AutoSchemaKG: Autonomous Knowledge Graph Construction through Dynamic Schema Induction from Web-Scale Corpora

Jiaxin Bai[*1], **Wei Fan**[*1], **Qi Hu**[*1], **Qing Zong**[1], **Chunyang Li**[1], **Hong Ting Tsang**[1]
Hongyu Luo[1], **Yauwai Yim**[1], **Haoyu Huang**[2], **Xiao Zhou**[1], **Feng Qin**[1], **Tianshi Zheng**[1]
Xi Peng[3], **Xin Yao**[3], **Huiwen Yang**[3], **Leijie Wu**[3], **Yi Ji**[3]
Gong Zhang[3], **Renhai Chen**[3], **and Yangqiu Song**[1]
[1]CSE, HKUST [2] CSE, CUHK [3] Theory Lab, Huawei
{jbai, wfanag, qhuaf, qzong, cliei, httsangaj, hluoay, ywyimaa, tzhengad
xzhoucs, fqinac}@cse.ust.hk, haoyuhuang@link.cuhk.edu.hk
{pancy.pengxi, yao.xin1, huiwen.yang, leijie.wu1, jiyi13,
nicholas.zhang, chenrenhai}@huawei.com yqsong@cse.ust.hk

Abstract

We present AutoSchemaKG, a framework for fully autonomous knowledge graph construction that eliminates the need for predefined schemas. Our system leverages large language models to simultaneously extract knowledge triples and induce comprehensive schemas directly from text, modeling both entities and events while employing conceptualization to organize instances into semantic categories. Processing over 50 million documents, we construct ATLAS (Automated Triple Linking And Schema induction), a family of knowledge graphs with 900+ million nodes and 5.9 billion edges. This approach outperforms state-of-the-art baselines on multi-hop QA tasks and enhances LLM factuality. Notably, our schema induction achieves 95% semantic alignment with human-crafted schemas with zero manual intervention, demonstrating that billion-scale knowledge graphs with dynamically induced schemas can effectively complement parametric knowledge in large language models[1].

1 Introduction

In the current era of information abundance, transforming vast amounts of unstructured data into structured, machine-readable knowledge remains one of the most significant challenges in artificial intelligence. Knowledge Graphs (KGs) have emerged as the cornerstone technology for this transformation (Zhao et al., 2024), providing the semantic backbone for applications ranging from search engines and question answering (Wu et al., 2024; Chen et al., 2024c; Zong et al., 2024; Sun et al., 2024b) to recommendation systems (Lyu et al., 2024) and complex reasoning tasks (Li et al., 2024b). Yet despite their critical importance, current KG construction approaches remain hampered by an inherent paradox: they require predefined schemas created by domain experts, which fundamentally limits their scalability, adaptability, and domain coverage.

We present AutoSchemaKG, a paradigm-shifting framework that breaks this dependency by enabling autonomous knowledge graph construction without predefined schemas (Ye et al., 2023), as shown in Figure 1. Our approach leverages the semantic understanding capabilities of large language models to simultaneously extract knowledge triples and dynamically induce schemas directly from text, eliminating this manual bottleneck in KG development (Zhang and Soh, 2024; Li et al., 2024a; Wang et al., 2025).

Our framework distinguishes itself by modeling events alongside entities (Zhang et al., 2020, 2022), recognizing that real-world knowledge is dynamic rather than static (Tan et al., 2024). By treating events as basic semantic units in the graph, AutoSchemaKG captures temporal relationships, causality, and procedural knowledge missed by entity-only graphs. This approach aligns with recent advancements in proposition-based text decomposition (Hoyle et al., 2023; Jhamtani et al., 2024; Chen et al., 2024b), which demonstrate that by doing text composition, the resulting semantic units can provide better text representation from implicit meanings, meanwhile providing higher information density and retrieval accuracy than traditional passage-level representations. On the top of extracted events from text decomposition, our event-centric modeling is further supported by discourse relation recognition research (Chan et al.,

[*] Equal contribution.
[1]https://github.com/HKUST-KnowComp/AutoSchemaKG

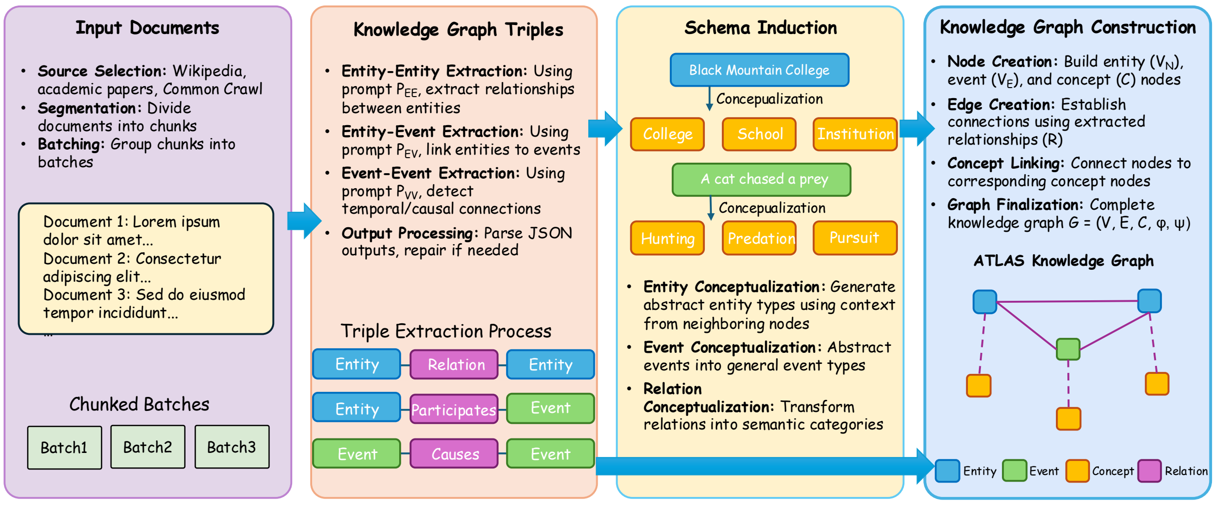
Input Documents
Source Selection: Wikipedia, academic papers, Common Crawl
Segmentation: Divide documents into chunks
Batching: Group chunks into batches
Document 1: Lorem ipsum dolor sit amet...
Document 2: Consectetur adipiscing elit...
Document 3: Sed do eiusmod tempor incididunt...
...
Chunked Batches
Batch1
Batch2
Batch3
Knowledge Graph Triples
Entity-Entity Extraction: Using prompt P_EE, extract relationships between entities
Entity-Event Extraction: Using prompt P_EV, link entities to events
Event-Event Extraction: Using prompt P_VV, detect temporal/causal connections
Output Processing: Parse JSON outputs, repair if needed
Triple Extraction Process
Entity
Relation
Entity
Entity
Participates
Event
Event
Causes
Event
Schema Induction
Black Mountain College
Concepualization
College
School
Institution
A cat chased a prey
Concepualization
Hunting
Predation
Pursuit
Entity Conceptualization: Generate abstract entity types using context from neighboring nodes
Event Conceptualization: Abstract events into general event types
Relation Conceptualization: Transform relations into semantic categories
Knowledge Graph Construction
Node Creation: Build entity (V_N), event (V_E), and concept (C) nodes
Edge Creation: Establish connections using extracted relationships (R)
Concept Linking: Connect nodes to corresponding concept nodes
Graph Finalization: Complete knowledge graph G = (V, E, C, φ, ψ)
ATLAS Knowledge Graph
Entity
Event
Concept
Relation

Figure 1: This figure illustrates the AutoSchemaKG pipeline for autonomous knowledge graph construction through four phases: (1) Input Processing: documents are filtered, segmented, and batched; (2) Triple Extraction: relationships between entities and events are extracted using LLM prompts; (3) Schema Induction: elements are conceptualized into abstract categories without predefined schemas; and (4) Knowledge Graph Construction: triples and schema are integrated into the ATLAS knowledge graph with entity nodes (blue), event nodes (green), concept nodes (orange), and relation edges (purple).

2023, 2024), which emphasizes the importance of understanding implicit temporal and causal relationships between statements—connections that entity-only representations cannot adequately capture. This discourse-aware approach enables AutoSchemaKG to preserve hierarchical path information similar to the tree-structured discourse relations in natural text. Experiments confirm that with event information, AutoSchemaKG preserves over 90% of original passage content versus just 70% for entity information alone, demonstrating the crucial role of events in knowledge representation.

