Tree-KG: An Expandable Knowledge Graph Construction Framework for Knowledge-intensive Domains

Songjie Niu[1*], **Kaisen Yang**[1*], **Rui Zhao**[1], **Yichao Liu**[1],
Zonglin Li[1], **Hongning Wang**[1†], **Wenguang Chen**[1,2†]
[1] Tsinghua University, [2] Tsinghua Shenzhen International Graduate School
{niusongjie, hw-ai, cwg}@tsinghua.edu.cn
{yks23, zhaor23, liuyicha23, lizl22}@mails.tsinghua.edu.cn

Abstract

In knowledge-intensive domains like scientific research, effective decisions rely on organizing and retrieving intricate data. Knowledge graphs (KGs) help by structuring entities, relations, and contextual dependencies, but building KGs in such domains is challenging due to inherent complexity, manual effort, and rapid evolution. Inspired by how humans organize knowledge hierarchically, we propose Tree-KG, an expandable framework that combines structured domain texts with advanced semantic techniques. First, Tree-KG builds a tree-like graph from textbook structures using large language models (LLMs) and domain-specific entities, creating an *explicit KG*. Then, through iterative expansion with flexible, predefined operators, it uncovers *hidden KG* while preserving semantic coherence. Experiments demonstrate that Tree-KG consistently surpasses competing methods, achieving the highest F1 scores (12–16% above the second-best), with notable performance (F1 0.81) on the Text-Annotated dataset, highlighting its effectiveness in extracting high-quality information from source texts. Additionally, Tree-KG provides superior structural alignment, domain-specific extraction, and cost-efficiency, delivering robust results with reduced token usage and adaptable, resource-conscious deployment. [1]

1 Introduction

Knowledge-intensive domains refer to areas where specialized knowledge, deep expertise, and extensive information processing are required, such as scientific research, healthcare, finance, and law (Xu et al., 2024; Marjanovic, 2011; Zhao et al., 2023). Knowledge graphs (KGs) provide a structured representation of domain-specific knowledge, capturing entities, relations, and their contextual dependencies (Chandak et al., 2022; Santos et al., 2022). In knowledge-intensive domains, knowledge graphs facilitate efficient information retrieval, reasoning and decision support (Su et al., 2024; Chen et al., 2024). However, constructing KGs for knowledge-intensive domains remains a challenge due to knowledge complexity, substantial manual effort, and rapid evolution (Yan et al., 2024).

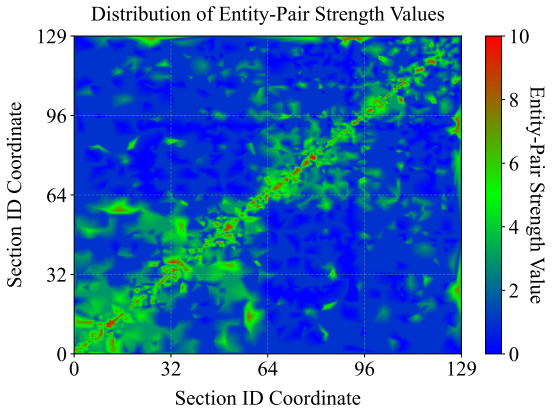


Figure 1: Distribution of Entity-Pair Strength Values. A physics textbook was divided into 129 sections, with approximately 1,000 extracted entities assigned section IDs. We randomly sampled 5,000 entity pairs (potential relations) and used LLMs to score their relation strength (0–10). The x-axis shows the first entity's section ID, and the y-axis shows the second's.

Several methods have been employed for constructing KGs. Rule-based systems rely on predefined logical rules to extract and structure knowledge (Suchanek et al., 2007; Galárraga et al., 2013; Muggleton, 1997). While they offer high precision and domain-specific control, they suffer from scalability issues, limited generalization and brittleness. Supervised learning models leverage annotated datasets to learn patterns for knowledge extraction (Yates et al., 2007; Banko et al., 2007; Kim and Moldovan, 1993; Mintz et al., 2009). These approaches face several challenges of high annotation costs, limited adaptability, and dependency on training data. LLMs have been increasingly used

[*] Equal contribution
[†] Corresponding authors
[1] https://github.com/thu-pacman/Tree-KG

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 18516–18529

for automated knowledge extraction and graph construction (Zhang and Soh, 2024; Feng et al., 2024; Couto and Ebecken, 2024). However, most of them lack a well-defined knowledge structure, semantic consistency, and incremental mechanisms, which limits their scalability across domains.

By drawing on insights from human cognition, we can better construct KGs for knowledge-intensive domains. Humans often structure information hierarchically, much like how textbooks are arranged. This organization aligns with the intuitive way that knowledge is typically accessed, processed, and connected. Figure 1 shows that entities close together in the textbook have stronger connections, with three key observations: (a) the textbook's structure is exploitable, (b) entities within the same section have the strongest links, and (c) local context matters, as even slightly deviating pairs show significant relationships. These findings inspire the design of our KG construction method.

To address the challenges of KG construction in knowledge-intensive domains, we propose a framework named Tree-KG that builds structured knowledge and supports continuous expansion. Our approach consists of two key phases: Phase 1 constructs a hierarchical ***explicit KG*** from structured domain corpora, using text segmentation and LLM-generated summaries to form context-aware nodes and vertical edges; Phase 2 iteratively expands this structure by revealing ***hidden KG*** via contextual convolution, aggregation, embedding, deduplication, and edge prediction, while merging new textual sources to ensure robust scalability.

The key contributions of this paper are:

Expandable KG Construction Framework. We introduce Tree-KG, a expandable framework that integrates structured domain corpora with large language models to build and continuously expand knowledge graphs. This framework leverages both explicit and hidden KG construction to provide comprehensive, context-aware representations.

Novel Tree-like Hierarchical Graph. Our approach formulates a tree-like hierarchical graph that organizes domain knowledge in a clear, rational structure. This design standardizes the form of knowledge representation and enhances the ease of extension and navigation through complex, knowledge-intensive domains.

Operator-Driven Expansion Mechanisms. We propose a set of flexible, predefined operators that drive the iterative expansion of the KG. These operators utilize textual and structural cues to extract latent information, refine entity relationships, and ensure systematic, expandable growth while maintaining semantic coherence.

Experiments demonstrate that Tree-KG consistently surpasses competing methods, achieving the highest F1 scores (12–16% above the second-best), with notable performance (F1 0.81) on the Text-Annotated dataset, highlighting its effectiveness in extracting high-quality information from source texts. Additionally, Tree-KG provides superior structural alignment, domain-specific extraction, and cost-efficiency, delivering robust results with reduced token usage and adaptable, resource-conscious deployment.

2 Related Work

The landscape of KG construction has evolved from traditional rule-based and supervised techniques to advanced approaches leveraging LLMs, each with distinct strengths and limitations.

Rule-based and Supervised Methods. Early approaches, such as YAGO (Suchanek et al., 2007), exploited hierarchical structures for semantic knowledge extraction. Concurrently, self-supervised models, TextRunner (Banko and Etzioni, 2008), leveraged CRFs for sequence labeling to extract relations, classes, and entities from text. Later, span-level models(Zhan and Zhao, 2019) and text-to-triple translation frameworks (Wang et al., 2021) expanded extraction scope. Despite performance, these methods require tuning, lack scalability.

