta_r), \end{aligned} \tag{7}$$

where δ_l and δ_r are integers controlling the left and right boundary shifts, respectively. We then define the perturbed surface form as $\tilde{e} = \mathcal{D}[p' : q']$. In the following, we describe the tail-entity perturbation; the same procedure is applied to the head entity analogously.

$$\mathcal{T}_{\texttt{REWRITE}} = \bigcup_{\mathcal{G} \in \mathcal{S}_{\mathcal{G}_{\text{train}}}} \Big\{ \tau_{\texttt{REW}} = (h, r, \tilde{t}) \;\Big|\; \tilde{t} = D_i[p'_t : q'_t] \Big\}. \tag{8}$$

4.2.2 Training Objective

The synthetic training pool is defined as the union of the operation-specific triple sets:

$$\mathcal{T}_{\text{train}} = \mathcal{T}_{\texttt{KEEP}} \cup \mathcal{T}_{\texttt{DELETE}} \cup \mathcal{T}_{\texttt{FIX}} \cup \mathcal{T}_{\texttt{REWRITE}}. \tag{9}$$

We convert $\mathcal{T}_{\text{train}}$ into instruction-following supervision and fine-tune the KG refiner. As illustrated in Figure 2, each training instance is formatted as a single prompt by concatenating the source document $\mathcal{D}$, an erroneous input triple $\tilde{\tau}$ generated by the corruption process, and an instruction $\mathcal{I}$ that asks the model to refine the triple. The target response is defined to contain an operation label $y \in \{\texttt{KEEP}, \texttt{DELETE}, \texttt{FIX}, \texttt{REWRITE}\}$ along with a target triple τ^* that corresponds to the ground-truth refinement result.[2] We represent each instance as token sequences, denoting the prompt as $\mathcal{X}_{\text{in}}$ and the target response as $\mathcal{X}_{\text{out}}$. Then, we optimize the model parameters with teacher-forced supervised fine-tuning, maximizing $p_\theta(\mathcal{X}_{\text{out}} \mid \mathcal{X}_{\text{in}})$. The training objective is the standard negative log-likelihood over the output tokens:

$$\mathcal{L}_{\text{SFT}} = -\sum_{i=1}^{L} \log p_\theta\big(x_i^{\text{out}} \mid \mathcal{X}_{\text{in}},\ x_{<i}^{\text{out}}\big), \tag{10}$$

where θ denotes the learnable parameters and L is the output length. With this instruction-tuning objective, the model is jointly trained to select an appropriate refinement operation and generate the corresponding refined triple.

4.3 Graph Refinement

Given a draft KG $\mathcal{G}_{\mathcal{D}}$ from GKE and a source document $\mathcal{D}$, a KG refiner assigns an operation label to each triple $\tau \in \mathcal{G}_{\mathcal{D}}$ and generates a refined triple $\hat{\tau}$ when applicable. Formally, the refiner defines a mapping as follows:

$$(y, \hat{\tau}) = f_\theta(\mathcal{D}, \tau), \tag{11}$$

where f_θ denotes the fine-tuned refiner. It then constructs the refined KG $\hat{\mathcal{G}}_{\mathcal{D}}$ by accumulating triples according to the predicted operation:

$$\hat{\mathcal{G}}_{\mathcal{D}} = \bigcup_{\tau \in \mathcal{G}_{\mathcal{D}}} g(\tau; y, \hat{\tau}). \tag{12}$$

The aggregation function $g(\tau; y, \hat{\tau})$ is defined as:

$$g(\tau; y, \hat{\tau}) = \begin{cases} \{\tau\}, & \text{if } y = \texttt{KEEP}, \\ \emptyset, & \text{if } y = \texttt{DELETE}, \\ \{\hat{\tau}\}, & \text{if } y \in \{\texttt{FIX}, \texttt{REWRITE}\}. \end{cases} \tag{13}$$

Finally, the refined graph $\hat{\mathcal{G}}_{\mathcal{D}}$ is obtained, yielding a KG that is more faithful to the source document $\mathcal{D}$ by reducing factual inconsistencies. GraphRefine is model-agnostic to the underlying GKE system and can thus refine KGs produced by diverse extractors. This property allows it to be seamlessly integrated into existing LLM-based KG construction pipelines as a general refinement module, without modifying the upstream extractor.

[2] Detailed instruction prompts are provided in Appendix F.

5 Experimental Setup

5.1 Datasets

In this study, we utilize KG–text aligned datasets that provide paired textual contexts and their corresponding structured knowledge. We employ **GenWiki** (Jin et al., 2020) and **DocRED** (Yao et al., 2019) as general-domain datasets, and **SciERC** (Luan et al., 2018) and **CDR** (Li et al., 2016) as domain-specific datasets, to comprehensively evaluate the methods across both broad and specialized domains. Detailed statistics and descriptions of the datasets are provided in Appendix A.

5.2 Evaluation Metrics

Conventional evaluation metrics rely on rule-based exact string matching. However, such approaches fail to account for semantic variation, limiting their ability to reliably assess KGs produced by GKE. Moreover, surface-level matching alone cannot capture broader quality dimensions of extracted KG triples, underscoring the need for a more comprehensive evaluation. Therefore, we adopt soft matching and semantic-level metrics such as **G-BLEU** (BL), **G-ROUGE** (RO), and **G-BERTScore** (BS) (Huang et al., 2025), and report their precision, recall, and macro F1-score. In addition, we employ the multi-dimensional evaluation suite GenRES (Jiang et al., 2024), which measures **Topical Similarity** (TS), **Uniqueness** (US), **Factualness** (FS), **Granularity** (GS), and **Completeness** (CS) of the extracted KG triples. Detailed descriptions of these evaluation metrics are provided in Appendix B.

5.3 Baselines

In our experiments, we consider seven baselines, including four vanilla LLMs and three LLM-based KG construction methods. **Llama-2-7B**, **Llama-2-13B** (Touvron et al., 2023): These open-source LLMs are used as baselines to analyze the impact of model scale on performance. **GPT-4o-mini**, **GPT-4o** (Hurst et al., 2024): These closed-source LLMs represent efficiency-oriented and high-performance models, respectively. The prompts used for these LLMs are provided in Appendix F. **EDC** (Zhang and Soh, 2024), **KGGen** (Mo et al., 2025): We follow the default settings of their official implementations, using GPT-4o-mini as the backbone LLM. **GraphJudge** (Huang et al., 2025): We follow the default settings of GraphJudge, using GPT-4o-mini for KG construction and Llama-2-7B as the judge model to evaluate extracted triples.

5.4 Implementation Details

Knowledge Extraction. GraphRefine is model-agnostic and can be applied post-hoc to draft KGs from diverse GKE systems. For fair comparison, we generate draft KGs using GraphJudge's Entity-Centric Text Denoising module (Huang et al., 2025) with GPT-4o-mini as the backbone.

Dataset Generation. Given the ground-truth KG and the source document, we synthesize one example per inconsistency type for each instance. For $\mathcal{T}_{\texttt{FIX}}$, we corrupt the target triple by randomly selecting either the head or tail entity. For $\mathcal{T}_{\texttt{REWRITE}}$, we perturb the mention span of a randomly chosen head or tail entity in the source document by expansion or shrinkage, where the word-level span offset δ is sampled from 5 to 10, and only if the entity mention appears in the document text.

LLM Fine-tuning. We utilize Llama-2-7B (Touvron et al., 2023) as the backbone of the KG refiner and fine-tune it with LoRA (Hu et al., 2021) for parameter-efficient training. Detailed hyperparameter settings are provided in Appendix C.

6 Experimental Results

In this section, we systematically evaluate the effectiveness of GraphRefine by focusing on the following research questions. **RQ1.** How effectively does GraphRefine mitigate factual inconsistencies arising in GKE and improve overall KG quality? **RQ2.** How broadly can GraphRefine be applied as a post-hoc refiner to KGs from diverse GKE models? **RQ3.** How well does GraphRefine generalize when applied across different datasets?

6.1 Overall Performance Comparison (RQ1)

Table 1 demonstrates that GraphRefine mitigates factual inconsistencies arising in GKE while improving KG quality from multiple perspectives. In terms of the G-Score (BL/RO/BS) measured by F1, GraphRefine consistently outperforms existing KG construction methods and GraphJudge, a filtering-based post-processing baseline, across all datasets. Directly compared with GraphJudge, GraphRefine improves precision while preserving recall, demonstrating the advantage of operation-based refinement over deletion-only filtering. Furthermore, under the multi-faceted GenRES metrics, GraphRefine maintains high factualness (FS) while substantially enhancing granularity (GS) and uniqueness (US). This suggests that refinement not only