Central to our innovation is the conceptualization (Wang et al., 2023c; Bai et al., 2024; Wang et al., 2023b; He et al., 2024; Wang et al., 2024a,c) process that drives schema induction. Rather than simply extracting triples, we employ a sophisticated abstraction mechanism that generalizes specific entities, events, and relations into broader conceptual categories. This conceptualization serves multiple critical functions: it creates semantic bridges between seemingly disparate information, enables zero-shot inferencing across domains, reduces sparsity in KGs, and provides hierarchical organization that supports both specific and abstract reasoning (Wang et al., 2024d,b). Our approach transforms the traditional static schema into a dynamic, multi-level conceptual framework that adapts to new domains without predefined ontologies. This conceptualization layer represents a fundamental advancement, transforming a collection of disconnected facts into a coherent knowledge ecosystem with emergent reasoning capabilities.

By applying our framework to the Dolma 1.7 (Soldaini et al., 2024) pretraining corpus across three diverse subsets, English Wikipedia, paper abstracts from Semantic Scholar, and 3% of Common Crawl data, we construct the ATLAS family of knowledge graphs (ATLAS-Wiki, ATLAS-Pes2o, and ATLAS-CC). Collectively, these knowledge graphs comprise over 900 million nodes connected by 5.9 billion edges, containing billions of facts that are comparable in scale to the parametric knowledge stored in large language models. This scale is crucial, as we demonstrate that knowledge graphs must reach a critical mass of billions of facts to effectively compete with and complement the parametric knowledge in contemporary LLMs.

Our work addresses a fundamental research question: can retrieval-augmented generation with comparable-sized knowledge graphs enhance LLM performance even when the retrieval corpus is drawn from the same sources as the model's pretraining data? Through comprehensive evaluations on general benchmarks, we demonstrate that properly structured knowledge representations offer advantages over traditional text-based retrieval even in these scenarios. Most significantly, our schema induction process achieves 92% semantic alignment with human-crafted schemas with zero manual intervention, demonstrating that automated schema generation can match expert quality while dramatically improving construction efficiency.

The power of AutoSchemaKG extends beyond its construction methodology to deliver substantial performance improvements in downstream applications. In rigorous evaluations, our approach outperforms state-of-the-art baselines by 12-18% on multi-hop question answering tasks (Trivedi et al., 2022; Yang et al., 2018; Ho et al., 2020) and enhances large language model factuality by up to 9% (Chen et al., 2023). Moreover, we found that our constructed knowledge graph is helpful for Llama3.1 7B models on general reasoning task on various domains that intensively requeries background knowledge, including Global Facts, History, Law, Religion, Philosophy and Ethics, Medicine and Health, and Social Sciences. These gains stem from our system's ability to create richer semantic representations through the integration of entities, events, and their conceptual abstractions—enabling better reasoning over complex information across different domains and data sources. Our key contributions include:

- We develop an entity-event-concept extraction framework that captures not only traditional entity relationships but also complex event structures and their conceptual categorizations, creating a multi-dimensional knowledge representation.
- We apply our efficient knowledge extraction and integration approach to web-scale data, processing billions of triples while maintaining semantic consistency. The resulting ATLAS-family knowledge graphs are, to the best of our knowledge, both the largest automatically constructed knowledge graphs and the largest graph-based Retrieval Augmented Generation (Graph RAG) datasets available.
- We build a retrieval augmented generation pipeline on the billion-scale ATLAS KGs, demonstrating AutoSchemaKG's effectiveness across diverse domains without domain-specific customization, establishing a truly general-purpose knowledge acquisition framework.

AutoSchemaKG represents a fundamental rethinking of knowledge graph construction, transforming what was once a heavily supervised process requiring significant domain expertise into a fully automated pipeline. This advancement not only accelerates KG development but also dramatically expands the potential application domains for knowledge-intensive AI systems.

2 Problem Definition

We formally outline the tasks involved in automatically constructing knowledge graphs. We begin by providing a precise definition of a knowledge graph equipped with a conceptual schema.

Definition 1 (Knowledge Graph with Conceptual Schema)**.** Consider a knowledge graph denoted as $G = (V, E, C, \phi, \psi)$, where: $V = V_E \cup V_N$ represents the collection of nodes, with V_E as the set of event nodes, V_N as the set of entity nodes, and $V_E \cap V_N = \emptyset$. $E \subseteq V \times V \times R$ defines the set of edges, where R denotes relation types. Edges may connect entity-entity, entity-event, or event-event nodes. C is the set of conceptual categories. $\phi : V \to \mathcal{P}(C)$ assigns each node a subset of concepts, where $\phi(v) \subseteq C$ for every $v \in V$. $\psi : R \to \mathcal{P}(C)$ links each relation type to a subset of concepts, where $\psi(r) \subseteq C$ for every $r \in R$. $\mathcal{P}(C)$ denotes the power set of C, encompassing all possible subsets. Additional constraints: $\forall v \in V : \phi(v) \neq \emptyset$ and $\forall r \in R : \psi(r) \neq \emptyset$.

3 AutoSchemaKG Framework

In this section, we elaborate on the process of fully automating knowledge graph construction.

3.1 Triple Extraction

Our approach employs a multi-phase pipeline using Large Language Models to convert unstructured text into knowledge triples from the Dolma corpus (Soldaini et al., 2024). This pipeline extracts Entity-Entity, Entity-Event, and Event-Event relationships through three sequential stages. We preprocess texts by filtering for English language and segmenting documents that exceed token limits. The segmented texts are grouped into processing batches. Stage 1 extracts Entity-Entity relationships using a system prompt P_{EE} that instructs the LLM to detect entities and their interrelations. The output is parsed into triples (e_1, r, e_2), where $e_1, e_2 \in V_N$ are entity nodes and $r \in R$ is a relation type. Stage 2 identifies Entity-Event relationships with prompt P_{EV}, producing triples (e, r, v) or (v, r, e), where $e \in V_N$, $v \in V_E$, and $r \in R$. Stage 3 targets Event-Event relationships with prompt P_{VV}, generating triples (v_1, r, v_2), where $v_1, v_2 \in V_E$ and $r \in R$. The pipeline supports various LLMs with

	Question Answering Corpora			Pre-training Corpora		
	MuSiQue	2WikiQA	HotpotQA	ATLAS-Wiki	ATLAS-Pes2o	ATLAS-CC
# Text Chunks	11,656	6,119	9,221	9.599M	7.918M	35.040M
# Entities	108,582	48,782	95,686	70.104M	75.857M	241.061M
# Events	99,747	50,910	82,833	165.717M	92.636M	696.195M
# Concepts	37,414	19,830	32,410	8.091M	5.895M	31.070M
# Nodes	**245,743**	**119,522**	**210,929**	**243.912M**	**174.387M**	**937.256M**
# Entity-Entity Edges	91,186	40,748	78,467	0.114B	0.076B	0.414B
# Event-Entity Edges	143,254	63,680	123,527	0.265B	0.208B	1.063B
# Event-Event Edges	45,157	21,062	36,602	0.071B	0.044B	0.295B
# Conceptulization Edges	933,330	432,869	789,608	1.041B	0.821B	4.178B
# Edges	**1,212,927**	**558,359**	**1,028,204**	**1.492B**	**1.150B**	**5.958B**

Table 1: Statistics of knowledge graph construction across QA datasets (MuSiQue, 2WikiQA, HotpotQA) and LLM pre-training corpora (En-Wiki, Pes2o-Abstract, Common Crawl) for ATLAS knowledge graphs, showing counts of text chunks, nodes (entities/events), concepts, edges (Entity-Entity, Event-Entity, Event-Event), and conceptualizations. M = million, B = billion.

optimized precision settings and GPU acceleration. Extracted triples with their corresponding texts and metadata are serialized into JSON files.

3.2 Schema Induction

Following triple extraction, we perform schema induction to abstract specific entities, events, and relations into generalized types. This process uses LLMs to generate conceptual phrases representing types of each graph element, aligning with our formal definition $G = (V, E, C, \phi, \psi)$. For each category (events, entities, and relations), we process elements in batches. The LLM generates at least three phrases per element that encapsulate its type or related concepts at varying abstraction levels. For entities ($e \in V_N$), we enhance abstraction by incorporating contextual information from neighboring nodes. We sample up to N_{ctx} neighbors to construct a context string that provides additional semantic cues. The schema induction pipeline processes the graph serialized from the triple extraction phase. Elements are partitioned into batches, with options for slicing for distributed computation. The generated phrases are recorded in a CSV file, mapping each node $v \in V$ and relation $r \in R$ to a subset of concepts in C via ϕ and ψ. This automated schema enhances the knowledge graph's adaptability across varied domains without requiring manual curation.