LLM-based Methods. Recent progress in LLMs has catalyzed the development of innovative techniques for constructing KGs. **(a) Ontology-based.** In this approach, raw, domain-specific corpora are processed using an ontology sourced from repositories such as Wikipedia. Candidate relations are generated through LLMs (Ding et al., 2024), resulting in high-quality graphs enriched by common-sense reasoning. However, the performance of this strategy is contingent upon the depth and accuracy of the external ontology. **(b) Fine-tuning.** Methods like AutoRE (Xue et al., 2024) employ document-level relation extraction without a predetermined relation set. By integrating Parameter Efficient Fine Tuning (PEFT) techniques, such as QLoRA, these approaches yield competitive results in controlled settings. Their limitations include difficulties in managing the vast number of relations in practical applications and challenges

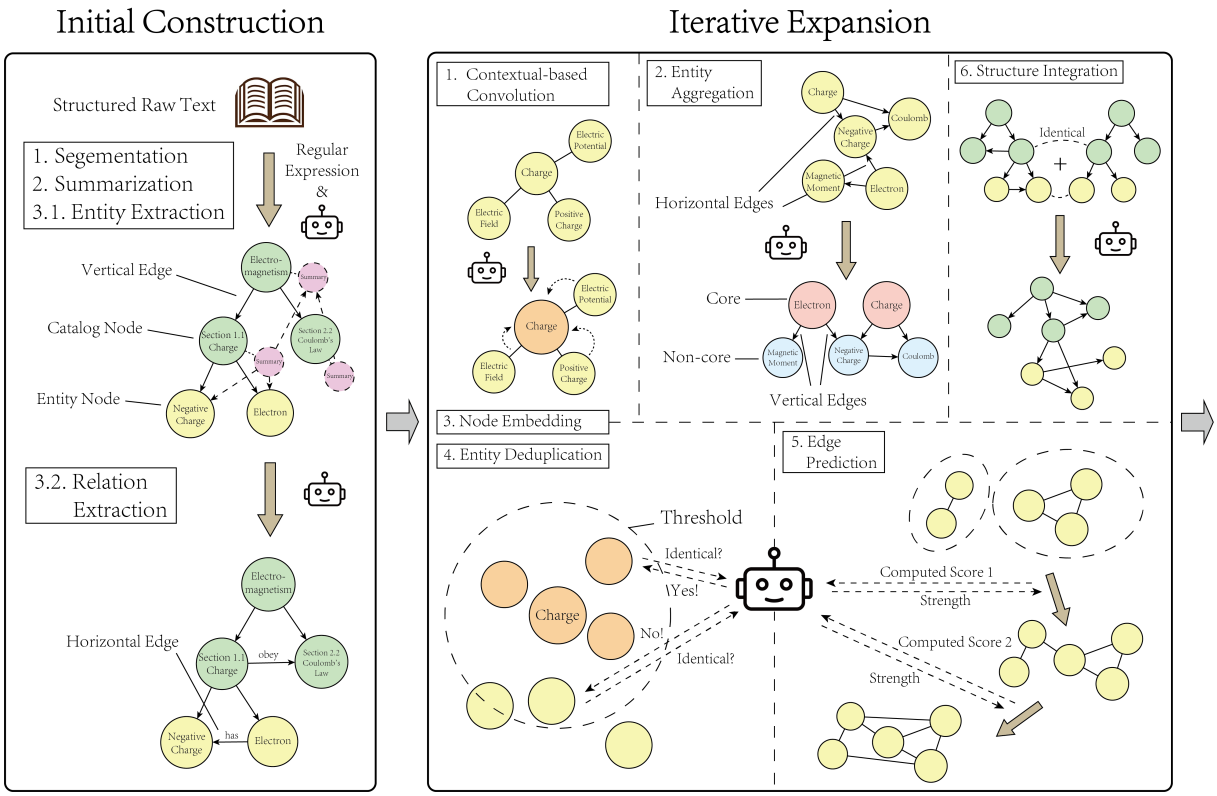


Figure 2: An overview of Tree-KG. Tree-KG consists of two major phases: initial construction and iterative expansion. Specifically, the pipeline starts by processing hierarchically structured textbooks to build the *explicit KG*. This forms the skeleton of our knowledge representation. Once the explicit KG is built, we perform an iterative expansion to reveal the *hidden KG*. This phase involves advanced operations to further enhance the KG.

in generalizing to unseen data. **(c) Zero- and Few-shot Learning.** Iterative zero-shot prompting (Carta et al., 2023) and few-shot strategies like GraphRAG (Edge et al., 2024) balance extraction efficiency and recall. While they reduce dependence on labeled data, they may face issues with imprecise entity resolution and computational overhead. **(d) Knowledge-intensive Domains.** Systems such as SAC-KG (Chen et al., 2024) treat LLMs as domain experts, leveraging multi-stage pipelines to enhance precision despite necessitating extensive calibration. Domain-specific systems like MathGraph (Zhao et al., 2019) and others (Dang et al., 2021) focused on educational KGs further demonstrate that tailored strategies can enrich domain knowledge, though their applicability remains limited to targeted scenarios.

Collectively, an effective framework for constructing KGs in knowledge-intensive domains must balance automation, accuracy, adaptability, and incremental knowledge integration. Our framework Tree-KG can make it possible to automatically extract, refine, and integrate new sources into a unified KG structure.

3 Methodology

Our framework Tree-KG is designed to systematically construct KGs for knowledge-intensive domains by leveraging the inherent structural and semantic information in domain literature. An overview of Tree-KG is demonstrated in Figure 2.

3.1 Definitions

We first define the key concepts that establish the design foundation of our framework Tree-KG discussed later.

3.1.1 Tree-like Hierarchical Graph

Let $G = (V, E)$ be a graph, where V and E denote the set of nodes and edges respectively. A tree-like hierarchical graph is a type of graph representation that follows a hierarchical structure similar to a tree. The node set V is partitioned into k layers as:

$$V = V_1 \cup V_2 \cup \cdots \cup V_k,$$

where each subset V_i corresponds to a specific layer, and $V_i \cap V_j = \emptyset$ for $i \neq j$.

The edge set E is further distinguished into two types and can be expressed as:

$$E = E_1 \cup E_2.$$

Vertical edges E_1: These edges connect nodes from different layers, i.e., they span across layers. For each edge $(u, v) \in E_1$, if $u \in V_i$ and $v \in V_j$, then $|i-j| = 1$, meaning the edge can only connect adjacent layers. The nodes along vertical edges form a tree.

Horizontal edges E_2: These edges connect nodes within the same layer, i.e., they connect nodes at the same level. For each edge $(u, v) \in E_2$, if $u, v \in V_i$, then the edge connects nodes within the same layer V_i.

3.1.2 KG Schema

Tree-KG adopts a tree-like hierarchical graph structure. Each node has three main attributes: **name**, the node's designated name; **description**, initially extracted from source text (3.2) and later enriched via LLMs by integrating descriptions from neighbors (3.3.1); and **relations**, the edges connecting it to other nodes. For edges, *homogeneous* **vertical edges** represent hierarchical or subordinate relationships, with three predefined types: has_subsection (section–subsection), has_entity (subsection–entity), and has_subordinate (core–non-core entity). In contrast, *heterogeneous* **horizontal edges** combine a general category (section_related or entity_related) with a specific LLM-predicted relation (e.g., obey, has in Figure 2).