Dataset	Method	BL_P	BL_R	BL_{F1}	RO_P	RO_R	RO_{F1}	BS_P	BS_R	BS_{F1}	TS	US	FS	GS	CS
GenWiki	Llama-2-7B	60.57	37.61	44.46	52.76	32.90	38.83	83.96	51.94	61.41	49.01	66.82	86.59	83.53	60.51
	Llama-2-13B	63.91	56.95	58.31	55.46	49.56	50.66	88.06	78.48	80.41	54.63	64.74	92.83	92.59	74.97
	GPT-4o-mini	64.39	62.22	62.02	55.57	53.91	53.62	87.65	84.68	84.43	**71.76**	67.91	**98.74**	95.22	77.36
	GPT-4o	64.78	63.51	62.80	56.00	54.97	54.30	87.04	85.32	84.38	68.56	66.60	98.40	95.03	80.65
	EDC	63.62	64.61	62.92	56.80	57.96	56.30	85.76	87.04	84.79	64.00	62.16	94.75	94.58	81.70
	KGGen	57.63	56.79	55.88	47.81	47.39	46.48	85.37	83.67	82.56	42.32	49.25	78.29	98.58	60.34
	GraphJudge	68.69	62.61	63.40	59.41	54.27	54.86	88.26	80.61	81.61	33.81	65.97	97.92	98.04	75.66
	GraphRefine	69.14	65.38	**65.23**	60.73	56.60	**56.41**	89.64	86.05	**85.32**	65.24	**72.68**	98.36	**98.67**	**85.53**
DocRED	Llama-2-7B	49.28	23.32	26.83	37.92	17.50	20.33	80.82	38.35	44.03	13.43	74.41	85.77	71.84	22.58
	Llama-2-13B	45.52	41.49	37.41	33.27	30.01	27.05	73.27	66.28	60.18	19.76	76.10	85.45	80.45	33.04
	GPT-4o-mini	34.72	58.33	39.98	25.19	43.24	29.20	54.55	89.48	62.35	**46.70**	88.34	96.25	82.76	45.72
	GPT-4o	35.24	59.34	40.71	25.87	44.41	30.06	54.44	89.74	62.48	43.95	86.91	**96.63**	84.57	48.79
	EDC	34.03	59.62	39.88	25.06	45.02	29.61	52.82	90.24	61.37	37.75	87.47	90.61	81.78	50.78
	KGGen	41.13	49.84	41.07	27.62	33.79	27.62	67.75	81.58	67.62	19.99	71.86	69.66	92.24	32.42
	GraphJudge	47.87	56.20	47.57	36.09	42.57	35.86	70.46	82.41	70.06	18.83	80.25	95.24	96.02	54.74
	GraphRefine	50.41	60.26	**52.94**	37.28	44.60	**38.29**	71.23	85.34	**73.84**	36.72	**89.46**	96.38	**97.10**	**59.63**
SciERC	Llama-2-7B	64.77	26.77	34.65	58.65	24.71	31.61	89.79	38.98	49.24	58.29	50.88	78.77	73.60	37.10
	Llama-2-13B	64.36	28.79	36.37	55.92	24.62	31.24	91.07	41.94	52.14	63.69	52.18	75.95	76.08	29.75
	GPT-4o-mini	55.68	58.92	54.02	48.17	51.49	46.70	76.83	82.13	74.82	85.03	93.18	97.14	72.68	57.64
	GPT-4o	55.49	59.23	54.10	48.11	52.25	47.01	76.87	82.51	75.10	**85.90**	92.34	**98.22**	69.99	60.42
	EDC	54.36	62.91	55.08	47.59	55.47	**48.22**	73.83	85.77	74.91	75.20	91.96	93.46	69.80	**64.64**
	KGGen	53.92	56.62	52.33	46.22	48.84	44.82	77.32	82.08	75.24	80.93	90.14	94.05	81.93	44.71
	GraphJudge	59.26	53.69	52.91	52.43	47.11	46.51	82.13	74.84	73.47	74.38	87.28	97.47	83.98	49.74
	GraphRefine	57.85	58.13	**55.43**	51.15	50.32	47.76	80.64	77.34	**76.29**	79.61	**94.42**	97.31	**86.33**	56.75
CDR	Llama-2-7B	19.88	8.05	9.02	15.17	6.01	6.77	43.28	17.58	20.06	33.73	89.39	42.86	72.75	6.61
	Llama-2-13B	34.26	23.07	22.01	25.39	16.95	16.12	69.59	45.40	44.34	59.86	72.55	81.78	60.14	18.42
	GPT-4o-mini	30.51	54.14	34.88	21.04	38.38	24.22	50.76	87.58	57.59	**82.86**	91.55	96.33	74.76	32.50
	GPT-4o	32.94	53.67	36.22	23.21	38.70	25.67	53.93	85.53	58.87	79.47	87.80	**96.80**	76.28	35.03
	EDC	29.35	54.24	33.84	20.38	38.86	23.70	48.91	87.62	55.88	78.21	**92.22**	92.45	68.00	35.68
	KGGen	37.49	50.98	38.38	25.40	35.40	26.18	61.57	81.93	62.68	67.82	82.66	86.58	88.69	29.33
	GraphJudge	38.78	53.84	40.16	27.62	39.13	28.74	61.03	82.96	62.93	61.33	84.62	93.36	91.17	35.17
	GraphRefine	38.12	55.36	**42.73**	28.49	40.71	**31.26**	61.24	84.22	**64.10**	66.54	92.15	94.21	**93.40**	**36.54**

Table 1: Overall performance comparison on GenWiki, DocRED, SciERC, and CDR. We report BL/RO/BS (P/R/F1) as KG quality metrics, and multi-faceted GenRES metrics (TS/US/FS/GS/CS) to assess improvement from multiple perspectives. Best and second-best scores are highlighted in bold and underlined, respectively.

improves factualness but also restructures triples into more atomic and consistent forms, reducing redundancy and stabilizing noisy representations. GraphRefine also achieves strong factual coverage (CS). Together, these findings suggest that it mitigates inconsistencies while preserving core information, alleviating the trade-off between noise removal and coverage.

For a more detailed analysis, we conducted a human evaluation on sampled KGs from DocRED and SciERC, two domain-distinct datasets.[3] Figure 3 presents the distribution of factual inconsistency types in our annotations. GraphJudge alleviates correctness-level inconsistencies (e.g., hallucination and misidentification), but its impact on representation-level inconsistencies (e.g., ambiguity and verbosity) is limited. These findings suggest that addressing representation-level issues often requires rewriting, rather than deletion alone.

[3]For each dataset, we sampled five generated KGs per method, yielding 15 KGs per dataset (30 total across two datasets). Annotators followed the guidelines in Appendix D.

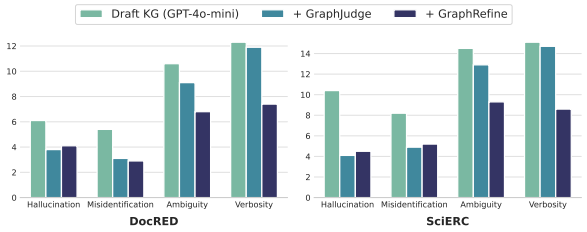


Figure 3: Distribution of factual inconsistency types in human-annotated KGs for DocRED and SciERC, where "+" indicates refinement applied to the drafts.

Meanwhile, GraphRefine substantially reduces the proportions of both inconsistency types, underscoring that operation-based refinement captures errors that deletion-only filtering may miss.

6.2 Model-Agnostic Applicability (RQ2)

Figure 4 illustrates a multi-aspect comparison on KGs generated by two distinct base extractors, including the unprocessed drafts and the results after applying GraphJudge or GraphRefine. Overall, GraphRefine consistently improves performance

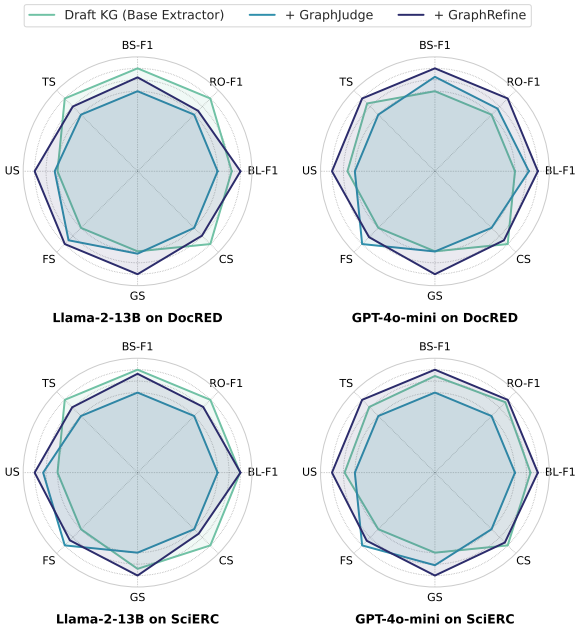


Figure 4: Multi-aspect comparison of draft KGs and their post-hoc refinement results. For visual clarity, each metric is min-max normalized and rescaled to $[0.7, 0.9]$.

across base extractors, boosting not only accuracy-oriented metrics (BL/RO/BS-F1) but also the multi-faceted GenRES measures. Across both datasets, it maintains or improves factualness (FS) while enhancing granularity (GS) and uniqueness (US), and it often improves coverage (CS) without sacrificing quality. By comparison, GraphJudge reduces correctness-level errors but tends to lower overall KG quality due to triple removal. This effect is more pronounced for Llama-2-13B, where the draft KGs are smaller and additional removals more directly reduce coverage and representation quality. Notably, GraphRefine improves factualness while also increasing GS and US even for smaller models, supporting its ability to correct and rewrite triples with document grounding. In summary, GraphRefine provides consistent gains for KGs produced by both open-source and proprietary LLM-based extractors, supporting its use as a model-agnostic, post-hoc KG refinement module. Extended results are provided in Appendix E.1.