4 Construction of ATLAS Families

Corpora As shown in Table 1, the ATLAS-Wiki, ATLAS-Pes2o, and ATLAS-CC are constructed from the subsets from Dolma's subset of Wikipedia & Wikibooks, Semantic Scholar, and Dolma's CC respectively.[2] We use the full Wikipedia & Wikibooks to construct the ATLAS-Wiki, and we use the abstract part of Semantic Scholar to construct ATLAS-Pes2o, and we use the each of 3% from cc-head, cc-middle, and cc-tail to construct ATLAS-CC. According to (Soldaini et al., 2024) the head, middle, tail of CC are used to measure the distribution similarity to Wikipedia text.

Computational Cost We constructed our knowledge graphs using 80GB GPUs with 1,513 TFLOPS of FP16 compute, running `Llama-3-8B-instruct` with Flash Attention. The computational demands were substantial: 14,300 GPU hours for En-Wiki (243.9M nodes, 1.49B edges), 11,800 GPU hours for Pes2o-Abstract (174.4M nodes, 1.15B edges), and 52,300 GPU hours for Common Crawl (937.3M nodes, 5.96B edges). Processing 1024-token chunks in batches, we invested approximately 78,400 GPU hours total to extract billions of semantic relationships.

5 Experiment

In the this section, we show that the AutoSchemaKG has accurate triplex extraction, can coherently induce schemas, and has very high information preservations in section 5.1.

5.1 Evaluating AutoSchemaKG

Evaluating Triple Extraction Accuracy We use a rigorous counting-based evaluation method. Rather than relying on subjective scoring, we employ DeepSeek-V3 (Liu et al., 2024) as a judge in a structured verification process. For each document: (1) We present DeepSeek-V3 with both the

[2]https://huggingface.co/datasets/allenai/dolma

Knowledge Graph	Triple Type	Precision	Recall	F1
ATLAS-Wiki	Entity-Entity	99.13	90.10	94.09
	Event-Entity	100.0	92.59	95.60
	Event-Event	99.60	93.59	96.01
ATLAS-Pes2o	Entity-Entity	97.66	89.89	93.03
	Event-Entity	100.0	94.29	96.83
	Event-Event	99.54	91.31	94.94
ATLAS-CC	Entity-Entity	95.65	84.64	88.82
	Event-Entity	99.93	87.92	92.72
	Event-Event	99.86	93.20	96.16

Table 2: Triple precision, recall and F1 score across datasets. Each row displays the performance of a type of extracted triples.

original text and the triples extracted by Llama-3-8B-Instruct; (2) DeepSeek-V3 identifies triples that are incorrectly extracted (false positives); (3) DeepSeek-V3 lists facts present in the original text but missing from the extracted triples (false negatives); This methodology allows us to calculate precise metrics: (1) Precision: proportion of correctly extracted triples out of all extracted triples; (2) Recall: proportion of correctly extracted triples among all ground-truth triples in the text; (3) F1 score: the harmonic mean of precision and recall. As shown in Table 2, our approach demonstrates exceptional extraction quality across all datasets, with particularly strong performance on Wikipedia content. The precision, recall, and F1 scores of the triples in our KG exceed 90% in most cases, demonstrating the high quality and reliability of our extracted triples.

Dataset	Context	Model	
		LLaMA-3-8B	LLaMA-3-70B
ATLAS-Wiki	[Lower-Upper]	46.29-99.29	65.69-99.70
	Entity	65.08	70.96
	Event	<u>92.69</u>	<u>94.82</u>
	Event + Entity	**93.30**	**95.13**
ATLAS-Pes2o	[Lower-Upper]	62.32-98.99	75.05-99.49
	Entity	80.00	83.33
	Event	<u>96.97</u>	<u>97.78</u>
	Event + Entity	**97.37**	**97.98**
ATLAS-CC	[Lower-Upper]	56.08-97.29	70.25-99.10
	Entity	76.78	81.01
	Event	<u>94.87</u>	<u>96.78</u>
	Event + Entity	**96.28**	**96.98**

Table 3: KG performance across datasets, showing bounds (no context to full passage) and results with different knowledge representations. Entity, Event, and combined representations preserve most information for MCQs, approaching full-passage performance across all datasets and models.

Measuring Information Preservation in Knowledge Graphs We evaluate the effectiveness of the entity-level triples and event-level triples of our constructed KG in preserving information from original passages. We test how well multiple-choice question (MCQ) performance is preserved when we convert the original passage into KG data. Following the evaluation protocol from the existing work (Schuhmann et al., 2025), we generate five MCQs with `LLaMA-3-70B-Instruct` for each original passage, and the prompts are shown in Figure 8. We sample 200 original passages and 1,000 MCQs are obtained for each dataset. We ask LLMs to answer them with no context (denoted as lower bound), then ask them again with the original passage (denoted as upper bound) for sanity check. Finally, we conduct tests using entity-level triples (denoted as Entity), event-level triples (denoted as Event), and a combination of both entity-level and event-level triples (denoted as Event + Entity). We evaluate on three pre-training datasets with our constructed KG in Table 1: En-Wiki, Pes2o-Abstract and Common Crawl. According to the results shown in Table 3, we have the following insights: (1) **Information is well preserved in our constructed KG.** MCQs performance with Entity, Event or Event + Entity remains far above the lower bound baseline and approaches the original-passage upper bound. It suggests that the information in the original passages is well preserved in our constructed KG; (2) **Events are more effective than entities.** The MCQs performance with Event or Event + Entity is much closer to the upper bound than that with Entity, which accuracy is more than 95% in most of the cases. It demonstrates that the event-level triples can preserve richer and more precious information than entity-level triples.

	Task	Dataset	BS-R	BS-C
LLaMA-3-8B	*Entity Typing*	FB15kET	88.57	86.54
		YAGO43kET	80.67	58.86
	Event Typing	wikiHow	99.18	99.26
	Relation Typing	FB15kET	88.75	88.41
LLaMA-3-13B	*Entity Typing*	FB15kET	89.25	88.59
		YAGO43kET	94.26	90.56
	Event Typing	wikiHow	98.97	99.33
	Relation Typing	FB15kET	88.58	88.66
LLaMA-3-70B	*Entity Typing*	FB15kET	89.49	87.30
		YAGO43kET	94.61	92.64
	Event Typing	wikiHow	99.41	99.15
	Relation Typing	FB15kET	88.70	90.33

Table 4: Results of schema induction with various LLaMA family LLMs across three kinds of typing tasks.

Measuring Schema Quality To demonstrate our schema induction method's capability, we apply it to entity, event, and relation typing tasks, measuring how many types our method recalls. Dataset details are in Appendix C.1. Since rule-based evaluation might overlook semantic similarities, we use two semantic-level metrics: **BS-R** and **BS-C**, explained in Appendix C.2. Table 4 shows results using three sizes of LLaMA-3. Our method achieves over 80% and often 90% recall for entity, event, and relation types in most cases, with performance improving as LLM parameter size increases.

5.2 Performance on Multi-hop QA Tasks

This subsection details the experimental setup for open-domain QA, focusing on multi-hop reasoning tasks where our knowledge graph's structure and schema induction are expected to excel.

Datasets We select three benchmark datasets known for their multi-hop reasoning demands, all derived from Wikipedia: MuSiQue (Trivedi et al., 2022), HotpotQA (Yang et al., 2018), and 2WikiMultihopQA (Ho et al., 2020), necessitating complex relational paths across articles. From each dataset, we randomly select one thousand questions following (Gutiérrez et al., 2024).

Baselines and Metrics We compare our knowledge graph-based RAG system against several state-of-the-art approaches. The graph-based baselines include HippoRAG (Gutiérrez et al., 2024), a framework that builds a memory graph from text using entity recognition and relation extraction; HippoRAG2 (Gutiérrez et al., 2025), an advanced iteration with enhanced graph construction; GraphRAG (Edge et al., 2024), Microsoft Research's technique combining text extraction, network analysis, and LLM prompting; LightRAG (Guo et al., 2024), a simpler alternative to GraphRAG focused on efficiency; and MiniRAG (Fan et al., 2025), an extremely simple framework tailored for Small Language Models. For MiniRAG and LightRAG we adjust its prompts so that it outputs brief answers for QA instead of generating long answers with explanations. For text-based RAG comparisons, we evaluate against BM25 + LLM using traditional retrieval with BM25 scoring; Contriever (Izacard et al., 2021), a dense retrieval-augmented system fine-tuned for QA; and RAPTOR (Sarthi et al., 2024), a hierarchical summarization system. These baselines allow us to benchmark our approach against diverse retrieval-augmented methods.