3.1.3 Adamic-Adar (AA) Score

(Adamic and Adar, 2003)

$$AA(u, v) = \sum_{w \in N(u) \cap N(v)} \frac{1}{\log(|N(w)|)},$$

where $N(u)$ and $N(v)$ are the neighbor sets of nodes u and v respectively, $N(u) \cap N(v)$ is the set of common neighbors between u and v, and $|N(w)|$ is the degree (number of neighbors) of w.

3.1.4 Number of Common Ancestors

Given two nodes u and v in an unconstrained layered graph, the number of common ancestors $CA(u, v)$ is defined as follows:

Lineage Paths: From node u and node v, traverse upwards through vertical edges (edges between adjacent layers) until reaching the first layer. Let the lineage path sets from node u and node v to the first layer be P_u and P_v, respectively.

Common Ancestors: For each pair of lineage paths from P_u and P_v, compute the intersection of the paths. The number of common ancestors $CA(u, v)$ is the maximum size of the intersection of any pair of lineage paths, i.e.,

$$CA(u, v) = \max\left(|P_u^{(i)} \cap P_v^{(j)}|\right),$$

where $P_u^{(i)}$ and $P_v^{(j)}$ are the i-th and j-th paths in P_u and P_v respectively, and $|\cdot|$ denotes the size of the intersection of the two paths.

3.2 Initial Construction

Starting with domain corpora (i.e., most notably hierarchically structured textbooks), we extract the primary skeleton of the knowledge graph. This *explicit KG* represents a top-down view of the domain, capturing relations like chapter–section-subsection–entity. We leverage this natural structure to build a tree-like hierarchical graph in 3.1.1.

3.2.1 Text Segmentation

We use tailored regex patterns to parse textbooks and identify chapter, section, and subsection boundaries based on formatting cues (e.g., numbered headings). This segmentation produces hierarchical table-of-contents (TOC) nodes corresponding to the book, its chapters, sections, and subsections, establishing initial has_subsection relations.

3.2.2 Bottom-Up Summarization

For subsection nodes without further subdivisions, we use LLMs to generate summaries based on texts that capture essential contents and domain-specific terms. Summaries for higher-level TOC nodes are created by aggregating the summaries of their child nodes, ensuring that overarching topics reflect detailed contents of their subsections. This bottom-up approach mimics human review processes.

3.2.3 Entity and Relation Extraction

For each TOC node, descriptions from its parent provide context to analyze child-node relations. LLMs extract these as section_related edges, paired with specific predicted types (see 3.1.2).

Subsection summaries are further processed using LLMs to identify entities and derive has_entity edges and entity_related edges combined with specific LLM-predicted relations. This grounds the KG in fine-grained, domain-specific knowledge.

3.3 Iterative Expansion

While the explicit KG constructed from notable textbooks provides a robust skeleton, it may miss latent or implicit connections within the domain. The iterative expansion phase enriches the initial KG by uncovering hidden relations between entities with the integration of textual and structural contexts. This phase employs operators in a predefined action set including contextual-based convolution (*conv*), entity aggregation (*aggr*), node embedding (*embed*), entity deduplication (*dedup*), edge prediction (*pred*), and structure integration (*merge*), to reveal the *hidden KG*.

3.3.1 Contextual-based Convolution

Inspired by traditional graph convolution, we update each entity's description using a contextual-based convolution. For each entity v with neighboring entities $\mathcal{N}(v)$, we use LLMs to generate a report that includes a concise definition of v, detailed descriptive and contextual information about v, and the role of v within its local subgraph reflecting both its features and those of $\mathcal{N}(v)$.

Mathematically, the update is expressed as:

$$h_v^{(\text{new})} = conv\Big(h_v, \{h_u : u \in \mathcal{N}(v)\}\Big),$$

where h_v is the original description of entity v and $h_v^{(\text{new})}$ is its updated description. Experiments show that a single convolution step is typically sufficient.

3.3.2 Entity Aggregation

Each entity is deliberately assigned a *local role* $r \in \{\text{core}, \text{non-core}\}$ by LLMs based on its description. For each core entity, we aggregate its neighboring peripheral non-core entities as its children by transforming horizontal edges into vertical ones, thereby forming has_subordinate edges (see 3.1.2). For instance, if "electric charge" is core, related phenomena like the "triboelectric effect" become subordinate. Thus, entities comprise the lowest two layers of the tree-like hierarchical graph: the core entity layer and the non-core entity layer. *aggr* helps to simplify analysis, boost query efficiency, and better reflect real-world relations.

3.3.3 Node Embedding

Each node's description is embedded into a vector $z \in \mathbb{R}^d$ by *embed* and normalized such that $\|z\|_2 = 1$. The distance between two nodes v_i and v_j is computed via L2 norm $dist(v_i, v_j) = \|z_i - z_j\|_2$. *embed* does not independently predict entities or relations. Rather, embeddings serve a crucial and nuanced role beyond mere operator usage. They facilitate downstream tasks such as retrieval, significantly enhancing performance in systems like Retrieval-Augmented Generation (RAG).

3.3.4 Entity Deduplication

To eliminate redundant representations, we perform entity deduplication (*dedup*). The procedure is as follows. Compute the enhanced embedding z for each entity v and assign its local role r. Initially, each entity forms its own equivalence class. For each entity, retrieve its 20 nearest neighbors (using a FAISS vector search). Retain only pairs (v, v') satisfying $dist(v, v') < threshold_{dedup}$ and $r_v = r_{v'}$. Sort candidate pairs by increasing distance and query the LLM to decide if they represent the same entity. If so, merge their equivalence classes using a union-find (disjoint set) algorithm.

3.3.5 Edge Prediction

We observed that two entities, u and v, are more likely to be connected if they exhibit higher similarity, share more common neighbors, and have more common ancestors (validated by domain experts in 4.1). Based on this insight, we propose a scoring function that integrates three key factors:

$$score = \alpha \cdot \cos(z_u, z_v) + \beta \cdot AA + \gamma \cdot CA. \quad (1)$$

Here, $\cos(z_u, z_v)$ measures the semantic similarity between the vector representations of u and v, AA represents the number of common neighbors, CA denotes the number of common ancestors (see in 3.1.3,3.1.4), and α, β, and γ are weighting coefficients. We set the similarity coefficient $\alpha = 0.6$.

The entity pairs are sorted in descending order based on the computed score, which are further evaluated by LLMs in line with their priority, according to the expected number of edges to be added, denoted as Δe. If the LLM assesses the relation strength as sufficiently strong—i.e., exceeding $threshold_{stren}$, an edge is added. Improving graph connectivity is achieved through a two-stage edge prediction process that combines structural and semantic cues (*pred*).

Stage 1: Initialize Connectivity. At the beginning, the graph exhibits weak connectivity, with few common neighbors among entities. To address this, we initially set $\beta = 0$ and $\gamma = 0.4$. Then the scoring function combined with LLM evaluation is used to add some preliminary edges, thereby enhancing the overall connectivity of entities.

Stage 2: Edge Completion. After the initial stage, as more common neighbors emerge, we adjust the weights to $\beta = 0.3$ and $\gamma = 0.1$. Then the same procedure is applied to establish additional edges. This stage focuses on entity pairs that span across the minimal structural units (i.e., entities not in the same subsection). Otherwise edges in the same subsection typically receive higher scores and are preferentially added, but many of them might have already been extracted during the initial construction.