6.3 Cross-Dataset Generalization (RQ3)

Table 2 presents the cross-dataset generalization performance of GraphRefine. First, GraphRefine achieves performance comparable to in-domain training under within-domain transfers (Doc → Gen and Gen → Doc), indicating stable transfer across datasets with similar domains. Moreover, even in cross-domain transfers (Doc → CDR and CDR → Doc), GraphRefine exhibits some degradation but largely maintains consistent performance across metrics. These findings suggest that GraphRefine captures document-grounded refinement behaviors that generalize beyond dataset-specific surface patterns. Notably, when trained on GenWiki and evaluated on SciERC (Gen → Sci), GraphRefine outperforms the SciERC in-domain model, implying that refinement behaviors learned from a more diverse training distribution improve transferability. In contrast, the reverse setting (Sci → Gen) leads to a substantially larger drop, suggesting that transfer becomes more challenging under larger domain shifts. This result motivates future work on mixed-domain training or domain adaptation to mitigate such domain gaps.

Src → Tgt	$\mathbf{BL_{F1}}(\Delta)$	$\mathbf{RO_{F1}}(\Delta)$	$\mathbf{BS_{F1}}(\Delta)$
Doc → Gen	64.60(↓0.63)	55.62(↓0.79)	84.19(↓1.13)
Gen → Doc	52.12(↓0.82)	37.54(↓0.75)	72.59(↓1.25)
Doc → CDR	41.01(↓1.72)	29.45(↓1.81)	62.92(↓1.18)
CDR → Doc	52.41(↓0.53)	37.91(↓0.38)	72.85(↓0.99)
Gen → Sci	56.46(↑1.03)	48.45(↑0.69)	77.87(↑1.58)
Sci → Gen	58.12(↓7.11)	51.40(↓5.01)	76.17(↓9.15)

Table 2: Results of cross-dataset generalization under train → test transfer across domain pairs, where Δ denotes change from in-domain training.

6.4 Further Analysis

We further analyze how GraphRefine improves performance by examining the behavior of its refinement operations and their contributions to each evaluation metric.

Operation-Level Confusion Analysis. We evaluate GraphRefine's operation prediction on synthetically generated test samples derived from DocRED. The confusion matrix in Figure 5 shows that GraphRefine separates non-edit vs. edit decisions reasonably well, while exhibiting a tendency toward REWRITE. For gold KEEP and DELETE, correct non-edit predictions dominate (78.1% and 71.3%), yet REWRITE is assigned to a non-trivial portion (11.1% and 18.0%). Among edit operations, FIX is the most challenging: 64.5% are correctly predicted, while 28.0% are mapped to REWRITE, reflecting overlap between meaning correction and surface-form normalization under document evidence. Meanwhile, REWRITE is relatively consistent (72.6%), suggesting a coherent rewrit-

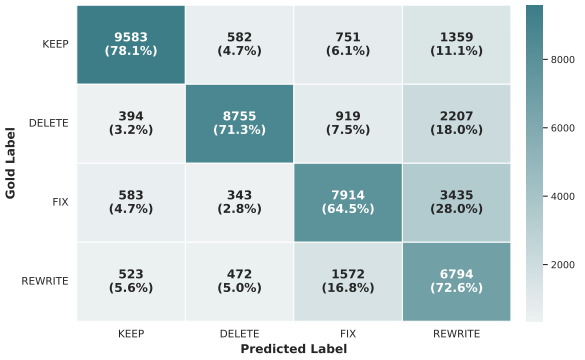


Figure 5: Operation-level confusion matrix of GraphRefine on the synthetically labeled DocRED test set.

ing policy. Overall, GraphRefine favors REWRITE-based refinement, which may improve representation quality, while leaving room to better calibrate rewriting to avoid unnecessary edits.

BLEU/ROUGE/BERTScore. Since BL/RO/BS capture surface- and semantic-level similarity to the ground-truth triples, all operations can improve them. DELETE removes unsupported triples, FIX corrects entities/relations, and REWRITE prunes redundant modifiers to better match the references. Overall, BL/RO/BS gains reflect both noise reduction and representational alignment.

Topical Similarity (TS). TS appears relatively low in Table 1 because it is influenced more by the base extractor's output characteristics than by GraphRefine itself. Since TS reflects how well extracted triples align with the document topic, unstable topical coherence or surface forms at extraction leave limited room for refinement to improve TS. This trend is also reflected in Figure 9, where TS can increase when GraphRefine is applied to high-TS extractors (e.g., GPT-4o-mini and GPT-4o). Therefore, GraphRefine does not inherently reduce TS; observed decreases are largely attributable to the underlying extractor.

Uniqueness (US). GraphRefine improves US by reducing redundancy among triples in the extracted KG. REWRITE makes triples more atomic and standardized, so the same fact is less likely to appear in multiple surface forms. FIX can provide a modest benefit by correcting spurious triples into valid facts, reducing noisy variants and better separating distinct facts. As a result, GraphRefine mainly reduces repeated paraphrases and can also alleviate cases where distinct meanings collapse into overly similar forms, leading to higher US.

Factualness (FS). Although GraphRefine does not always achieve the best FS, it consistently maintains robust factualness. By performing document-grounded refinement, it suppresses unsupported content while preserving document-supported statements. In particular, DELETE and FIX remove or correct clear factual inconsistencies, thereby sustaining high FS in practice.

Granularity (GS). LLM-based GKE often produces overly descriptive phrases or merges multiple attributes into a single triple. GraphRefine yields large gains in GS because REWRITE refactors such outputs into more atomic and clearer triples by removing unnecessary modifiers and making the core relation explicit, while staying faithful to the source document. This improves triple granularity, leading to higher GS.

Completeness (CS). CS remains high because GraphRefine edits triples beyond deletion to preserve supported information. FIX revises partially incorrect but correctable triples into document-supported statements instead of discarding them. REWRITE standardizes surface forms by resolving unclear mention spans and trimming verbose phrasing, making outputs closer to the ground truth. Together, these operations reduce errors while maintaining coverage, leading to strong CS.

7 Conclusion

In this paper, we proposed GraphRefine, a post-hoc refinement method to effectively mitigate factual inconsistencies in KGs constructed via LLM-based GKE. By analyzing the KGs generated across diverse GKE models and datasets, we systematically defined fine-grained factual inconsistency types. To this end, we fine-tuned an LLM as a KG refiner with operation-specific synthetic supervision to predict a refinement label for each triple and generate an edited triple. Extensive experiments using a diverse set of evaluation metrics show that GraphRefine consistently improves overall KG quality while reducing inconsistencies and preserving core information. Beyond correctness-level issues, it also improves representation-level quality, particularly in terms of granularity and uniqueness, leading to more atomic and consistent triples. Moreover, consistent gains across different base extractors and datasets further support its model-agnostic applicability and cross-dataset generalization, making it broadly applicable in real-world settings.

Limitations

LLM Stability and Reliability. Since GraphRefine relies on an LLM to choose refinement operations and perform edits, its quality depends on the model's reasoning stability and output reliability. When the model is uncertain or inconsistent, it may still edit decisively, causing unnecessary changes and over-correction. Still, document-grounded refinement helps suppress unsupported speculation and align edits with the input document. In future work, reliability could be further improved by leveraging approaches such as post-edit verification or evidence-constrained decoding.

Synthetic–Real Gap. GraphRefine fine-tunes on operation-specific synthetic noise, which may not fully match the complex errors produced by real GKE systems. However, this supervision scales without large manual labeling and yielded consistent gains across datasets and extractors. To reduce the gap, future work may calibrate synthesis rules with a small amount of human-verified data, mine failure patterns from real outputs to update the synthetic distribution, and iteratively add hard cases via self-training or active learning.

Cost and Scalability. Because GraphRefine runs LLM inference at the triple level, cost and latency can grow for large KGs or long documents. Practical deployment may require system optimizations for throughput, cost, and response time, especially to avoid redundant document–triple calls. These issues are common to LLM-based methods, and we observed improvements even with lightweight backbones, suggesting applicability under constrained budgets.

Ethical Considerations

All authors of this paper acknowledge and adhere to the ACL Code of Ethics. Annotators were compensated in accordance with institutional policies, and the study was designed to avoid undue burden or risk. We used existing datasets derived from publicly available and open-licensed resources. Our use complies with the terms of use and licensing policies of the corresponding datasets and platforms. While we did not observe any sensitive or potentially harmful content in our study, such content may still be present in the source corpora. We also leveraged AI assistants for editorial support, such as language polishing and grammar checking, to improve the writing quality of this paper.

References

Hanzhu Chen, Xu Shen, Qitan Lv, Jie Wang, Xiaoqi Ni, and Jieping Ye. 2024. Sac-kg: Exploiting large language models as skilled automatic constructors for domain knowledge graph. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4345–4360.

Jiuzhou Han, Nigel Collier, Wray Buntine, and Ehsan Shareghi. 2024. Pive: Prompting with iterative verification improving graph-based generative capability of llms. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 6702–6718.

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

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Elwin Huaman, Elias Kärle, and Dieter Fensel. 2020. Knowledge graph validation. *arXiv preprint arXiv:2005.01389*.

Haoyu Huang, Chong Chen, Zeang Sheng, Yang Li, and Wentao Zhang. 2025. Can llms be good graph judge for knowledge graph construction? In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 10940–10959.

Xiao Huang, Jingyuan Zhang, Dingcheng Li, and Ping Li. 2019. Knowledge graph embedding based question answering. In *Proceedings of the twelfth ACM international conference on web search and data mining*, pages 105–113.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, and 1 others. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Pengcheng Jiang, Jiacheng Lin, Zifeng Wang, Jimeng Sun, and Jiawei Han. 2024. Genres: Rethinking evaluation for generative relation extraction in the era of large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 2820–2837.

Zhijing Jin, Qipeng Guo, Xipeng Qiu, and Zheng Zhang. 2020. Genwiki: A dataset of 1.3 million content-sharing text and graphs for unsupervised graph-to-text generation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2398–2409.