Model/Dataset	**MuSiQue**		**2Wiki**		**HotpotQA**	
Metric	**EM**	**F1**	**EM**	**F1**	**EM**	**F1**
Baseline Retrievers						
No Retriever	17.6	26.1	36.5	42.8	37.0	47.3
Contriever	24.0	31.3	38.1	41.9	51.3	62.3
BM25	20.3	28.8	47.9	51.2	52.0	63.4
LLM Embeddings						
GTE-Qwen2-7B	30.6	40.9	55.1	60.0	58.6	71.0
GritLM-7B	33.6	44.8	55.8	60.6	60.7	73.3
NV-Embed-v2 (7B)	34.7	45.7	57.5	61.5	**62.8**	75.3
Existing Graph-based RAG Methods						
RAPTOR	20.7	28.9	47.3	52.1	56.8	69.5
GraphRAG	27.3	38.5	51.4	58.6	55.2	68.6
LightRAG	20.0	29.3	38.6	44.6	33.3	44.9
MiniRAG	9.6	16.8	13.2	21.4	47.1	59.9
HippoRAG	26.2	35.1	65.0	71.8	52.6	63.5
HippoRAG2	**37.2**	**48.6**	65.0	71.0	62.7	75.5
AutoSchemaKG (`LLama-3.1-8B-Instruct`) + *Think-on-Graph*						
Entity-KG	14.8	26.0	36.9	44.0	41.9	55.2
Entity-Event-KG	19.4	32.8	39.0	47.1	47.7	61.2
Full-KG	20.1	31.2	40.0	47.7	48.2	60.5
AutoSchemaKG (`LLama-3.1-8B-Instruct`) + *HippoRAG*						
Entity-KG	22.5	36.4	57.7	65.8	50.3	65.8
Entity-Event-KG	22.9	36.1	56.4	64.4	48.6	64.6
Full-KG	23.6	36.5	54.8	63.2	50.0	65.3
AutoSchemaKG (`LLama-3.1-8B-Instruct`) + *HippoRAG2*						
Entity-KG	31.4	47.2	64.2	73.3	60.9	77.5
Entity-Event-KG	31.6	47.3	65.2	73.7	60.0	77.0
Full-KG	31.8	47.3	**65.3**	**73.9**	61.8	**78.3**

Table 5: Performance comparison of AutoSchemaKG integrated with ToG, HippoRAG and HippoRAG2 with bold indicating the highest performance per dataset.

We evaluate our system using standard metrics for open-domain QA. Exact Match (EM) measures binary correctness after normalization: $\text{EM}(a, g) = \mathbf{1}[\text{norm}(a) = \text{norm}(g)]$, where a is the predicted answer, g is the gold answer, and normalization includes lowercasing and removing articles, punctuation, and whitespace. F1 Score measures token overlap between normalized answers: $\text{F1} = \frac{2 \cdot P \cdot R}{P+R}$, where $P = |a \cap g|/|a|$ and $R = |a \cap g|/|g|$ are precision and recall.

Think on Graph Settings We implement Think on Graph (ToG) (Sun et al., 2024a) using a knowledge graph derived from our corpus. Nodes represent extracted entities and concepts, with edges showing semantic relations. We use `multi-qa-MiniLM-L6-dot-v1` to compute embeddings for all graph elements, indexed with FAISS. `LLama-3.3-70B-Instruct` performs entity recognition, path scoring, reasoning, and answer generation. The workflow extracts query entities, retrieves starting nodes, explores graph paths through depth-limited search, prunes irrelevant paths, and generates answers based on retrieved paths. Algorithm 1 in the Appendix provides details.

HippoRAG 1&2 Settings In our implementation of HippoRAG (Gutiérrez et al., 2024), we extend the original framework to operate on a customized graph. Initially presented in the foundational paper, we employ Named Entity Recognition (NER) to build a personalized dictionary for PageRank execution. Regarding HippoRAG2 (Gutiérrez et al., 2025), we select the top 30 edges (musique dataset 50 edges) for LLM filtering, incorporating a weight adjustment factor of 0.9. Considering the capability of our graph to effectively locating subgraphs, combined with various graph configurations (entity, event, concept) resulting in graphs of differing densities, we set the damping factor to 0.9 to concentrate on propagation within the local subgraph. For further implementation details, please refer to Algorithm 2.

Implementation Details The knowledge graph is constructed from the corresponding context corpora for each dataset folowing (Gutiérrez et al., 2024) using the framework of AutoSchemaKG with $L_{max} = 1024$ and $B = 16$, and the schema induction pipeline (Section B.2) with $B_s = 5$. We employ Meta's `LLaMA-3.1-8B-Instruct` to construct the graphs, optimized with bfloat16 precision and Flash Attention 2. The graph is stored in NetworkX for retrieval, with subgraphs fed into `LLaMA-3.3-70B-Instruct` for answer generation.

Evaluation Results The experimental results in Figure 5 and Figure 8 demonstrate AutoSchemaKG's effectiveness in multi-hop question answering across three benchmark datasets. With HippoRAG2 integration, the Full-KG configuration (entities, events, and concepts) outperforms traditional retrieval approaches like BM25 and Contriever by 12-18%, highlighting its strength in complex reasoning scenarios. Notably, AutoSchemaKG achieves comparable or better results using `LLaMA-3.1-8B-Instruct` as graph constructor compared to the original HippoRAG2 implementation that requires `LLaMA-3.3-70B-Instruct` for both construction and QA reading.

Advantages of Events and Concepts Our case studies revealed two key benefits of event and concept nodes: **1) Event nodes provide enriched context.** As shown in Figure 9, they serve as valuable retrieval targets when critical information in triples is ambiguous or missed, helping identify relevant subgraphs containing passage nodes; **2) Concept nodes create alternative pathways.** These nodes establish connections beyond direct entities and events, addressing complex multi-hop question answering limitations. Figure 10 shows how concept nodes link knowledge across disparate subgraphs, enabling systems like HippoRAG to bridge separate subgraph influences via PageRank algorithms.

Corpus	**Method**	**Acc**	**F1**
-	-	54.08	26.79
Text Corpora			
Wikipedia	Random	52.77	25.56
	BM25	<u>56.15</u>	<u>30.43</u>
	Dense Retrieval	56.04	30.33
Pes2o-Abstract	Random	53.34	26.00
	BM25	54.60	27.95
	Dense Retrieval	55.43	29.19
Common Crawl	Random	53.31	26.45
	BM25	54.56	28.32
	Dense Retrieval	54.42	28.49
Knowledge Graph			
Freebase	Think on Graph	53.75	24.81
ATLAS-Wiki	HippoRAG2	**56.43**	**30.48**
ATLAS-Pes2o		55.30	28.12
ATLAS-CC		55.56	29.57

Table 6: Balanced accuracy (%) and F1 score (%) on FELM benchmark of `Llama-3.1-8b-Instruct` with retrieval methods. The best results are in **bold**, and the second best results are <u>underlined</u>.