3.3.6 Structure Integration

Since knowledge-intensive domains are continuously evolving, our framework Tree-KG supports incrementally integrating new textual sources while preserving structural coherence by *merge*.

For unstructured texts, we extract entities and relations and apply operations (conv, aggr, embed) to create subgraph G'. Overlapping entities between graphs G and G' are merged based on embedding similarity, retaining original subsection catalog node labels. New entities in merged components are labeled using graph coloring to minimize cross-subsection edges. Isolated components are linked to the nearest subsection catalog node based on embedding centers.

For structured texts, we generate and merge hierarchical subgraphs and catalog trees recursively by embedding similarity or unique identifiers. Then the same operations are conducted as the unstructured texts above.

4 Experiments

In this section, we present a comprehensive experimental evaluation of our Tree-KG approach. We first provide an overall description of the experimental setup, then detail the comparative experiments, and finally report our ablation studies.

4.1 Experimental Setup

Datasets. The diverse datasets utilized in this study stems from a comprehensive initiative within the AI4EDU project at our institute. We compiled a large pre-collected corpus of approximately 69,000 professional materials, including textbooks, lecture notes, and academic papers, spanning around 100 subfields and encompassing 9.95 million knowledge points. These materials were sourced from various departments across the institution, providing a rich foundation for constructing KGs in diverse domains.

We ensured robustness and consistency through a structured, course-level annotation process: annotators received detailed guidelines and training; multiple TAs independently annotated data, with discrepancies resolved via instructor-led consensus; and we assessed inter-annotator agreement (IAA) using Fleiss' Kappa on randomly selected subsets.

We focus on constructing KGs for three primary domains: Physics (Electromagnetism, Optics, and Quantum Physics; hereafter collectively referred to as Physics), Digital Electronics, and Educational Psychology. For the Physics domain, we developed two expert-annotated KGs: one derived from comprehensive expert knowledge covering the entire Physics field (Domain-Annotated), and the other manually extracted specifically from textbooks (Text-Annotated). Both reference KGs are structured as trees with clearly defined spatial topologies. In addition, we leveraged the knowledge point lists provided in the textbook appendices (Appendix-List). Together, these three sources serve as ground truth references to further validate and guide the KG construction process.

Baseline Methods. We compare Tree-KG (T) against four LLM-based KG construction baselines: GraphRAG (G) (Edge et al., 2024), iText2KG (I) (Lairgi et al., 2024), LangChain (L) (lan), and AutoKG (A) (Zhu et al., 2023). GraphRAG, a RAG framework, is used here only for LLM-based entity/relation extraction, summarization, and basic graph construction; it lacks Tree-KG's iterative, hierarchical, and reasoning-based refinements. iText2KG relies on manually defined schemas but lacks iterative expansion or alignment. LangChain applies template-based graph transformations with no deeper semantic processing. AutoKG employs multi-agent interactions for KG construction but may lack structure in domain-specific contexts. We used GraphRAG v2.1.0, iText2KG v0.0.7, LangChain v0.3.0, AutoKG (GitHub), and Python 3.11. SAC-KG (Chen et al., 2024) was not included due to unavailability.

All methods were evaluated on SiliconFlow with DeepSeek-V3 (et al., 2025) (2 RMB/M input tokens, 8 RMB/M output), while Tree-KG ablation used GLM-4-Air (et al., 2024) (1 RMB/million tokens, now 0.5 RMB/M).

Evaluation strategies. Due to the scarcity of high-quality annotated KGs, we employ hybrid evaluation strategies:

(1) Ground Truth. We treat the three sources—the Domain-Annotated KG, the Text-Annotated KG,

Method	Domain-Annotated					Text-Annotated					Appendix-List			LLM	
	ER↑	PC↑	F1↑	MEC↑	MED↓	ER↑	PC↑	F1↑	MEC↑	MED↓	ER↑	PC↑	F1↑	ES↑	RS↑
Tree-KG	**0.67**	0.34	**0.67**	**0.60**	0.46	**0.78**	0.84	**0.81**	**0.65**	**0.44**	0.74	0.71	**0.72**	**0.98**	**7.87**
GraphRAG	0.53	0.19	0.53	0.38	0.54	0.68	0.51	0.58	0.34	0.52	**0.77**	0.50	0.60	0.88	7.71
iText2KG	0.46	0.30	0.46	0.37	0.48	0.52	0.84	0.65	0.36	0.49	0.49	0.69	0.58	0.93	7.62
LangChain	0.45	**0.38**	0.45	0.18	**0.43**	0.45	**0.89**	0.59	0.13	0.55	0.39	**0.84**	0.53	0.91	7.40
AutoKG	0.46	0.34	0.46	0.03	–	0.50	0.60	0.55	0.04	–	0.45	0.53	0.49	0.89	7.79

Table 1: Ground truth and LLM evaluation on Physics.

Method	Physics					Digital Electronic					Educational Psychology				
	T	G	I	L	A	T	G	I	L	A	T	G	I	L	A
T	–	**0.52**	0.81	**0.84**	0.62	–	**0.55**	**0.82**	0.58	**0.56**	–	**0.32**	0.75	0.71	0.54
G	**0.74**	–	**0.82**	0.80	**0.68**	0.52	–	0.74	0.52	0.48	**0.63**	–	**0.77**	**0.83**	**0.58**
I	0.54	0.36	–	0.63	0.46	**0.54**	0.36	–	**0.63**	0.46	0.50	0.21	–	0.54	0.37
L	0.44	0.29	0.44	–	0.36	0.40	0.32	0.47	–	0.31	0.37	0.19	0.39	–	0.30
A	0.49	0.37	0.52	0.58	–	0.49	0.39	0.64	0.45	–	0.56	0.26	0.56	0.58	–

Table 2: Entity Recall (ER) of mutual evaluation. The domain columns represent KGs constructed by the methods as the ground truth, while the rows indicate the methods to evaluate. For instance, "T-G" denotes that we evaluate method Tree-KG on GraphRAG KG entity sets in metric ER.

and the Appendix-List—as the ground truth references for evaluating automatically constructed KGs from Physics.

(2) Mutual Evaluation. We performed a 5×5 mutual evaluation across the three domains. Entities extracted by one method serve as the ground truth for another method.

(3) LLM-Based Evaluation. We employ LLMs with few-shot prompting to efficiently and consistently assess the quality of the KGs, with their accuracy validated by (Chen et al., 2024).

Metrics. We define metrics (a-h) applied to the three strategies to quantitatively assess the the automatically constructed KG $G = (V, E)$. Assume that a ground truth graph $G_{gt}(V_{gt}, E_{gt})$ is available.

(a) Entity Recall (ER). We first embed entity names from both G_{gt} and G. For each entity in G_{gt}, we retrieve the top-5 most similar entities from G, and an LLM selects the final mapping from these candidates. $\text{ER} = \frac{\#\text{ mapped entities}}{|V_{gt}|}$.

(b) Precision (PC). Similarly, we retrieve each predicted entity in G from G_{gt} using the same procedure. $\text{PC} = \frac{\#\text{ mapped entities}}{|V|}$.

(c) F1 Score (F1). Harmonic mean of recall and precision: $\text{F1} = 2 \times \frac{\text{ER} \times \text{PC}}{\text{ER} + \text{PC}}$.

(d) Mapping-based Edge Connectivity (MEC). For each edge in G_{gt}, we identify the corresponding entity pair in G by applying the above mapping process to both source and target entities. Note that the mapped entities need not be immediate neighbors; they are considered connected if there exists a path between them in G.