Ryo Kamoi, Sarkar Snigdha Sarathi Das, Renze Lou, Jihyun Janice Ahn, Yilun Zhao, Xiaoxin Lu, Nan Zhang, Yusen Zhang, Ranran Haoran Zhang, Sujeeth Reddy Vummanthala, and 1 others. 2024. Evaluating llms at detecting errors in llm responses. *arXiv preprint arXiv:2404.03602*.

Jiho Kim, Sungjin Park, Yeonsu Kwon, Yohan Jo, James Thorne, and Yoonjae Choi. 2023. Factkg: Fact verification via reasoning on knowledge graphs. In *61st Annual Meeting of the Association for Computational Linguistics, ACL 2023*, pages 16190–16206. Association for Computational Linguistics (ACL).

Diederik P Kingma. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Jiao Li, Yueping Sun, Robin J Johnson, Daniela Sciaky, Chih-Hsuan Wei, Robert Leaman, Allan Peter Davis, Carolyn J Mattingly, Thomas C Wiegers, and Zhiyong Lu. 2016. Biocreative v cdr task corpus: a resource for chemical disease relation extraction. *Database*, 2016.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Yuxing Lu and Jinzhuo Wang. 2025. Karma: Leveraging multi-agent llms for automated knowledge graph enrichment. *arXiv preprint arXiv:2502.06472*.

Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. 2018. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3219–3232.

Belinda Mo, Kyssen Yu, Joshua Kazdan, Proud Mpala, Lisa Yu, Chris Cundy, Charilaos Kanatsoulis, and Sanmi Koyejo. 2025. Kggen: Extracting knowledge graphs from plain text with language models. *arXiv preprint arXiv:2502.09956*.

Christina Niklaus, Matthias Cetto, André Freitas, and Siegfried Handschuh. 2018. A survey on open information extraction. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3866–3878.

Songjie Niu, Kaisen Yang, Rui Zhao, Yichao Liu, Zonglin Li, Hongning Wang, and Wenguang Chen. 2025. Tree-kg: An expandable knowledge graph construction framework for knowledge-intensive domains. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 18516–18529.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

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

Swarnadeep Saha, Prateek Yadav, Lisa Bauer, and Mohit Bansal. 2021. Explagraphs: An explanation graph generation task for structured commonsense reasoning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7716–7740.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Zhen Wan, Fei Cheng, Zhuoyuan Mao, Qianying Liu, Haiyue Song, Jiwei Li, and Sadao Kurohashi. 2023. Gpt-re: In-context learning for relation extraction using large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3534–3547.

Shuhe Wang, Xiaofei Sun, Xiaoya Li, Rongbin Ouyang, Fei Wu, Tianwei Zhang, Jiwei Li, Guoyin Wang, and Chen Guo. 2025. Gpt-ner: Named entity recognition via large language models. In *Findings of the association for computational linguistics: NAACL 2025*, pages 4257–4275.

Xiangyu Wang, Lyuzhou Chen, Taiyu Ban, Muhammad Usman, Yifeng Guan, Shikang Liu, Tianhao Wu, and Huanhuan Chen. 2021. Knowledge graph quality control: A survey. *Fundamental Research*, 1(5):607–626.

Bingcong Xue and Lei Zou. 2022. Knowledge graph quality management: A comprehensive survey. *IEEE Transactions on Knowledge and Data Engineering*, 35(5):4969–4988.

Yuhao Yang, Chao Huang, Lianghao Xia, and Chenliang Li. 2022. Knowledge graph contrastive learning for recommendation. In *Proceedings of the 45th international ACM SIGIR conference on research and development in information retrieval*, pages 1434–1443.

Yuan Yao, Deming Ye, Peng Li, Xu Han, Yankai Lin, Zhenghao Liu, Zhiyuan Liu, Lixin Huang, Jie Zhou, and Maosong Sun. 2019. Docred: A large-scale document-level relation extraction dataset. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 764–777.

Bowen Zhang and Harold Soh. 2024. Extract, define, canonicalize: An llm-based framework for knowledge graph construction. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 9820–9836.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Weiyan Zhang, Wanpeng Lu, Jiacheng Wang, Yating Wang, Lihan Chen, Haiyun Jiang, Jingping Liu, and Tong Ruan. 2024. Unexpected phenomenon: Llms' spurious associations in information extraction. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 9176–9190.

Zikang Zhang, Wangjie You, Tianci Wu, Xinrui Wang, Juntao Li, and Min Zhang. 2025. A survey of generative information extraction. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 4840–4870.

A Datasets

In this appendix, we provide additional details of the datasets used in our experiments. Table 3 summarizes the dataset statistics, and Figure 6 illustrates the distributions of triple counts and document lengths for each dataset.

GenWiki. GenWiki (Jin et al., 2020) is a large-scale non-parallel graph-to-text dataset built from Wikipedia text and DBpedia graphs, consisting of 1.3 million text–graph pairs. For experimental efficiency, we employ the GenWiki-fine subset for both training and evaluation, removing extremely short documents and triples with formatting errors, following Huang et al. (2025).

DocRED. DocRED (Yao et al., 2019) is a dataset constructed for document-level relation extraction, containing manual annotations of entities, relations, and coreference information for 5,053 Wikipedia documents aligned with Wikidata. Compared to GenWiki, DocRED contains longer documents and a larger number of annotated triples, making it particularly suitable for evaluating model performance in more challenging and realistic scenarios. In this study, we use the human-annotated train and dev subsets while excluding documents with insufficient annotation quality.

SciERC. SciERC (Luan et al., 2018) is a domain-specific dataset built from 500 scientific abstracts, annotated with scientific entity types such as Task, Method, and Metric, as well as diverse semantic relations and coreference links. Due to the limited dataset size, we merge the train and dev subsets into a unified training set. In addition, following Huang et al. (2025), we exclude samples that contain empty graphs.

CDR. CDR (Li et al., 2016) is a biomedical-domain dataset designed for chemical–disease relation extraction, where expert annotators curated chemical entities, disease entities, and their interactions from 1,500 PubMed abstracts. Owing to its relatively large number of triples and longer documents compared to SciERC, CDR serves as a challenging benchmark for evaluating knowledge extraction in specialized domains. We also merge the train and dev subsets into a unified training set due to the limited volume of data.

Dataset	General-Domain		Domain-Specific	
	GenWiki	DocRED	SciERC	CDR
Train KGs	69,788	3,027	395	1,000
Test KGs	1,000	985	100	500
Train Triples	295,380	38,180	3,674	10,327
Test Triples	4,235	12,275	974	5,204
Avg. Triples	4.23	12.58	9.39	10.35
Avg. Length	24.22	198.33	130.99	231.66

Table 3: Statistics of the datasets. Avg. Triples and Avg. Length denote the average numbers of triples per KG and words per document, respectively.

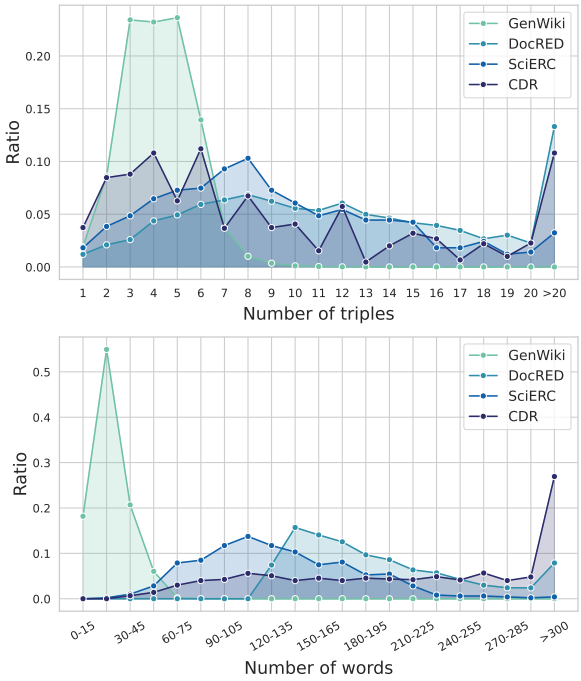


Figure 6: Distributional characteristics of the datasets. The top plot shows the average number of triples per document for each dataset, and the bottom plot shows the distribution of document lengths.

B Evaluation Metrics

B.1 G-Score

G-Score (Huang et al., 2025) is a family of automatic evaluation metrics that extends widely used text generation metrics to the task of KG evalua-

tion. It linearizes the predicted and gold KGs into triple sentences and measures their structural and semantic-level similarity.

G-BLEU. G-BLEU adapts BLEU (Papineni et al., 2002) for KG evaluation by measuring structural similarity based on n-gram overlap between the linearized predicted and ground-truth triple sentences. This metric focuses on the fidelity of generated triples to the phrasing and structural patterns of the gold triples. In our experiments, Precision, Recall, and Macro F1-score based on G-BLEU are reported as BL_P, BL_R, and BL_F1, respectively. Following Huang et al. (2025), the n-gram order is set to $n = 4$.

G-ROUGE. G-ROUGE applies ROUGE (Lin, 2004) to KG evaluation by measuring information coverage through n-gram co-occurrence statistics between the linearized predicted and ground-truth triple sentences. This metric evaluates how well the generated triples capture the key information contained in the gold triples. In our experiments, Precision, Recall, and Macro F1-score based on G-ROUGE are reported as RO_P, RO_R, and RO_F1, respectively. Following Huang et al. (2025), the n-gram order is set to $n = 2$.