5.3 Enhancing LLM Factuality with KGs

We evaluated our KG's effectiveness in enhancing factuality using the FELM benchmark (Chen et al., 2023), which contains 847 samples across five domains with 4,425 fine-grained text segments. Following FELM's protocol, we applied RAG to three domains (world knowledge, science/technology, and writing/recommendation) while maintaining vanilla settings for math and reasoning domains. For a comprehensive comparison, we evaluated against multiple retrieval methods: HippoRAG v2, BM25, and dense retrieval using MiniLM (Wang et al., 2021). The retrieval process on text corpora is implemented in ElasticSearch database system (Elasticsearch, 2018). Our decision to use MiniLM rather than larger language model-based embedding approaches was driven by computational constraints. Implementing dense retrieval with higher-dimensional embeddings (e.g., 4096 dimensions) across our one billion nodes would require approximately 16 terabytes of storage using standard 32-bit floating-point representation. These

Knowledge Source	History	Law	Religion	Phil/Eth	Med/Hlth	GlbFct	SocSci	Logic	Average
None	76.59	66.86	83.04	63.55	70.38	66.72	79.74	64.35	72.10
Freebase-ToG	**78.42**	69.00	75.44	65.67	72.65	67.27	76.00	66.03	72.34
Random Baseline									
Wikipedia	76.64	66.82	79.53	59.26	70.34	66.46	77.78	59.21	70.62
Common Crawl	74.89	66.52	79.53	61.74	69.82	68.11	77.52	59.30	70.60
Pes2o-Abstract	76.24	64.16	80.70	62.01	70.62	66.59	77.16	62.39	70.82
Text Corpora + DBM25									
Wikipedia	76.67	67.35	78.36	63.34	69.35	61.98	76.99	62.30	70.61
Common Crawl	76.15	66.36	80.12	60.43	69.58	64.67	76.71	63.18	70.36
Pes2o-Abstract	78.01	65.89	78.95	63.83	71.01	65.78	77.07	59.34	71.22
Text Corpora + Dense Retrieval									
Wikipedia	73.59	69.60	79.53	63.58	70.82	62.41	76.83	62.21	70.86
Common Crawl	74.47	68.98	79.53	60.46	69.29	64.09	75.21	61.86	69.56
Pes2o-Abstract	75.79	61.82	78.36	65.15	69.72	66.77	76.47	63.05	70.52
ATLAS + HippoRAG2									
ATLAS-Wiki	76.73	67.38	84.21	**66.01**	70.82	68.36	79.16	63.65	72.53
ATLAS-CC	78.16	**70.85**	83.04	65.60	71.28	63.95	78.16	65.42	72.66
ATLAS-Pes2o	77.13	68.41	81.29	65.05	**72.75**	65.67	81.19	62.98	73.25
ATLAS + ToG									
ATLAS-Wiki	77.91	66.60	84.21	65.10	70.69	63.85	78.31	67.08	72.18
ATLAS-CC	77.07	68.18	83.63	65.24	72.03	66.87	79.72	66.59	73.07
ATLAS-Pes2o	77.52	66.95	**84.80**	63.44	71.15	**68.92**	**81.59**	**67.87**	**73.28**

Table 7: Performance comparison of `Llama-3.1-8b-Instruct` with our KG-integrated HippoRAG2 and ToG versus baseline methods across Wikipedia, Common Crawl, and Pes2o-Abstract corpora on MMLU benchmarks. Tasks are grouped by subject, with bold and underlined values indicating first and second-highest scores. Phil/Eth, Med/Hlth, GlbFct, and SocSci denote Philosophy/Ethics, Medicine/Health, Global Facts, and Social Sciences.

baselines represent state-of-the-art approaches in graph-based RAG and standalone retrieval systems. All experiments were implemented using the same `LLaMA-3.1-8B-Instruct` model with Neo4j integration and zero-shot CoT settings, ensuring fair comparison across methods. Performance was measured using balanced accuracy (giving equal weight to true and false segments) and F1 score for detecting factual errors (Table 6). Our results demonstrate that HippoRAG2 with our KG consistently outperforms baselines on Wikipedia (56.43% accuracy, 30.48% F1) and Common Crawl corpora, while achieving competitive results on Pes2o-Abstract. The superior performance on Wikipedia likely stems from FELM samples being partially sourced from Wikipedia content. Detailed implementation specifics and extended results are available in Appendix G.1.

5.4 General Domain Knowledge Capabilities

To assess AutoSchemaKG's ability to construct knowledge graphs across various domains, we evaluated it on MMLU (Hendrycks et al., 2021), a comprehensive benchmark for LLM reasoning. Previous research on KNN-LMs (Khandelwal et al.) suggests that retrieval-augmented generation can sometimes hinder LLMs' reasoning capabilities (Wang et al., 2023a; Geng et al., 2025). While we do not expect RAG to universally improve LLM performance, our findings demonstrate significant improvements in knowledge-intensive domains, even those covered in LLM training data. Using the same retrieval and generation settings as our FELM experiments, we classified MMLU tasks into subject categories (detailed mapping in Appendix G.2) and focused on knowledge-intensive domains including History, Law, Religion, Philosophy/Ethics, Medicine/Health, Global Facts, Social Sciences, and Logic.

As shown in Table 7, our ATLAS knowledge graphs enhanced performance across these domains on all tested corpora. Notably, each ATLAS variant demonstrated distinct strengths: ATLAS-Pes2o excelled in Religion, Medicine/Health, Global Facts, and Social Sciences, reflecting its academic paper-sourced knowledge; ATLAS-Wiki showed advantages in general knowledge areas like Religion, Philosophy/Ethics, and Global Facts; while ATLAS-

CC performed best in Law and History, leveraging its broader web-sourced content. All ATLAS variants consistently outperformed both the no-retrieval baseline and Freebase-ToG in these humanities and social science domains. For example, in Law, our approach achieved a 4-point improvement over the baseline, while some other retrieval methods actually decreased performance, as shown in Table 7. The retrieval method also matters for some specific tasks. For example, in Logic, ToG on our ATLAS knowledge graphs performs much better than all the other methods.

The domain-specific performance pattern aligns with intuitive expectations: knowledge graphs excel in retrieving factual relationships critical for humanities and social sciences, while showing limited benefits in mathematical and technical domains where node-relation structures are less effective for capturing procedural knowledge. The complete subject analysis, including technical domains, is available in Appendix G.2.

6 Further Related Work

Knowledge graph (KG) construction transforms unstructured data into machine-readable formats. Traditional approaches using predefined schemas limit cross-domain scalability, while LLMs now enable autonomous construction through improved extraction and schema induction. Recent advances include SAC-KG (Chen et al., 2024a), which uses LLMs as domain experts to generate specialized KGs with high precision on million-node graphs; Docs2KG (Sun et al., 2024c) for heterogeneous document processing; and KAG (Liang et al., 2024), which enhances multi-hop reasoning through KG-text mutual indexing. One of the earliest approaches to extracting schemas or concepts from entities, events, or textual descriptions leveraged lexico-syntactic patterns to automatically harvest novel lexico-semantic concepts from documents (Hearst, 1998), which enhanced WordNet (Miller, 1995). Agirre et al. (2000) further enhanced WordNet by extracting topically related concepts from web documents. Txt2onto (Hawkins et al., 2022) proposed a NLP-ML approach to create numerical representations of texts and use these features in a supervised learning classifier that predicts terms and concepts. Recently, due to the strong ability of LLMs to capture complex language patterns in different knowledge domains, LLMs4OL (Babaei Giglou et al., 2023) uses zero-shot prompting method with LLMs for ontology learning tasks. Schema induction automatically derives KG structure without predefined ontologies. Hofer et al. (2024) survey construction pipelines emphasizing ontology learning, while Dash et al. (2021) address canonicalization using variational autoencoders and Dognin et al. (2021) employ reinforcement learning for bidirectional text-to-graph conversion.

7 Conclusion

AutoSchemaKG transforms knowledge graph construction by eliminating predefined schemas through LLM-based triple extraction and schema induction. Our methodology constructs the ATLAS family of knowledge graphs (900+ million nodes, 5.9 billion edges) with high-quality triple extraction (>95% precision) and schema induction. This approach outperforms baselines on multi-hop QA tasks and enhances LLM factuality across diverse domains. Our work demonstrates that billion-scale knowledge graphs with dynamically induced schemas can be effectively constructed without expert intervention, providing valuable complements to parametric knowledge in large language models.

8 Limitations

Despite promising results, our work has several important limitations. The construction of billion-scale knowledge graphs requires substantial computational resources (78,400+ GPU hours), limiting accessibility for researchers with resource constraints. Our approach inherits biases and limitations from the underlying LLMs used for triple extraction and schema induction, potentially affecting performance in specialized domains where these models lack expertise. While achieving high semantic alignment with human-crafted schemas, our induction method still struggles with extremely technical domains requiring expert-level conceptual organization. Despite extracting billions of facts, our knowledge graphs may contain inconsistencies, contradictions, or information gaps in sparse knowledge regions.

9 Ethical Statement

Our research on AutoSchemaKG and ATLAS knowledge graphs adheres to rigorous ethical standards. We exclusively utilized publicly available corpora (Dolma 1.7) with proper attribution and transparently disclosed our computational requirements (approximately 78,400 GPU hours). We acknowledge potential inherited biases from source texts and the large language models used in our pipeline, while emphasizing that our approach minimizes manual intervention that might introduce additional biases. Privacy considerations were addressed by processing only public data without extracting personally identifiable information. We candidly discuss limitations including resource constraints, LLM biases, and challenges with specialized domains. Through detailed methodological descriptions, we promote reproducibility and scientific validation. Our work aims to democratize knowledge graph construction without requiring specialized expertise or predefined schemas, potentially enabling more accurate and explainable AI systems across diverse applications and domains.