$\text{MEC} = \frac{\#\text{ mapped entity pairs connected in } G}{|E_{gt}|}$.

(e) Mapping-based Edge Distance (MED).

$$\text{MED} = \frac{1}{|E_{gt}|} \sum_{(u,v) \in E_{gt}} \frac{d_G(u, v)}{\overline{d_G}},$$

where $d_G(u, v)$ denotes the shortest path distance between the mapped entity pair (u, v) in G, and $\overline{d_G}$ is the average shortest path distance in G. In the ground truth, adjacent entities have strong logical links. Thus, a knowledge graph should keep logically adjacent entities close, i.e., $\text{dist}(u, v) \propto \text{dist}(u', v')$ (lower values indicate better preservation). Also, different methods can yield graphs with varying diameters and connectivity, necessitating normalization.

In the absence of a ground truth graph, we evaluate G using LLMs to score: for each entity, specificity (0 or 1) and completeness (0 to 10); for each relation, strength (0 to 10).

(f) Entity Specificity (ES). Mean entity domain relevance: $\text{ES} = \frac{\sum \text{specificity}}{|V|}$.

(g) Entity Completeness (EC). Mean completeness of entity descriptions: $\text{EC} = \frac{\sum \text{completeness}}{|V|}$.

(h) Relation Strength (RS). Mean closeness and clarity of underlying relations: $\text{RS} = \frac{\sum \text{strength}}{|E|}$.

This multi-faceted evaluation approach validates the robustness and accuracy of TREE-KG

4.2 Main Results

Table 1 and Table 2 are comparisons of our Tree-KG with other methods by ground truth, LLM-based evaluation, and mutual evaluation. Overall, experiments demonstrate that Tree-KG consistently

outperforms the other competing approaches across multiple metrics.

Firstly, whether against ground truth or via mutual evaluation, Tree-KG achieves entity recall (ER) comparable to or exceeding GraphRAG, while far surpassing other methods. Specifically, Tree-KG outperforms all the other methods, with a 14% improvement over the 2nd-best on domain-annotation. In mutual evaluation, it performs similarly to or better than GraphRAG. This indicates superior sufficiency of Tree-KG in entity extraction. For precision (PC) on the ground truth, Tree-KG consistently ranks second among all methods. While LangChain achieves the highest precision due to its smaller extracted graphs—with fewer nodes and edges—this comes at the cost of significantly lower ER, revealing that its extraction is incomplete and lacks sufficient coverage. In contrast, when considering the F1 score, which reflects the overall balance between recall and precision, Tree-KG clearly outperforms all competing methods, achieving a 12–16% improvement over the 2nd-best on the ground truth. Notably, on the Text-Annotated dataset, Tree-KG reaches 0.81 F1 score, underscoring its ability to extract high-quality information from the original text. This substantial gain highlights Tree-KG's superior balance of coverage and accuracy, making it the most effective approach.

Second, our approach exhibits the highest mapping-based edge connectivity (MEC) (1.6x-20x) and the lowest normalized mapping-based edge distance (MED) (0.8x-1.1x) on others, indicating the best structural alignment with the original graph. These results suggest that Tree-KG better supports downstream reasoning tasks and meets connectivity requirements.

Moreover, Tree-KG leads in both entity specificity (ES) and relation strength (RS), indicating that the extracted entities are not only high-quality and domain-specific, but also supported by high-quality relational edges.

Method	#Tokens (M)	Cost (RMB)
T	6.1 (**1.5 for initial construction** + 4.6 for expansion)	18 (**4.5** + 13.5)
G	4.0	12
I	2.3	7.4
L	15.0	28
A	7.2	18

Table 3: Token usage and cost across methods.

Token/API Cost. We compare the token and API costs of Tree-KG against baseline methods using the DeepSeek-V3 API on the Physics domain, as summarized in Table 3.

summary	ES↑	RS↑	$\|V\|$	$\|E\|$
with	0.94 ± 0.01	7.10 ± 0.04	1301 ± 5	930 ± 12
without	0.86 ± 0.01	6.57 ± 0.08	1530 ± 150	1050 ± 50

Table 4: KG quality with vs. without summary.

stage	EC↑	RS↑
0	6.92 ± 0.08	7.10 ± 0.04
1	9.30 ± 0.05	7.79 ± 0.05
2	9.39 ± 0.02	7.86 ± 0.02
3	9.50 ± 0.04	7.89 ± 0.04

Table 5: Impact of multiple contextual-based convolution steps on the KG quality.

Table 3 shows that GraphRAG and the initial phase of Tree-KG both build hierarchical structures. However, GraphRAG uses a bottom-up approach (via the Hierarchical Leiden Algorithm and heavy summarization), consuming many tokens, while Tree-KG leverages textbook structures to build top-down without token cost. Tree-KG's bottom-up phase only summarizes text, making its initial construction far more token-efficient than GraphRAG.

Overall, Tree-KG's initial phase incurs lower costs, and its iterative expansion is optional, allowing flexibility based on budget or application needs. While some methods (e.g., iText2KG) show slightly lower costs, their extraction quality is limited. Tree-KG offers a balanced trade-off between cost and performance, achieving strong results with reasonable resource use.

4.3 Ablation

We perform systematic ablation experiments to validate the effectiveness of Tree-KG designs.

4.3.1 Summarization

We compare the performance of Tree-KG with and without the preliminary summary step. Table 4 shows that summary improves ES by 9% and RS by 8% compared to without summary, enhancing the quality of extracted entities and relations and reducing irrelevant or weak associations. We can also observe greater numerical stability in the number of entities and edges, confirming its robustness.

4.3.2 Contextual-based Convolution

We evaluated the impact of multiple contextual-based convolution steps on the quality of entities and relations. Table 5 shows that one *conv* step yields substantial improvements (EC +34%, RS +10%). Consistent with the convergence behavior in traditional graph convolution which typically converge within 2–3 iterations, our *conv* also con-

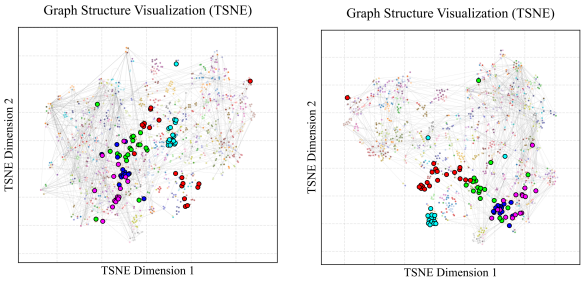


Figure 3: Entities in the same subsection become compact after contextual-based convolution.

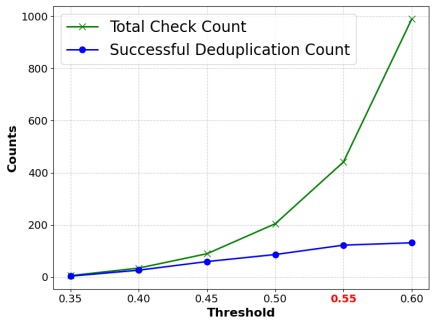


Figure 4: Entity deduplication threshold.

verges rapidly. Further steps offer only about 1% gain while significantly increasing token usage, indicating that a single step is generally sufficient unless the graph structure changes.