G-BERTScore. G-BERTScore (Saha et al., 2021) extends BERTScore (Zhang et al., 2019) to the KG evaluation setting by treating each triple as a sentence and computing the semantic similarity between the predicted and ground-truth triples. In our experiments, Precision, Recall, and Macro F1-score based on G-BERTScore are reported as BS_P, BS_R, and BS_F1, respectively.

B.2 GenRES

GenRES (Jiang et al., 2024) is a suite of multi-aspect automatic metrics for evaluating generative triple extraction results. We adopt these metrics to assess KG quality. The detailed settings used for computing each metric are provided in Table 4. For any aspects not specified in our paper, we follow the original implementation.

Topical Similarity Score. Topical Similarity Score (TS) measures how well the extracted triples in $\mathcal{G}_\mathcal{D}$ align with the main topics of the source document $\mathcal{D}$. TS is computed by deriving LDA-based topic distributions for $\mathcal{D}$ and $\mathcal{G}_\mathcal{D}$ and evaluating their divergence as:

$$\text{TS}(\mathcal{D}, \mathcal{G}_\mathcal{D}) = e^{-\text{KL}(\theta_\mathcal{D} \| \theta_{\mathcal{G}_\mathcal{D}})}, \tag{14}$$

Hyperparameter/Model		Setting
LDA latent topics	GenWiki	200
	DocRED	100
	SciERC	50
	CDR	50
Triple threshold ϕ		0.95
LLM		`gpt-3.5-turbo-0125`
Embedding model		`text-embedding-ada-002`

Table 4: Detailed hyperparameters and model configurations used to compute the GenRES metrics.

where $\theta_\mathcal{D}$ and $\theta_{\mathcal{G}_\mathcal{D}}$ denote the topic distributions of $\mathcal{D}$ and $\mathcal{G}_\mathcal{D}$, respectively. A higher TS indicates stronger topical consistency of the extracted triples with the source document.

Uniqueness Score. Uniqueness Score (US) quantifies the diversity of the extracted knowledge by penalizing overly similar or duplicated triples. Let v_i denote the embedding of triple τ_i in $\mathcal{G}_\mathcal{D}$. US is computed as:

$$\text{US}(\mathcal{G}_\mathcal{D}) = \frac{1}{n(n-1)} \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} \mathbf{1}[\cos(v_i, v_j) < \phi], \tag{15}$$

where $n = |\mathcal{G}_\mathcal{D}|$ and ϕ is a cosine similarity threshold. A higher US indicates greater diversity of the extracted triples in $\mathcal{G}_\mathcal{D}$.

Factualness Score. Factualness Score (FS) measures whether each triple $\tau \in \mathcal{G}_\mathcal{D}$ is supported by the source document $\mathcal{D}$:

$$\text{FS}(\mathcal{D}, \mathcal{G}_\mathcal{D}) = \frac{1}{|\mathcal{G}_\mathcal{D}|} \sum_{\tau \in \mathcal{G}_\mathcal{D}} \mathbf{1}[\tau \text{ is supported by } \mathcal{D}]. \tag{16}$$

For each triple, the support decision is obtained by prompting an LLM-based fact checker. A higher FS indicates stronger factual correctness of the extracted triples in $\mathcal{G}_\mathcal{D}$.

Granularity Score. Granularity Score (GS) evaluates whether extracted triples are overly complex or properly decomposed into atomic relational units. Let n_τ denote the number of atomic triples derived from τ. To obtain n_τ, an LLM is prompted to determine whether a given triple can be further decomposed into finer-grained triples. GS is computed as:

$$\text{GS}(\mathcal{G}_\mathcal{D}) = \frac{1}{|\mathcal{G}_\mathcal{D}|} \sum_{\tau \in \mathcal{G}_\mathcal{D}} \frac{1}{e^{n_\tau}}. \tag{17}$$

A higher GS indicates better decomposition and relational granularity.

Completeness Score. Completeness Score (CS) measures how well the extracted KG $\mathcal{G}_\mathcal{D}$ covers the gold relational information in the ground-truth KG $\mathcal{G}^*_\mathcal{D}$. Let v_τ and v_{τ^*} denote the embedding vectors of an extracted triple τ and a gold triple τ^*, respectively. CS is computed as:

$$\mathrm{CS}(\mathcal{G}_\mathcal{D}, \mathcal{G}^*_\mathcal{D}) = \frac{1}{|\mathcal{G}^*_\mathcal{D}|} \sum_{\tau^* \in \mathcal{G}^*_\mathcal{D}} \mathbf{1}[\max_{\tau \in \mathcal{G}_\mathcal{D}} \cos(v_\tau, v_{\tau^*}) \geq \phi], \tag{18}$$

where ϕ is a cosine similarity threshold. A higher CS indicates stronger coverage of relational information in the ground-truth KG.

C Implementation Details

We fine-tune an LLM as a KG refiner in a supervised manner to judge and refine triples from a draft KG based on evidence in the source document. The model is optimized with Adam (Kingma, 2014) using 100 warmup steps, a learning rate of 3×10^{-4}, and 500 training steps. We train with a global batch size of 128 by using a micro-batch size of 8 and accumulating gradients for 16 steps. We apply LoRA to the `q_proj` and `v_proj` projection layers, with $r = 8$, $\alpha = 16$, and dropout $= 0.05$. For validation, we sample 2,000 instances from the training set for GenWiki and DocRED, and use a 0.2 split for SciERC and CDR. All results are reported as the average performance across three independent runs, each using a different random seed for generating the dataset used for fine-tuning. All experiments are conducted on two NVIDIA V100 32GB GPUs.

D Factual Inconsistencies in GKE

In this section, we describe the evaluation protocol used to analyze the factual inconsistency types and report detailed quantitative statistics.

D.1 Sampling and Evaluation Protocol

To comprehensively examine factual inconsistencies arising in GKE, we conducted a human evaluation by sampling KGs generated by multiple models across four datasets (GenWiki, DocRED, SciERC, and CDR). To cover a diverse range of LLM families and model sizes, we considered Llama-2-7B, Llama-2-13B, GPT-4o-mini, and GPT-4o. For each dataset, we sampled 20 KGs (4 models $\times$ 5 KGs each), resulting in 80 KGs evaluated at the triple level with a total of 658 triples. This human evaluation was performed by three graduate student annotators with sufficient expertise in KG construction. During evaluation, annotators were provided with the source document, the ground-truth KG, and the model-extracted KG for triple-level consistency assessment. We also adopted an evaluation criterion that maximizes recall from an OpenIE perspective (Niklaus et al., 2018). Specifically, we did not count a triple as an error if it does not explicitly appear in the ground-truth KG, as long as it can be reasonably inferred from the document and is semantically and structurally valid.

Task
Goal:
Given a source document and an extracted triple [head, relation, tail], determine whether the triple is **factually consistent** with the document by assigning a binary label: **CONSISTENT** or **INCONSISTENT**.
Provided Materials:
- Source document
- Model-extracted triple
- Ground-truth KG (for reference)

Notes
- The ground-truth KG may be **non-exhaustive**. Do **not** mark a triple as an error solely because it does not appear in the ground-truth KG.
- Following an OpenIE-style, recall-maximizing criterion, label a triple as **CONSISTENT** if it can be **reasonably inferred from the document** and is semantically and structurally valid, even if it is absent from the ground-truth KG.
- **Allowed inference** is limited to document-grounded reasoning (e.g., clear coreference, explicitly stated appositions/aliases, and cross-sentence entailment directly supported by the document).
- **Disallowed inference** includes relying on external/world knowledge or making speculative assumptions beyond textual evidence.
- **Surface form matters**: even if the meaning matches the document, mark a triple **INCONSISTENT** if the entity mentions are unclear or verbose.

Decision Criteria
CONSISTENT (all must hold)
- **Document support**: the document entails the relation stated in the triple.
- **Semantic alignment**: the relation meaning matches the document (no contradiction or polarity reversal).
- **Correct directionality**: head–tail roles align with the document (no subject–object swap).
- **Entity grounding**: head and tail refer to identifiable mentions in the document (or unambiguous coreference).
- **Appropriate surface form**: entity mentions are clear and concise.

INCONSISTENT (any applies)
- The triple is **unsupported** by the document or **contradicts** it.
- The meaning is **distorted** due to incorrect entity/relation interpretation.
- The triple has **incorrect directionality** (subject–object reversal).
- The entities are **not grounded** in the document (hallucinated or unresolvable mentions).
- **Inappropriate surface form**: entity mentions are unclear or overly verbose.

Figure 7: Guidelines for triple consistency annotation.

Stage 1 (Binary Consistency). The annotators independently and blindly judged whether each triple is factually consistent with the source document at the triple level. We used majority voting (at least 2 out of 3 annotators) to determine the final binary label, and a total of 283 triples were labeled as inconsistent and forwarded to Stage 2. The annotators followed a written guideline (Figure 7), and we report Fleiss' κ to measure inter-annotator agreement on this binary decision ($\kappa = 0.67$).

Stage 2 (Inconsistency Typing). For triples labeled as inconsistent, the annotators jointly performed a consolidation step to assign a salient in-

consistency type. During this stage, we excluded minor surface variations that do not affect factual meaning or were too sparse and idiosyncratic to form a meaningful category. As a result, among the 283 inconsistent triples, we assigned types to 221 triples using four categories, {UH, MI, EA, EV}. Each triple received a single primary type, and final labels were determined by consensus.

D.2 Types of Factual Inconsistencies

We now provide a detailed description of the factual inconsistencies observed in our human evaluation.