References

Eneko Agirre, Olatz Ansa, Eduard Hovy, and David Martínez. 2000. Enriching very large ontologies using the www. *arXiv preprint cs/0010026*.

Hamed Babaei Giglou, Jennifer D'Souza, and Sören Auer. 2023. Llms4ol: Large language models for ontology learning. In *International Semantic Web Conference*, pages 408–427. Springer.

Jiaxin Bai, Zhaobo Wang, Junfei Cheng, Dan Yu, Zerui Huang, Weiqi Wang, Xin Liu, Chen Luo, Qi He, Yanming Zhu, Bo Li, and Yangqiu Song. 2024. Intention knowledge graph construction for user intention relation modeling. *Preprint*, arXiv:2412.11500.

Kurt D. Bollacker, Colin Evans, Praveen K. Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, Vancouver, BC, Canada, June 10-12, 2008*, pages 1247–1250. ACM.

Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 2787–2795.

Chunkit Chan, Cheng Jiayang, Weiqi Wang, Yuxin Jiang, Tianqing Fang, Xin Liu, and Yangqiu Song. 2024. Exploring the potential of ChatGPT on sentence level relations: A focus on temporal, causal, and discourse relations. In *Findings of the Association for Computational Linguistics: EACL 2024*, pages 684–721, St. Julian's, Malta. Association for Computational Linguistics.

Chunkit Chan, Xin Liu, Jiayang Cheng, Zihan Li, Yangqiu Song, Ginny Wong, and Simon See. 2023. DiscoPrompt: Path prediction prompt tuning for implicit discourse relation recognition. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 35–57, Toronto, Canada. Association for Computational Linguistics.

Hanzhu Chen, Xu Shen, Qitan Lv, Jie Wang, Xiaoqi Ni, and Jieping Ye. 2024a. 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, Bangkok, Thailand. Association for Computational Linguistics.

Muhao Chen, Hongming Zhang, Haoyu Wang, and Dan Roth. 2020. " what are you trying to do?" semantic typing of event processes. *arXiv preprint arXiv:2010.06724*.

Shiqi Chen, Yiran Zhao, Jinghan Zhang, I-Chun Chern, Siyang Gao, Pengfei Liu, and Junxian He. 2023. Felm: benchmarking factuality evaluation of large language models. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA. Curran Associates Inc.

Tong Chen, Hongwei Wang, Sihao Chen, Wenhao Yu, Kaixin Ma, Xinran Zhao, Hongming Zhang, and Dong Yu. 2024b. Dense X retrieval: What retrieval granularity should we use? In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 15159–15177, Miami, Florida, USA. Association for Computational Linguistics.

Zhongwu Chen, Long Bai, Zixuan Li, Zhen Huang, Xiaolong Jin, and Yong Dou. 2024c. A new pipeline for knowledge graph reasoning enhanced by large language models without fine-tuning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 1366–1381, Miami, Florida, USA. Association for Computational Linguistics.

Sarthak Dash, Gaetano Rossiello, Nandana Mihindukulasooriya, Sugato Bagchi, and Alfio Gliozzo. 2021. Open knowledge graphs canonicalization using variational autoencoders. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10379–10394, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Pierre Dognin, Inkit Padhi, Igor Melnyk, and Payel Das. 2021. ReGen: Reinforcement learning for text and knowledge base generation using pretrained language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1084–1099, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. 2024. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*.

BV Elasticsearch. 2018. Elasticsearch. *software], version*, 6(1).

Tianyu Fan, Jingyuan Wang, Xubin Ren, and Chao Huang. 2025. Minirag: Towards extremely simple retrieval-augmented generation. *arXiv preprint arXiv:2501.06713*.

Shangyi Geng, Wenting Zhao, and Alexander M Rush. 2025. Great memory, shallow reasoning: Limits of kNN-LMs. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pages 471–482, Albuquerque, New Mexico. Association for Computational Linguistics.

Zirui Guo, Lianghao Xia, Yanhua Yu, Tu Ao, and Chao Huang. 2024. Lightrag: Simple and fast retrieval-augmented generation.

Bernal Jiménez Gutiérrez, Yiheng Shu, Weijian Qi, Sizhe Zhou, and Yu Su. 2025. From rag to memory: Non-parametric continual learning for large language models. *arXiv preprint arXiv:2502.14802*.

Bernal Jiménez Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. 2024. Hipporag: Neurobiologically inspired long-term memory for large language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Nathaniel T Hawkins, Marc Maldaver, Anna Yannakopoulos, Lindsay A Guare, and Arjun Krishnan. 2022. Systematic tissue annotations of genomics samples by modeling unstructured metadata. *Nature Communications*, 13(1):6736.

Mutian He, Tianqing Fang, Weiqi Wang, and Yangqiu Song. 2024. Acquiring and modeling abstract commonsense knowledge via conceptualization. *Artificial Intelligence*, 333:104149.

Marti A Hearst. 1998. Automated discovery of wordnet relations. *WordNet: an electronic lexical database*, 2.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. *Preprint*, arXiv:2009.03300.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Marvin Hofer, Daniel Obraczka, Alieh Saeedi, Hanna Köpcke, and Erhard Rahm. 2024. Construction of knowledge graphs: Current state and challenges. *Information*, 15(8):509.

Alexander Hoyle, Rupak Sarkar, Pranav Goel, and Philip Resnik. 2023. Natural language decompositions of implicit content enable better text representations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13188–13214, Singapore. Association for Computational Linguistics.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2021. Unsupervised dense information retrieval with contrastive learning.

Harsh Jhamtani, Hao Fang, Patrick Xia, Eran Levy, Jacob Andreas, and Benjamin Van Durme. 2024. Natural language decomposition and interpretation of complex utterances. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, pages 6306–6314.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. Generalization through memorization: Nearest neighbor language models. In *International Conference on Learning Representations*.

Mahnaz Koupaee and William Yang Wang. 2018. Wikihow: A large scale text summarization dataset. *arXiv preprint arXiv:1810.09305*.

Muzhi Li, Minda Hu, Irwin King, and Ho-fung Leung. 2024a. The integration of semantic and structural knowledge in knowledge graph entity typing. 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 6625–6638, Mexico City, Mexico. Association for Computational Linguistics.

Yinghao Li, Haorui Wang, and Chao Zhang. 2024b. Assessing logical puzzle solving in large language models: Insights from a minesweeper case study. 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 59–81, Mexico City, Mexico. Association for Computational Linguistics.

Lei Liang, Mengshu Sun, Zhengke Gui, Zhongshu Zhu, Zhouyu Jiang, Ling Zhong, Yuan Qu, Peilong Zhao, Zhongpu Bo, Jin Yang, Huaidong Xiong, Lin Yuan, Jun Xu, Zaoyang Wang, Zhiqiang Zhang, Wen Zhang, Huajun Chen, Wenguang Chen, and Jun Zhou. 2024. Kag: Boosting llms in professional domains via knowledge augmented generation. *Preprint*, arXiv:2409.13731.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

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

Hanjia Lyu, Song Jiang, Hanqing Zeng, Yinglong Xia, Qifan Wang, Si Zhang, Ren Chen, Chris Leung, Jiajie Tang, and Jiebo Luo. 2024. LLM-rec: Personalized recommendation via prompting large language models. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 583–612, Mexico City, Mexico. Association for Computational Linguistics.

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

Changsung Moon, Paul Jones, and Nagiza F Samatova. 2017a. Learning entity type embeddings for knowledge graph completion. In *Proceedings of the 2017 ACM on conference on information and knowledge management*, pages 2215–2218.

Changsung Moon, Paul Jones, and Nagiza F. Samatova. 2017b. Learning entity type embeddings for knowledge graph completion. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, CIKM '17, page 2215–2218, New York, NY, USA. Association for Computing Machinery.

Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher D. Manning. 2024. Raptor: Recursive abstractive processing for tree-organized retrieval. In *International Conference on Learning Representations (ICLR)*.