We also select entities from the same minimal chapter (i.e., subsection in our case) and visualize their description embeddings using t-SNE before and after *conv*. Identical colors represent the same subsection. Figure 3 shows that *conv* clusters embeddings of entities from the same subsection, indicating an alignment of textual descriptions with the graph structure, which supports subsequent entity deduplication.

4.3.3 Entity Deduplication

We can see from Figure 4 that, as the value of $threshold_{dedup}$ increases, both the total check count and the successful deduplication count also increase. However, raising the threshold from 0.55 to 0.6 yields only a 7% increase in successful deduplications, while the total check count increases by 124%. Thus, a $threshold_{dedup}$; of 0.55 is adopted for optimal accuracy and efficiency.

4.3.4 Edge Prediction

As described in Section 3.3.5, the weights α, β, and γ correspond to three key edge prediction goals: semantic relevance (cosine similarity), structural similarity (Adamic-Adar Score, AA), and hierarchical proximity (Common Ancestors, CA). We set $\alpha > \beta > \gamma$, as cosine similarity provides the richest cues, followed by AA, with CA offering the least. We performed a hyperparameter sensitivity analysis by testing all 66 combinations of α, β, and γ summing to 1 (at 0.1 steps). For each, we computed the combined edge prediction score (Formula 1) across ∼336K candidate pairs, ranked them, and evaluated the top 1% (∼3,000 pairs) based on their average percentile under each metric. This mean percentile indicates how well the combined score balances the three goals. As shown in Figure 5, our original setting ($\alpha = 0.6$, $\beta = 0.3$, $\gamma = 0.1$) achieved a strong mean percentile of 82.86%. Interestingly, a slightly different setting ($\alpha = 0.8$, $\beta = 0.1$, $\gamma = 0.1$) performed even better (83.14%), suggesting that increasing the weight on semantic similarity further improves predictions.

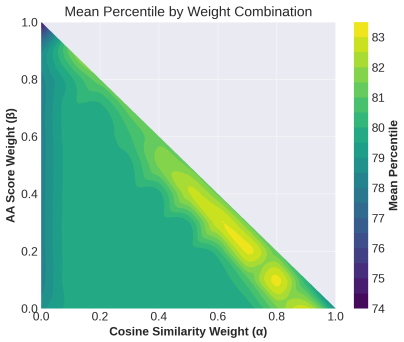


Figure 5: The mean percentile of top 1% entity pairs in all hyperparameter settings of edge prediction score.

5 Conclusion

In this paper, we propose Tree-KG, an expandable framework for constructing and iteratively expanding KGs in knowledge-intensive domains. By integrating structured document cues with advanced semantic techniques, our method effectively builds an explicit, tree-like hierarchical graph and uncovers hidden relationships through iterative operator-driven expansion. Experiments demonstrate that Tree-KG consistently surpasses competing methods, achieving the highest F1 scores (12–16% above the second-best), with notable performance (F1 0.81) on the Text-Annotated dataset, highlighting its effectiveness in extracting high-quality information from source texts. Additionally, Tree-KG provides superior structural alignment, domain-specific extraction, and cost-efficiency, delivering robust results with reduced token usage and adaptable, resource-conscious deployment.

Limitations

Although our Tree-KG performs well in KG construction, the current method relies on inherent knowledge, which may not perform well for certain downstream tasks (e.g., inference). We plan to optimize our design by introducing task-specific features to construct knowledge graphs that better support these more complex tasks.

Acknowledgments

This work was supported by the Postdoctoral Fellowship Program of CPSF (No.GZC20231295), the National Key Research and Development Program of China (No.2024YFC3606800), and DCST Student Academic Training Program. We would like to thank all the anonymous reviewers for their insightful comments.

References

https://www.langchain.com/. 2025/2/7.

Lada A. Adamic and Eytan Adar. 2003. Friends and neighbors on the web. *Soc. Networks*, 25:211–230.

Michele Banko, Michael J. Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. 2007. Open information extraction from the web. In *CACM*.

Michele Banko and Oren Etzioni. 2008. The tradeoffs between open and traditional relation extraction. In *Annual Meeting of the Association for Computational Linguistics*.

Salvatore M. Carta, Alessandro Giuliani, Lee Cecilia piano, Alessandro Sebastian Podda, Livio Pompianu, and Sandro Gabriele Tiddia. 2023. Iterative zero-shot llm prompting for knowledge graph construction. *ArXiv*, abs/2307.01128.

Payal Chandak, Kexin Huang, and Marinka Zitnik. 2022. Building a knowledge graph to enable precision medicine. *Scientific Data*, 10.

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 *Annual Meeting of the Association for Computational Linguistics*.

Cec´ılia F. V. Couto and Nelson F. F. Ebecken. 2024. Data exploration: large language models in the construction of knowledge graphs. *Ibero-Latin American Congress on Computational Methods in Engineering (CILAMCE)*.

Fu-Rong Dang, Jintao Tang, Kunyuan Pang, Ting Wang, Shasha Li, and Xiao Li. 2021. Constructing an educational knowledge graph with concepts linked to wikipedia. *Journal of Computer Science and Technology*, 36:1200 – 1211.

Linyi Ding, Sizhe Zhou, Jinfeng Xiao, and Jiawei Han. 2024. Automated construction of theme-specific knowledge graphs. *ArXiv*, abs/2404.19146.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, and Jonathan Larson. 2024. From local to global: A graph rag approach to query-focused summarization. *ArXiv*, abs/2404.16130.

DeepSeek-AI et al. 2025. Deepseek-v3 technical report. *Preprint*, arXiv:2412.19437.

Team GLM et al. 2024. Chatglm: A family of large language models from glm-130b to glm-4 all tools. *Preprint*, arXiv:2406.12793.

Xiaohan Feng, Xixin Wu, and Helen M. Meng. 2024. Ontology-grounded automatic knowledge graph construction by llm under wikidata schema. In *HI-AI@KDD*.

Luis Galárraga, Christina Teflioudi, Katja Hose, and Fabian M. Suchanek. 2013. Amie: association rule mining under incomplete evidence in ontological knowledge bases. *Proceedings of the 22nd international conference on World Wide Web*.

Jun-Tae Kim and Dan I. Moldovan. 1993. Acquisition of semantic patterns for information extraction from corpora. *Proceedings of 9th IEEE Conference on Artificial Intelligence for Applications*, pages 171–176.

Yassir Lairgi, Ludovic Moncla, R'emy Cazabet, Khalid Benabdeslem, and Pierre Cl'eau. 2024. itext2kg: Incremental knowledge graphs construction using large language models. In *WISE*.

Olivera Marjanovic. 2011. Improving knowledge-intensive health care processes beyond efficiency. In *International Conference on Interaction Sciences*.

Mike D. Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Annual Meeting of the Association for Computational Linguistics*.

Stephen H. Muggleton. 1997. Inductive logic programming: 6th international workshop, ilp-96, stockholm, sweden, august 26-28, 1996, selected papers.

Alberto Santos, Ana R. Colaço, Annelaura Bach Nielsen, Lili Niu, Maximilian T. Strauss, Philipp E. Geyer, Fabian Coscia, Nicolai J. Wewer Albrechtsen, Filip Mundt, Lars Juhl Jensen, and Matthias Mann. 2022. A knowledge graph to interpret clinical proteomics data. *Nature Biotechnology*, 40:692 – 702.