Correctness-level Inconsistencies. This type captures cases where a triple is factually incorrect with respect to the source document, thus should be removed or corrected. Such errors undermine the factual correctness of the KG and can substantially degrade its overall reliability. We categorize such cases into the following subtypes:

- **Unsupported Hallucination** (UH): Triples generated without document support, including cases where the fact is absent from or contradicts the source document.

- **Misidentification** (MI): Factually incorrect triples caused by misidentified entities or relations, which can often be corrected using evidence from the source document.

Representation-level Inconsistencies. This type captures cases where a triple is not necessarily incorrect, but its surface form is inconsistent or suboptimal, and thus can be improved via normalization or rewriting. Due to such representational issues, the resulting KG is not sufficiently consistent with the facts stated in the source document. We categorize such cases into the following subtypes:

- **Entity Ambiguity** (EA): Entity spans are overly narrow, yielding incomplete mentions (e.g., missing modifiers) that make the entity reference ambiguous or unstable.

- **Entity Verbosity** (EV): Entity spans are overly broad, including unnecessary surrounding words or clauses (e.g., descriptive phrases) that introduce redundancy and reduce representational consistency.

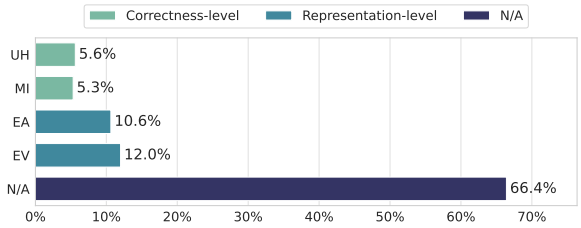


Figure 8: Proportions of factual inconsistencies in sampled KG triples. N/A includes valid triples and minor inconsistencies not covered by the four types.

Domain	Dataset	Model	NT	Corr.-level		Repr.-level	
				UH	MI	EA	EV
General	GenWiki	Llama-2-7B	2.6	-	-	1	2
		Llama-2-13B	3.2	2	1	2	1
		GPT-4o-mini	3.4	1	-	3	2
		GPT-4o	3.2	1	-	1	-
	DocRED	Llama-2-7B	2.6	1	-	3	4
		Llama-2-13B	7.8	4	2	5	6
		GPT-4o-mini	19	3	4	10	10
		GPT-4o	19	2	2	6	12
Specific	SciERC	Llama-2-7B	3.6	1	2	2	2
		Llama-2-13B	5.6	2	3	3	3
		GPT-4o-mini	9.8	4	4	6	7
		GPT-4o	11.4	1	1	5	6
	CDR	Llama-2-7B	1.8	1	-	2	1
		Llama-2-13B	6	4	3	4	5
		GPT-4o-mini	17.2	6	8	10	11
		GPT-4o	15.4	3	5	7	8

Table 5: Counts of factual inconsistency types by dataset and model. NT denote the average number of triples in the extracted KGs.

D.3 Quantitative Analysis

Figure 8 shows the distribution of factual inconsistencies. Among all sampled triples, Correctness-level inconsistencies account for 10.9% (UH+MI), while Representation-level inconsistencies account for 22.6% (EA+EV), indicating that representation issues are more frequent than strictly factual errors. This observation suggests that, beyond filtering clearly incorrect triples, improving extracted KGs often requires normalization and rewriting. In Table 5, the Corr.-level and Repr.-level columns report the counts of inconsistencies observed for each model on each dataset. Smaller models yield fewer inconsistencies in absolute terms because they produce smaller KGs, but they may exhibit a higher inconsistency density. Larger models reduce inconsistency rates, but non-trivial inconsistencies remain across all datasets. Notably, while LLMs tend to perform better on general-domain extraction, inconsistency issues persist across both general- and domain-specific datasets, indicating that refinement is needed regardless of domain.

Dataset	Model	Number of Triples		
		Draft KG	GraphJudge	GraphRefine
GenWiki	Llama-2-7B	2.35	2.08	2.16
	Llama-2-13B	3.75	3.27	3.34
	GPT-4o-mini	4.14	3.79	3.95
	GPT-4o	4.22	3.97	4.08
DocRED	Llama-2-7B	4.84	3.79	4.11
	Llama-2-13B	11.20	8.01	9.42
	GPT-4o-mini	20.48	15.82	18.13
	GPT-4o	20.51	16.17	19.40
SciERC	Llama-2-7B	3.32	2.74	2.86
	Llama-2-13B	3.77	3.33	3.38
	GPT-4o-mini	9.68	7.20	8.01
	GPT-4o	9.83	7.49	8.54
CDR	Llama-2-7B	1.47	1.14	1.25
	Llama-2-13B	4.63	3.67	3.81
	GPT-4o-mini	15.88	10.72	12.47
	GPT-4o	14.46	10.94	12.11

Table 6: Average number of triples per KG after applying GraphJudge or GraphRefine to draft KGs generated by different base extractors across datasets.

E Additional Experimental Results

In this appendix, we provide additional results to complement the main experiments.

E.1 GraphJudge vs. GraphRefine

Figure 9 presents a multi-metric comparison of applying GraphJudge and GraphRefine to draft KGs generated across all base extractors and datasets.

Overall Results. GraphRefine forms the largest polygon in most settings, showing consistent improvements not only on accuracy-oriented metrics (BL/RO/BS-F1) but also on the multi-faceted GenRES measures. In particular, it repeatedly improves granularity (GS) and uniqueness (US) while maintaining or improving factualness (FS). Coverage (CS) is also largely preserved or improved, suggesting that GraphRefine enhances KG quality through document-grounded correction and rewriting rather than simple deletion. In contrast, GraphJudge tends to reduce correctness-level errors, but its deletion-based filtering often decreases CS and is frequently accompanied by drops in representation quality (GS/US). This trade-off is more pronounced for the Llama-2 family (7B and 13B), which produces relatively smaller draft KGs, where additional removals more directly translate into coverage loss. Table 6 compares the average number of triples per KG for the draft KGs and the outputs after applying GraphJudge or GraphRefine. The results confirm that GraphJudge substantially reduces the absolute number of triples, whereas GraphRefine preserves the triple count to a much greater extent.

Dataset	Method	BL_{F1}	RO_{F1}	BS_{F1}	TS	US	FS	GS	CS
DocRED	D+F+R	52.94	38.29	75.83	36.72	89.46	96.38	97.10	59.63
	D+F	50.10	36.91	72.69	25.43	83.59	96.17	96.35	57.49
	D+R	49.84	37.23	71.08	27.29	88.13	95.32	97.21	55.15
	D	47.32	35.80	68.64	18.31	78.49	96.10	95.82	52.96
SciERC	D+F+R	55.43	48.07	76.29	79.61	94.42	97.31	86.33	56.75
	D+F	53.26	46.74	74.85	74.58	88.25	97.20	83.76	53.82
	D+R	52.84	46.38	73.18	76.14	92.61	96.45	85.61	50.37
	D	51.76	46.12	72.80	72.69	86.03	97.35	83.39	48.13

Table 7: Ablation results of GraphRefine on DocRED and SciERC. D, F, and R denote DELETE, FIX, and REWRITE, respectively.

Results by Base Extractor. For Llama-2-7B and Llama-2-13B, GraphRefine exhibits larger gains across all datasets, with repeated improvements in GS and US. By contrast, GraphJudge may improve some accuracy metrics but is often accompanied by decreases in CS and representation-level measures, highlighting the limitation of deletion-based refinement. For GPT-4o-mini and GPT-4o, GraphRefine provides consistent additional gains even on high-quality draft KGs, with stable improvements particularly on representation quality (GS/US). This indicates that GraphRefine does not depend heavily on model-specific error patterns. Instead, it leverages document-grounded correction and rewriting to function as a post-hoc module across diverse base extractors.

Summary across Datasets. Across GenWiki, DocRED, SciERC, and CDR, GraphRefine consistently improves BL/RO/BS-F1 while preserving factualness (FS) and enhancing granularity (GS) and uniqueness (US). Coverage (CS) is also often maintained or improved, indicating that the gains come from document-aligned correction and rewriting rather than simple removal. Overall, these results suggest that GraphRefine provides reliable post-hoc improvements across domains.

E.2 Ablation Study

Table 7 reports an ablation analysis of the refinement operations used in GraphRefine. D+F+R denotes the full model trained with DELETE, FIX, and REWRITE. D+F and D+R are trained without REWRITE and without FIX, respectively. D is trained with DELETE only.

Overall Results. The D+F+R demonstrates the strongest overall performance across both datasets, suggesting that a richer set of refinement operations improves KG quality more effectively than deletion alone. In particular, compared to D, D+F+R yields the largest gains on accuracy-oriented met-

rics (BL/RO/BS) and also attains the best scores on representation-related measures, including TS and US. Overall, these results indicate that GraphRefine is most effective when it combines document-grounded correction with rewriting, rather than relying solely on deletion.

Limitations of Deletion-Only Refinement. The deletion-only model variant (D) performs the worst on BL/RO/BS across both datasets and shows particularly large drops in TS and US, indicating degraded representation quality. It also achieves the lowest coverage (CS), suggesting that deletion-based refinement can reduce errors but often does so at the expense of information retention and structural quality. By contrast, D+F+R improves both correctness- and representation-related measures while maintaining higher CS, supporting the view that effective refinement requires editing operations beyond mere removal.