Christoph Schuhmann, Gollam Rabby, Ameya Prabhu, Tawsif Ahmed, Andreas Hochlehnert, Huu Nguyen, Nick Akinci Heidrich, Ludwig Schmidt, Robert Kaczmarczyk, Sören Auer, et al. 2025. Project alexandria: Towards freeing scientific knowledge from copyright burdens via llms. *arXiv preprint arXiv:2502.19413*.

Luca Soldaini, Rodney Kinney, Akshita Bhagia, Dustin Schwenk, David Atkinson, Russell Authur, Ben Bogin, Khyathi Chandu, Jennifer Dumas, Yanai Elazar, Valentin Hofmann, Ananya Harsh Jha, Sachin Kumar, Li Lucy, Xinxi Lyu, Nathan Lambert, Ian Magnusson, Jacob Morrison, Niklas Muennighoff, Aakanksha Naik, Crystal Nam, Matthew E. Peters, Abhilasha Ravichander, Kyle Richardson, Zejiang Shen, Emma Strubell, Nishant Subramani, Oyvind Tafjord, Pete Walsh, Luke Zettlemoyer, Noah A. Smith, Hannaneh Hajishirzi, Iz Beltagy, Dirk Groeneveld, Jesse Dodge, and Kyle Lo. 2024. Dolma: an open corpus of three trillion tokens for language model pretraining research. *CoRR*, abs/2402.00159.

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

Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, Saizhuo Wang, Chen Lin, Yeyun Gong, Lionel Ni, Heung-Yeung Shum, and Jian Guo. 2024a. Think-on-graph: Deep and responsible reasoning of large language model on knowledge graph. In *The Twelfth International Conference on Learning Representations*.

Kai Sun, Yifan Xu, Hanwen Zha, Yue Liu, and Xin Luna Dong. 2024b. Head-to-tail: How knowledgeable are large language models (LLMs)? A.K.A. will LLMs replace knowledge graphs? 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 311–325, Mexico City, Mexico. Association for Computational Linguistics.

Qiang Sun, Yuanyi Luo, Wenxiao Zhang, Sirui Li, Jichunyang Li, Kai Niu, Xiangrui Kong, and Wei Liu. 2024c. Docs2kg: Unified knowledge graph construction from heterogeneous documents assisted by large language models. *Preprint*, arXiv:2406.02962.

Xingwei Tan, Yuxiang Zhou, Gabriele Pergola, and Yulan He. 2024. Set-aligning framework for auto-regressive event temporal graph generation. 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 3872–3892, Mexico City, Mexico. Association for Computational Linguistics.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. MuSiQue: Multi-hop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*.

Shufan Wang, Yixiao Song, Andrew Drozdov, Aparna Garimella, Varun Manjunatha, and Mohit Iyyer. 2023a. kNN-LM does not improve open-ended text generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 15023–15037, Singapore. Association for Computational Linguistics.

Weiqi Wang, Tianqing Fang, Wenxuan Ding, Baixuan Xu, Xin Liu, Yangqiu Song, and Antoine Bosselut. 2023b. CAR: Conceptualization-augmented reasoner for zero-shot commonsense question answering. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 13520–13545, Singapore. Association for Computational Linguistics.

Weiqi Wang, Tianqing Fang, Chunyang Li, Haochen Shi, Wenxuan Ding, Baixuan Xu, Zhaowei Wang, Jiaxin Bai, Xin Liu, Cheng Jiayang, Chunkit Chan, and Yangqiu Song. 2024a. CANDLE: Iterative conceptualization and instantiation distillation from large language models for commonsense reasoning. In

Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 2351–2374, Bangkok, Thailand. Association for Computational Linguistics.

Weiqi Wang, Tianqing Fang, Haochen Shi, Baixuan Xu, Wenxuan Ding, Liyu Zhang, Wei Fan, Jiaxin Bai, Haoran Li, Xin Liu, and Yangqiu Song. 2024b. On the role of entity and event level conceptualization in generalizable reasoning: A survey of tasks, methods, applications, and future directions. *Preprint*, arXiv:2406.10885.

Weiqi Wang, Tianqing Fang, Baixuan Xu, Chun Yi Louis Bo, Yangqiu Song, and Lei Chen. 2023c. CAT: A contextualized conceptualization and instantiation framework for commonsense reasoning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13111–13140, Toronto, Canada. Association for Computational Linguistics.

Wenhui Wang, Hangbo Bao, Shaohan Huang, Li Dong, and Furu Wei. 2021. MiniLMv2: Multi-head self-attention relation distillation for compressing pretrained transformers. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2140–2151, Online. Association for Computational Linguistics.

Zehong Wang, Sidney Liu, Zheyuan Zhang, Tianyi Ma, Chuxu Zhang, and Yanfang Ye. 2025. Can LLMs convert graphs to text-attributed graphs? In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1412–1432, Albuquerque, New Mexico. Association for Computational Linguistics.

Zhaowei Wang, Wei Fan, Qing Zong, Hongming Zhang, Sehyun Choi, Tianqing Fang, Xin Liu, Yangqiu Song, Ginny Wong, and Simon See. 2024c. AbsInstruct: Eliciting abstraction ability from LLMs through explanation tuning with plausibility estimation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 973–994, Bangkok, Thailand. Association for Computational Linguistics.

Zhaowei Wang, Haochen Shi, Weiqi Wang, Tianqing Fang, Hongming Zhang, Sehyun Choi, Xin Liu, and Yangqiu Song. 2024d. AbsPyramid: Benchmarking the abstraction ability of language models with a unified entailment graph. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 3991–4010, Mexico City, Mexico. Association for Computational Linguistics.

Yike Wu, Yi Huang, Nan Hu, Yuncheng Hua, Guilin Qi, Jiaoyan Chen, and Jeff Z. Pan. 2024. CoTKR: Chain-of-thought enhanced knowledge rewriting for complex knowledge graph question answering. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 3501–3520, Miami, Florida, USA. Association for Computational Linguistics.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Hongbin Ye, Honghao Gui, Xin Xu, Xi Chen, Huajun Chen, and Ningyu Zhang. 2023. Schema-adaptable knowledge graph construction. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 6408–6431, Singapore. Association for Computational Linguistics.

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, Miami, Florida, USA. Association for Computational Linguistics.

Hongming Zhang, Xin Liu, Haojie Pan, Haowen Ke, Jiefu Ou, Tianqing Fang, and Yangqiu Song. 2022. ASER: towards large-scale commonsense knowledge acquisition via higher-order selectional preference over eventualities. *Artificial Intelligence*, 309:103740.

Hongming Zhang, Xin Liu, Haojie Pan, Yangqiu Song, and Cane Wing-Ki Leung. 2020. Aser: A large-scale eventuality knowledge graph. In *Proceedings of the web conference 2020*, pages 201–211.

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

Wenting Zhao, Ye Liu, Tong Niu, Yao Wan, Philip Yu, Shafiq Joty, Yingbo Zhou, and Semih Yavuz. 2024. DIVKNOWQA: Assessing the reasoning ability of LLMs via open-domain question answering over knowledge base and text. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 51–68, Mexico City, Mexico. Association for Computational Linguistics.

Chang Zong, Yuchen Yan, Weiming Lu, Jian Shao, Yongfeng Huang, Heng Chang, and Yueting Zhuang. 2024. Triad: A framework leveraging a multi-role LLM-based agent to solve knowledge base question answering. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 1698–1710, Miami, Florida, USA. Association for Computational Linguistics.

A Prompts for Triple Extractions

Entity Relationship Extraction

Given a passage, summarize all the important entities and the relations between them in a concise manner. Relations should briefly capture the connections between entities, without repeating information from the head and tail entities. The entities should be as specific as possible. Exclude pronouns from being considered as entities. The output should strictly adhere to the following JSON format:

```
[
  {
    "Head": "{a noun}",
    "Relation": "{a verb}",
    "Tail": "{a noun}",
  }
  ...
]
```

Here is the passage:

Figure 2: The figure demonstrates the prompts we use to generate the text triples describing relations between entities.

Event and Entity Triple Extraction

Please analyze and summarize the participation relations between the events and entities in the given paragraph. Each event is a single independent sentence. Additionally, identify all the entities that participated in the events. Do not use ellipses. Please strictly output in the following JSON format:

```
[
  {
    "Event": "{a simple sentence describing an event}",
    "Entity": ["{entity 1}", "{entity 2}", "..."]
  }
  ...
]
```

Figure 3: The figure demonstrates the prompts we use to generate the text triples describing relations between entities and events.