Xiaorui Su, Yibo Wang, Shanghua Gao, Xiaolong Liu, Valentina Giunchiglia, Djork-Arné Clevert, and Marinka Zitnik. 2024. Knowledge graph based agent for complex, knowledge-intensive qa in medicine. *ArXiv*, abs/2410.04660.

Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: A core of semantic knowledge unifying wordnet and wikipedia. In *The Web Conference*.

Chenguang Wang, Xiao Liu, Zui Chen, Haoyun Hong, Jie Tang, and Dawn Xiaodong Song. 2021. Zero-shot information extraction as a unified text-to-triple translation. *ArXiv*, abs/2109.11171.

Fangyuan Xu, Kyle Lo, Luca Soldaini, Bailey Kuehl, Eunsol Choi, and David Wadden. 2024. Kiwi: A dataset of knowledge-intensive writing instructions for answering research questions. *ArXiv*, abs/2403.03866.

Lilong Xue, Dan Zhang, Yuxiao Dong, and Jie Tang. 2024. Autore: Document-level relation extraction with large language models. *ArXiv*, abs/2403.14888.

Chenwei Yan, Xiangling Fu, Xinxin You, Ji Wu, and Xien Liu. 2024. Graph-based cross-granularity message passing on knowledge-intensive text. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 32:4409–4419.

Alexander Yates, Michele Banko, Matthew Broadhead, Michael J. Cafarella, Oren Etzioni, and Stephen Soderland. 2007. Textrunner: Open information extraction on the web. In *North American Chapter of the Association for Computational Linguistics*.

Junlang Zhan and Hai Zhao. 2019. Span model for open information extraction on accurate corpus. In *AAAI Conference on Artificial Intelligence*.

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

Tianyu Zhao, Chengliang Chai, Yuyu Luo, Jianhua Feng, Y. F. Huang, Songfan Yang, Haitao Yuan, Haoda Li, Kaiyu Li, Fu Zhu, and Kang Pan. 2019. Towards automatic mathematical exercise solving. *Data Science and Engineering*, 4:179 – 192.

Yilun Zhao, Hongjun Liu, Yitao Long, Rui Zhang, Chen Zhao, and Arman Cohan. 2023. Financemath: Knowledge-intensive math reasoning in finance domains.

Yuqi Zhu, Xiaohan Wang, Jing Chen, Shuofei Qiao, Yixin Ou, Yunzhi Yao, Shumin Deng, Huajun Chen, and Ningyu Zhang. 2023. Llms for knowledge graph construction and reasoning: Recent capabilities and future opportunities. *World Wide Web (WWW)*, 27:58.

A More Results

A.1 Comparative Experiments

We also conducted comparative evaluations based on GLM and GPT. We use GLM-4-air for Tree-KG and LLM-based evaluation, GPT-4o for iText2KG, GPT-4-Turbo-Preview for GraphRAG, GPT-4-Turbo for LangChain, and GPT-4o for AutoKG. We designed a paradigm with low requirements for the backzone model and therefore employed the GLM-4-Air model at a cost of 1 yuan per million tokens. We attempted to use GLM-4-Air for other comparative methods; however, the results were unsatisfactory, and in some cases, the model failed to function correctly. This issue was likely due to the design of the prompt. To accurately reproduce the results reported in the original papers of the comparative methods, we opted not to use the same API but instead employed the default GPT API specified in the respective papers.

Overall, as shown in Table 6 and Table 7, despite our approach Tree-KG based on GLM, which is **100 times more cost-effective**, experiments demonstrate that Tree-KG consistently outperforms the comparative methods that rely on GPT across multiple metrics. This demonstrates that our approach is both **efficient and cost-effective**.

A.2 Edge Prediction

As shown in Table 6, the T+e1000 variant outperforms the others in both relation strength (RS) and normalized mapping-based edge distance (MED). Iterative edge addition increases RS (with a 6% improvement over the baseline T) and reduces MED by 3%, confirming that the priority score effectively enhances edge quality and the structure consistency with ground truth graph. These findings support predicting an additional number of edges equal to the node count (#nodes approximately 1000), though the optimal number may vary with different graph structures.

A.3 Contextual-based Convolution

The PCA visualizations before and after contextual-base convolution in Figure 6 reveals that equivalence classes become more compact after *conv*. A noteworthy side effect of the *conv* process is that it draws entities with similar intrinsic properties closer together while pushing apart those that are fundamentally different, thereby aligning the semantic and logical spaces. This will help the entity-deduplication afterwards.

A.4 Case Study

Figure 7 illustrates certain differences between our method Tree-KG and other methods for the same two texts. In the first text, "proton" serves as an example to explain the core entity of "charge". In the

Method	Domain-Annotated			Text-Annotated			LLM	
	ER↑	MEC↑	MED↓	ER↑	MEC↑	MED↓	ES↑	RS↑
T	**0.58**	0.41	0.38	**0.81**	**0.65**	0.57	**0.94**	7.79
T+e500	**0.58**	**0.43**	0.38	**0.81**	**0.65**	0.55	**0.94**	8.05
T+e1000	**0.58**	**0.43**	**0.37**	**0.81**	**0.65**	0.55	**0.94**	**8.23**
T+e1500	**0.58**	**0.43**	0.39	**0.81**	**0.65**	0.55	**0.94**	8.22
G	0.55	0.20	0.75	0.66	0.18	0.87	0.80	7.10
I	0.42	0.20	0.54	0.45	0.15	**0.53**	0.87	7.23
L	0.26	<0.02	–	0.40	<0.02	–	0.92	7.61
A	0.40	<0.02	–	0.67	<0.02	–	0.64	6.24

Table 6: Expert-annotated and LLM evaluation on Physics. The notation "T+e500" indicates that Tree-KG has been added an extra 500 edges by edge prediction. We merge the results of edge prediction into this table.

Method	Physics					Digital Electronic					Educational Psychology				
	T	G	I	L	A	T	G	I	L	A	T	G	I	L	A
T	–	**0.84**	0.77	0.73	**0.68**	–	0.70	0.86	0.64	**0.65**	–	0.57	**0.71**	**0.61**	0.69
G	**0.77**	–	**0.79**	**0.88**	0.67	**0.73**	–	**0.89**	**0.77**	0.63	**0.73**	–	0.66	0.59	**0.73**
I	0.43	0.67	–	0.55	0.44	0.58	**0.80**	–	0.57	0.39	0.62	0.51	–	0.29	0.56
L	0.44	0.33	0.63	–	0.60	0.49	0.70	0.35	–	0.35	0.59	**0.63**	0.48	–	0.25
A	0.55	0.48	0.34	0.69	–	0.57	0.55	0.57	0.53	–	0.67	0.42	0.50	0.43	–

Table 7: Entity Recall (ER) of mutual evaluation. The domain columns represent KGs constructed by the methods as the ground truth, while the rows indicate the methods to evaluate. For instance, "T-G" denotes that we evaluate method Tree-KG on GraphRAG KG entity sets in metric ER.

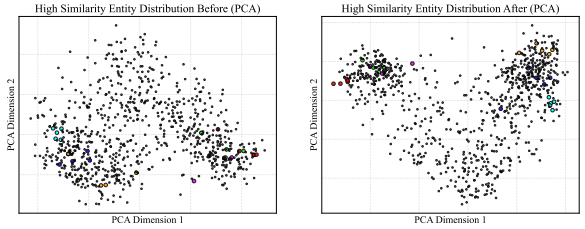
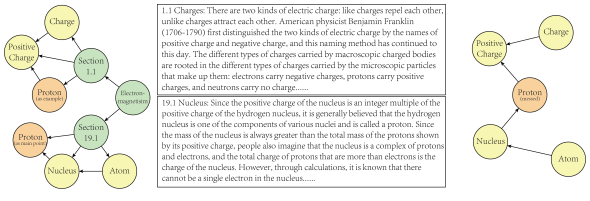


Figure 6: Equivalence classes become more compact after contextual-base convolution.