Contributions of FIX and REWRITE. Both D+F and D+R outperform D, indicating complementary contributions from FIX and REWRITE. D+F consistently improves BL/RO/BS and also increases FS and CS, suggesting that FIX corrects erroneous entities or relations using document evidence and strengthens factual correctness while preserving coverage. Meanwhile, D+R yields stronger gains on representation-oriented measures, especially US and GS, indicating that REWRITE reduces redundancy and rewrites triples into more atomic and consistent surface forms. Nevertheless, D+R shows lower CS in some settings, implying that rewriting alone may not fully preserve supported information and that combining it with FIX leads to the most stable improvements.

E.3 Case Study

Table 8 compares the outputs on the same source document from (i) a draft KG generated by GPT-4o-mini, (ii) GraphJudge, and (iii) GraphRefine. While the draft KG captures many core facts, it is fragmented by inconsistent entity mentions. For example, the same target is referred to as *UNESCO Confucius Prize for Literacy*, *Confucius Prize*, and *Prize*, which splits information across multiple subject nodes and weakens graph connectivity.

GraphJudge. GraphJudge improves KG quality primarily through deletion. As shown in Table 8, it removes many triples that are likely uncertain or weakly supported, resulting in a smaller KG. While this filtering suppresses noise and reduces factual errors, it also discards several document-grounded details (e.g., multiple beneficiary groups and topical descriptors), leading to lower coverage and a sparser description of the entity. Moreover, deletion does not resolve reference fragmentation: the output still distributes information across *UNESCO Confucius Prize for Literacy*, *Confucius Prize*, and *Prize*, so the graph remains partially disconnected even when remaining triples are correct.

GraphRefine. By contrast, GraphRefine edits triples via FIX/REWRITE operations rather than removal. A representative change is rewriting an ambiguous subject such as *Prize* into *UNESCO Confucius Prize for Literacy*, which consolidates previously scattered attributes and improves connectivity. Moreover, REWRITE makes triples more concise and atomic by splitting conflated phrases (e.g., separating *rural adults* and *out-of-school youth*) and removing unnecessary modifiers while staying faithful to the document. Overall, GraphRefine preserves most information from the draft KG while presenting it in a clearer, more compact form without substantially reducing the triple count.

F Prompts

For KG refinement, we input the source document together with a candidate triple and ask the model to select an operation and produce the corresponding output triple grounded in document evidence. Figure 10 shows the refinement prompt, which instructs the model to choose an appropriate operation and return either the original triple, a deletion decision, or a corrected/rewritten triple.

For knowledge extraction, we prompt an LLM to extract relations between entities in a document as [*head*, *relation*, *tail*] triples, using dataset-specific templates across domains following the prompt design of Jiang et al. (2024). Figure 11 presents the extraction prompt for the general-domain datasets GenWiki and DocRED, which restricts the output to a list of triples and includes two-shot examples. Figures 12 and 13 provide analogous prompts for the scientific dataset SciERC and the biomedical dataset CDR, respectively, tailored to each domain while preserving the same output constraints and two-shot examples.

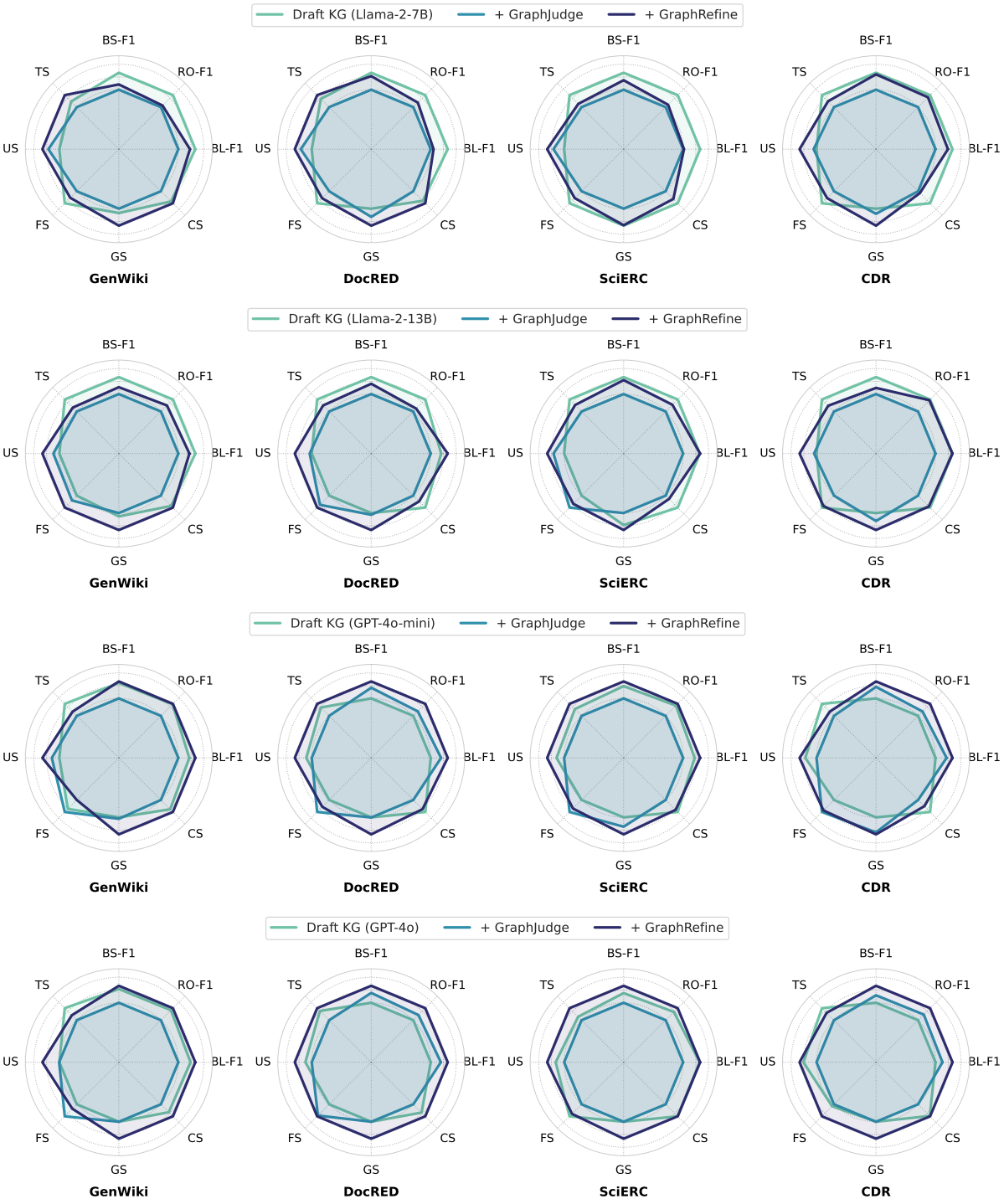
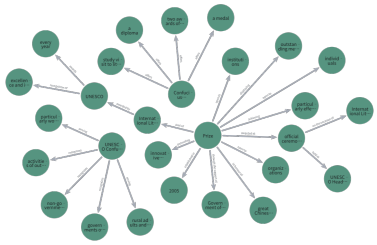


Figure 9: Multi-aspect comparisons between draft KGs and their post-hoc refinement results using GraphJudge or GraphRefine across all datasets (GenWiki, DocRED, SciERC, and CDR) and base extractors (Llama-2-7B, Llama-2-13B, GPT-4o-mini, and GPT-4o). For visual clarity, all metrics are min–max normalized and rescaled to $[0.7, 0.9]$.

Original Document:

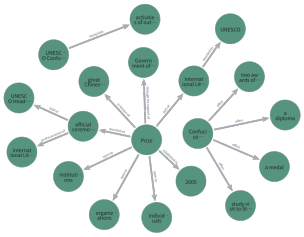
The UNESCO Confucius Prize for Literacy recognizes the activities of outstanding individuals, governments or governmental agencies and non-governmental organizations (NGOs) working in literacy serving rural adults and out-of-school youth, particularly women and girls. The Prize was established in 2005 through the support of the Government of the People's Republic of China in honour of the great Chinese scholar Confucius. It is part of the International Literacy Prizes, which UNESCO awards every year in recognition of excellence and inspiring experiences in the field of literacy throughout the world. The Confucius Prize offers two awards of US$ 20,000 each, a medal and a diploma, as well as a study visit to literacy project sites in China. The Prize is open to institutions, organizations or individuals displaying outstanding merit in literacy, achieving particularly effective results and promoting innovative approaches. The selection of prizewinners is made by an International Jury appointed by UNESCO's Director-General, which meets in Paris once a year. The Prize is awarded at an official ceremony held for that purpose at UNESCO Headquarters in Paris on the occasion of International Literacy Day (8 September).

Draft KG (GPT-4o-mini):



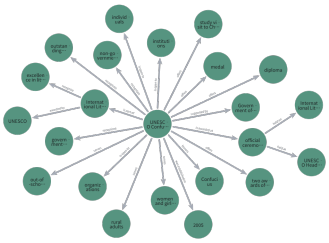
[['UNESCO Confucius Prize for Literacy', 'recognizes', 'activities of outstanding individuals'], ['UNESCO Confucius Prize for Literacy', 'recognizes', 'governments or governmental agencies'], ['UNESCO Confucius Prize for Literacy', 'recognizes', 'non-governmental organizations (NGOs)'], ['UNESCO Confucius Prize for Literacy', 'serving', 'rural adults and out-of-school youth'], ['UNESCO Confucius Prize for Literacy', 'serving', 'particularly women and girls'], ['Prize', 'established in', '2005'], ['Prize', 'through the support of', "Government of the People's Republic of China"], ['Prize', 'in honour of', 'great Chinese scholar Confucius'], ['Prize', 'part of', 'International Literacy Prizes'], ['International Literacy Prizes', 'awarded by', 'UNESCO'], ['UNESCO', 'awards', 'every year'], ['UNESCO', 'recognition of', 'excellence and inspiring experiences in the field of literacy'], ['Confucius Prize', 'offers', 'two awards of US$ 20,000 each'], ['Confucius Prize', 'offers', 'a medal'], ['Confucius Prize', 'offers', 'a diploma'], ['Confucius Prize', 'offers', 'study visit to literacy project sites in China'], ['Prize', 'open to', 'institutions'], ['Prize', 'open to', 'organizations'], ['Prize', 'open to', 'individuals'], ['Prize', 'displaying', 'outstanding merit in literacy'], ['Prize', 'achieving', 'particularly effective results'], ['Prize', 'promoting', 'innovative approaches'], ['Prize', 'awarded at', 'official ceremony'], ['official ceremony', 'held at', 'UNESCO Headquarters in Paris'], ['official ceremony', 'on the occasion of', 'International Literacy Day (8 September)']]

GraphJudge:



[['UNESCO Confucius Prize for Literacy', 'recognizes', 'activities of outstanding individuals'] ['Prize', 'established in', '2005'] ['Prize', 'through the support of', "Government of the People's Republic of China"] ['Prize', 'in honour of', 'great Chinese scholar Confucius'] ['Prize', 'part of', 'International Literacy Prizes'] ['International Literacy Prizes', 'awarded by', 'UNESCO'] ['Confucius Prize', 'offers', 'two awards of US$ 20,000 each'] ['Confucius Prize', 'offers', 'a medal'] ['Confucius Prize', 'offers', 'a diploma'] ['Confucius Prize', 'offers', 'study visit to literacy project sites in China'] ['Prize', 'open to', 'institutions'] ['Prize', 'open to', 'organizations'] ['Prize', 'open to', 'individuals'] ['Prize', 'awarded at', 'official ceremony'] ['official ceremony', 'held at', 'UNESCO Headquarters in Paris'] ['official ceremony', 'on the occasion of', 'International Literacy Day (8 September)']]

GraphRefine:



[['UNESCO Confucius Prize for Literacy', 'recognizes', 'outstanding individuals'], ['UNESCO Confucius Prize for Literacy', 'recognizes', 'governmental agencies'], ['UNESCO Confucius Prize for Literacy', 'recognizes', 'non-governmental organizations (NGOs)'], ['UNESCO Confucius Prize for Literacy', 'serves', 'rural adults'], ['UNESCO Confucius Prize for Literacy', 'serves', 'out-of-school youth'], ['UNESCO Confucius Prize for Literacy', 'serves', 'women and girls'], ['UNESCO Confucius Prize for Literacy', 'was established in', '2005'], ['UNESCO Confucius Prize for Literacy', 'supported by', "Government of the People's Republic of China"], ['UNESCO Confucius Prize for Literacy', 'honors', 'Confucius'], ['UNESCO Confucius Prize for Literacy', 'is part of', 'International Literacy Prizes'], ['International Literacy Prizes', 'awarded by', 'UNESCO'], ['UNESCO', 'awards', 'International Literacy Prizes annually'], ['International Literacy Prizes', 'recognize', 'excellence in literacy'], ['UNESCO Confucius Prize for Literacy', 'offers', 'two awards of US$ 20,000'], ['UNESCO Confucius Prize for Literacy', 'offers', 'medal'], ['UNESCO Confucius Prize for Literacy', 'offers', 'diploma'], ['UNESCO Confucius Prize for Literacy', 'offers', 'study visit to China'], ['UNESCO Confucius Prize for Literacy', 'is open to', 'institutions'], ['UNESCO Confucius Prize for Literacy', 'is open to', 'organizations'], ['UNESCO Confucius Prize for Literacy', 'is open to', 'individuals'], ['UNESCO Confucius Prize for Literacy', 'is awarded at', 'official ceremony'], ['official ceremony', 'held at', 'UNESCO Headquarters, Paris'], ['official ceremony', 'held on', 'International Literacy Day (8 September)']]

Table 8: Qualitative comparison of knowledge extraction results (Draft KG vs. GraphJudge vs. GraphRefine).

Knowledge Graph Refinement

```
Given a source document and a candidate triple extracted from it, decide how to refine
the triple based only on the document.

Choose exactly one operation: KEEP, DELETE, FIX, or REWRITE.
- KEEP: The triple is supported by the document and its representation is appropriate;
keep it unchanged.
- DELETE: The triple is not supported by the document or is factually wrong; remove it.
- FIX: The document supports a similar fact, but the candidate triple's meaning is
incorrect (e.g., wrong entity or relation); correct it to match the document.
- REWRITE: The fact is supported, but the triple's representation is ambiguous,
imprecise, or verbose; rewrite it to a clearer and more consistent form without changing
the meaning.

Return ONLY one item in the exact format:
['OPERATION', ['ENTITY 1', 'RELATIONSHIP', 'ENTITY 2']]
If OPERATION is DELETE, return:
['DELETE', ['NULL', 'NULL', 'NULL']]

Do not output any additional text. Do not explain your decision.

document: $TEXT$
triple: $TRIPLE$
output:
```

Figure 10: Prompt template for KG refinement.

General-Domain Knowledge Extraction

```
Given a prompt, identify and list the relationships between entities within the text.
Extract relationships both within a single sentence (intra-sentence) and across multiple
sentences (inter-sentence).
Provide a list of triplets in the format ['ENTITY 1', 'RELATIONSHIP', 'ENTITY 2']. The
relationship is directed, so the order of entities in each triplet matters.
The output should only be a list of triplets ([['ENTITY 1', 'RELATIONSHIP', 'ENTITY 2'],
...]) without any additional information. Do not explain how you extract them.

Example 1:
prompt: In 2020, the Nobel Peace Prize was awarded to the World Food Programme for its
efforts to combat hunger. The organization has been operational since 1961.
relations:
[['Nobel Peace Prize', 'awarded in', '2020'], ['Nobel Peace Prize', 'awarded to', 'World
Food Programme'], ['World Food Programme', 'efforts to', 'combat hunger'], ['World Food
Programme', 'operational since', '1961']]

Example 2:
prompt: The Great Barrier Reef, located off the coast of Australia, is the world's
largest coral reef system. It has been severely affected by climate change, leading to
coral bleaching.
relations:
[['Great Barrier Reef', 'located at', 'coast of Australia'], ['Great Barrier Reef', 'is',
'world's largest coral reef system'], ['Great Barrier Reef', 'affected by', 'climate
change'], ['Climate change', 'leads to', 'coral bleaching']]

prompt: $TEXT$
relations:
```

Figure 11: Prompt template for general-domain knowledge extraction (GenWiki and DocRED).

Scientific-Domain Knowledge Extraction

```
Given a prompt, identify and list the relationships between entities within the text.
Extract relationships both within a single sentence (intra-sentence) and across multiple
sentences (inter-sentence).
Provide a list of triplets in the format ['ENTITY 1', 'RELATIONSHIP', 'ENTITY 2']. The
relationship is directed, so the order of entities in each triplet matters.
The output should only be a list of triplets ([['ENTITY 1', 'RELATIONSHIP', 'ENTITY 2'],
...]) without any additional information. Do not explain how you extract them.

Example 1:
prompt: Sources of training data suitable for language modeling of conversational speech
are limited . In this paper , we show how training data can be supplemented with text
from the web filtered to match the style and/or topic of the target recognition task ,
but also that it is possible to get bigger performance gains from the data by using
class-dependent interpolation of N-grams .
relations:
[['conversational speech', 'used for', 'language modeling'], ['class-dependent
interpolation of N-grams', 'used for', 'recognition task']]

Example 2:
prompt: We propose a draft scheme of the model formalizing the structure of communicative
context in dialogue interaction . The relationships between the interacting partners are
considered as system of three automata representing the partners of the dialogue and
environment .
relations:
[['model', 'used for', 'structure of communicative context'], ['dialogue interaction',
'feature of', 'structure of communicative context']]

prompt: $TEXT$
relations:
```

Figure 12: Prompt template for scientific-domain knowledge extraction (SciERC).

Biomedical-Domain Knowledge Extraction

```
Given a prompt, identify and list the relationships between entities within the text.
Extract relationships both within a single sentence (intra-sentence) and across multiple
sentences (inter-sentence).
Provide a list of triplets in the format ['ENTITY 1', 'RELATIONSHIP', 'ENTITY 2']. The
relationship is directed, so the order of entities in each triplet matters.
The output should only be a list of triplets ([['ENTITY 1', 'RELATIONSHIP', 'ENTITY 2'],
...]) without any additional information. Do not explain how you extract them.

Example 1:
prompt: Penicillin is an antibiotic that treats bacterial infections. It was discovered
by Alexander Fleming.
relations:
[['Penicillin', 'is a type of', 'antibiotic'], ['Penicillin', 'treats', 'bacterial
infections'], ['Penicillin', 'discovered by', 'Alexander Fleming']]

Example 2:
prompt: Metformin is commonly prescribed for managing type 2 diabetes. It helps by
lowering glucose production in the liver and increasing the body's sensitivity to insulin.
relations:
[['Metformin', 'is prescribed for', 'managing type 2 diabetes'], ['Metformin', 'helps
by', 'lowering glucose production in the liver'], ['Metformin', 'increases', 'body\'s
sensitivity to insulin']]

prompt: $TEXT$
relations:
```

Figure 13: Prompt template for biomedical-domain knowledge extraction (CDR).