Our Method Raw Text Other Methods

Figure 7: Case study.

second text, "proton" is the primary introduced entity. Other methods tend to merge these two entities of "proton," whereas our method Tree-KG, through *conv* and *aggr*, ultimately keeps them distinct. It avoids unnecessary branches in downstream tasks, thereby improving efficiency.

A.5 Visualization

A.5.1 Overall KGs

We present an overview of the knowledge graphs generated using different methods (a) Tree-KG, (b) GraphRAG, (c) iText2KG, (d) LangChain and (e) AutoKG, from a physics textbook in Figure 8. For our method, blue represents section nodes, red represents core entities, and yellow represents non-core entities. For the comparison methods, all entities are colored brown.

A.5.2 Tree-KG Illustrations

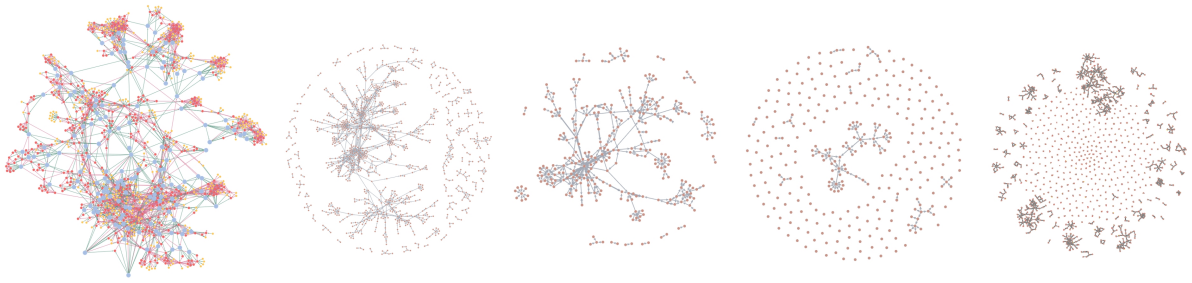
As we can see from Figure 9, (a) shows the extracted hierarchical structure of the chapters, (b) visualizes the entity hierarchy, including both core and non-core entities, (c) illustrates the graph before the aggregation operation, while (d) shows the graph after aggregation. Entities that have been reclassified as non-core are colored yellow (from red), and those that remain core entities are still colored red.

B Prompt

B.1 General Techniques

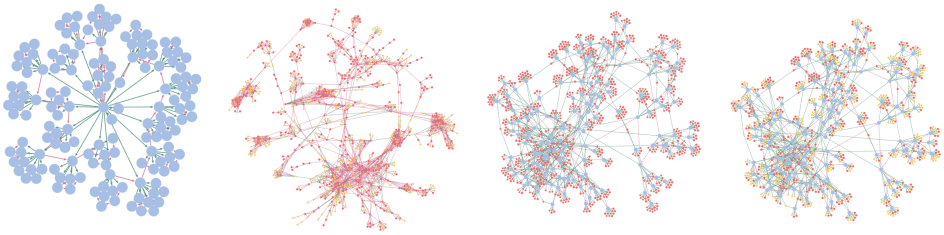
B.1.1 Unified and Clear Structure

- Role: Assume a specific role.



(a) Tree-KG (b) GraphRAG (c) iText2KG (d) LangChain (e) AutoKG

Figure 8: Comparison of overall KGs on Physics.



(a) Sections (b) Entities (c) Before aggregation (d) After aggregation

Figure 9: Partial KGs from Tree-KG.

- Task: Provide clear objectives.
- Constraints: Clearly define standards and requirements.
- Output Template: Use formatted output to increase stability.
- Example: Include few-shot examples to ensure output quality.

B.1.2 Quantification

For prediction, evaluation, and similar tasks, prioritize requesting scores for specific metrics over absolute choices. This helps obtain more precise results.

B.2 Extraction

This section's prompts are primarily used to generate summaries of source texts and asynchronously extract entities and relationships through leaf chapter summaries.

- Summary: Input the original text of a leaf chapter or subchapter summary, output the summary of that chapter.
- Extraction: First input the leaf chapter summary to output all entities; then input the leaf chapter summary and the extracted entity list to output relationships.

B.3 Expansion

These prompts are used by various operators for graph augmentation.

- Convolution: Input an entity's information and all related relationships (or relationships with entities), supplemented by model domain knowledge, to enhance entity descriptions and assess local interactions (or enhance relationship descriptions).
- Aggregation: Input a central entity, adjacent nodes, and relationships. Determine if aggregation to the central entity is needed and output aggregated entity information.
- Deduplication: Input entity pairs with high similarity and overlapping local roles to determine if they represent the same entity.
- Edge Prediction: Input entity pairs with related information and model domain knowledge to determine if a relationship exists, its nature, and strength.

B.4 Evaluation

Prompts for assessing the quality of extracted graphs compared to baselines.

- Entities: Input domain knowledge, chapter overview, and entities with descriptions. Score entity relevance and completeness.
- Relationships: Input chapter overview and relationship triples. Score relationship closeness, logicality, and significance.

B.5 Example

B.5.1 Entity Scoring

- **Role**

  You are an educational knowledge graph expert skilled in evaluating entity extraction quality.

- **Task**

  Given a knowledge domain, chapter overview, and extracted entities, score each entity's specificity and description completeness.

- **Constraints**
  - Score two metrics (0 or 1, 0-10):
    - **Specificity**: 0=irrelevant, 1=highly relevant.
    - **Completeness**: 1=incomplete, 5=basic, 9=comprehensive.
  - Output must be a valid JSON list.

- **Output Template**

```
[
  {
    "id": "Entity ID",
    "specificity": 0 or 1,
    "completency": 0-10
  }
]
```

B.5.2 Entity Extraction

- **Role**

  You are a subject matter expert specializing in entity and relationship extraction.

- **Task**

  Users will provide a chapter summary from a subject-specific textbook. Please extract entities closely related to that subject in JSON format.

- **Constraints**
  - Entities should be concise, specific, and strongly relevant to the subject.
  - Different names for the same entity should be merged into the 'alias' field.
  - Extracted entities should be noun phrases.
  - Output must be valid JSON.

- **Output Template**

```
{
  "entities": [
    {
      "name": "Entity Name",
      "alias": ["Alias 1", "Alias
          2"],
      "type": "Entity Type",
      "raw_content": "Original text
          describing the entity"
    }
  ]
}
```

B.5.3 Relationship Prediction

- **Role**

  You are a relationship extraction expert.

- **Task**

  Given two entities and their information, determine possible relationships or conclude no relationship exists.

- **Constraints**
  - Output must be valid JSON.
  - Relationship types should be concise and meaningful.
  - Relationship descriptions must be detailed and grounded in provided information.

- **Output Template**

```
{
    "is_relevant": true/false,
    "description": "### Association
        Description\n...\n###
        Explanation\n...",
    "type": "Relationship type(s)",
    "strength": 0-10,
    "reason": "Scoring rationale"
}
```