The prompts used for extracting entity-entity, entity-event, and event-event triples are given in Figure 2, Figure 3, and Figure 4 respectively.

B Implementation Details of Knowledge Graph Construction Framework

In this section, we elaborate on the process of fully automating knowledge graph construction. Given a collection of n documents $D = \{D_1, D_2, D_3, \dots, D_n\}$, our method systematically builds the graph.

B.1 Triple Extraction

Our approach to triple extraction employs a multi-phase pipeline that utilizes the generative power of Large Language Models (LLMs) to convert unstructured text into structured knowledge triples, drawing from the Dolma corpus (Soldaini et al., 2024). This pipeline systematically extracts three categories of relationships—Entity-Entity, Entity-Event, and Event-Event—forming the foundation of a comprehensive knowledge graph. Designed for scalability and resilience, the method incorporates batch processing, text segmentation, and robust output parsing to efficiently process large-scale datasets.

The extraction unfolds across three sequential stages, each tailored to a specific relationship type, leveraging a single LLM guided by distinct prompts to produce structured outputs in JSON format. To manage the constraints of LLM input capacity, denoted as L_{max} tokens, we preprocess the text corpus to ensure compatibility, segmenting documents as needed and organizing them into batches for efficient processing. This section outlines the preprocessing strategy, the staged extraction process, and key implementation details.

B.1.1 Text Preprocessing and Data Organization

Given a corpus $D = \{D_1, D_2, \dots, D_n\}$ of n documents, we first filter the dataset to include only English-language texts, identified through metadata or assumed if unspecified, to match the linguistic capabilities of our LLMs. To adhere to the token limit L_{max}, we account for an instructional prompt length, L_{inst} derived from empirical observations. The maximum token length per text segment, C_{max}, is calculated as: $C_{max} = (L_{max} - L_{inst})$

Documents exceeding C_{max} are divided into smaller chunks, each tagged with a unique identifier and metadata to maintain traceability. This segmentation ensures that inputs remain within the LLM's token capacity, preserving contextual integrity without truncation.

The preprocessed text is then grouped into batches of size B, utilizing a custom data management framework that integrates with standard dataset loading tools. Tokenization is applied to each batch, adjusting for padding and truncation to produce consistent input representations suitable for LLM processing.

Event Relationship Extraction

Please analyze and summarize the relationships between the events in the paragraph. Each event is a single independent sentence. Identify temporal and causal relationships between the events using the following types: before, after, at the same time, because, and as a result. Each extracted triple should be specific, meaningful, and able to stand alone. Do not use ellipses. The output should strictly adhere to the following JSON format:

```
[
  {
    "Head": "{a simple sentence describing the event 1}",
    "Relation": "{temporal or causality relation between the events}",
    "Tail": "{a simple sentence describing the event 2}"
  }
  ...
]
```

Figure 4: The figure demonstrates the prompts we use to generate the text triples describing relations between events.

B.1.2 Stage 1: Extraction of Entity-Entity Relationships

In the initial stage, as shown in Figure 2, we extract Entity-Entity relationships, identifying connections between named entities such as individuals, organizations, or locations. For each batch, we prepend a system prompt, P_{EE}, which instructs the LLM to detect entities and their interrelations, followed by the segmented text. The LLM generates a response in JSON format, which is subsequently decoded and parsed. The parsing process isolates the structured content by locating the model's answer start token, T_{start}, repairs any malformed JSON, and extracts a list of dictionaries. Each dictionary represents a triple (e_1, r, e_2), where $e_1, e_2 \in V_N$ are entity nodes and $r \in R$ is a relation type. If parsing encounters errors, an empty list is returned to ensure pipeline continuity .

B.1.3 Stage 2: Extraction of Entity-Event Relationships

The second stage focuses on Entity-Event relationships, as shown in Figure 3, linking entities to specific occurrences or events. Using the original text segments from Stage 1, we apply a new prompt, P_{EV}, directing the LLM to identify events and their associated entities. The generation and parsing steps mirror those of Stage 1, producing triples of the form (e, r, v) or (v, r, e), where $e \in V_N$, $v \in V_E$ (event nodes), and $r \in R$. This bidirectional extraction captures both entities involved in events and events tied to entities, enhancing the graph's relational depth.

B.1.4 Stage 3: Extraction of Event-Event Relationships

The third stage targets Event-Event relationships, as shown in Figure 4,, detecting causal, temporal, or logical connections between events. A specialized prompt, P_{VV}, is applied to the text segments, prompting the LLM to generate triples of the form (v_1, r, v_2), where $v_1, v_2 \in V_E$ and $r \in R$. The parsing process follows the same methodology, repairing JSON outputs as needed. To accommodate potentially intricate event descriptions, we extend the generation limit to $L_{ext} = \alpha \cdot L_{max}$, where $\alpha > 1$ is a scaling factor, ensuring comprehensive capture of event interactions.

B.1.5 Implementation Considerations

The pipeline supports a variety of LLMs, including models from Google, Meta, Mistral, Microsoft, and others, configured with optimized precision settings (e.g., bfloat16 or float16) and enhanced with acceleration techniques where applicable. Deployment occurs on GPUs, with input-output formatting governed by model-specific chat templates, T_{chat}, to ensure compatibility. The extracted triples, along with their corresponding texts and metadata, are serialized into JSON files per batch, enabling subsequent schema induction and evaluation.

This multi-stage pipeline achieves thorough triple extraction by addressing each relationship type systematically, harnessing the LLM's generative capabilities within a scalable and fault-tolerant framework. The use of variables such as L_{max}, C_{max}, and B ensures flexibility across different models and datasets, reinforcing the methodology's adaptability and generalizability.

B.2 Schema Induction

Following the extraction of knowledge triples, our methodology advances to schema induction, a critical step that abstracts specific entities, events, and relations into generalized types to form a coherent and adaptable schema for the knowledge graph. This process leverages the contextual understanding of Large Language Models (LLMs) to generate conceptual phrases that represent the types or related concepts of each graph element, enabling the graph to scale across diverse domains without manual schema design. The induced schema aligns with the formal definition of a knowledge graph $G = (V, E, C, \phi, \psi)$, where C denotes the set of concepts, and ϕ and ψ map nodes and relations to subsets of C, respectively.

Our schema induction pipeline processes the triples extracted from the Dolma corpus (Soldaini et al., 2024), organizing them into batches and employing a generative approach to derive abstract representations. The process targets three components—events (V_E), entities (V_N), and relations (R)—producing a set of conceptual phrases for each, which collectively form the concept set C. This section outlines the abstraction methodology, the role of context in entity conceptualization, and key implementation details.

Abstract Event Phrase Generation

I will give you an EVENT. You need to give several phrases containing 1-2 words for the ABSTRACT EVENT of this EVENT. You must return your answer in the following format: phrases1, phrases2, phrases3,... You can't return anything other than answers. These abstract event words should fulfill the following requirements:

1. The ABSTRACT EVENT phrases can well represent the EVENT, and it could be the type of the EVENT or the related concepts of the EVENT.
2. Strictly follow the provided format, do not add extra characters or words.
3. Write at least 3 or more phrases at different abstract level if possible.
4. Do not repeat the same word and the input in the answer.
5. Stop immediately if you can't think of any more phrases, and no explanation is needed.

Examples:
EVENT: A man retreats to mountains and forests Your answer: retreat, relaxation, escape, nature, solitude
EVENT: A cat chased a prey into its shelter Your answer: hunting, escape, predation, hidding, stalking
EVENT: Sam playing with his dog Your answer: relaxing event, petting, playing, bonding, friendship
EVENT: [EVENT] Your answer:

Figure 5: This figure shows the prompt used for generating the concepts for an event.

B.2.1 Abstraction Methodology

The schema induction begins by categorizing the nodes and edges of the knowledge graph G into events, entities, and relations. For each category, we process the elements in batches of size B_s to optimize computational efficiency and scalability. The LLM is prompted with tailored instructions to generate a list of phrases, each containing one to two words, that abstractly represent the input element. These phrases must satisfy several criteria: they should encapsulate the element's type or related concepts, vary in abstraction level, and avoid repetition or inclusion of the original input term. For each element, a minimum of three phrases is targeted, though more may be generated depending on the LLM's output.

For events ($v \in V_E$), the prompt directs the LLM to identify abstract event types or related notions. For example, an event such as "Sam playing with his dog" might yield phrases like "playing," "bonding," and "relaxing event," reflecting different levels of generality. For entities ($e \in V_N$), the prompt similarly elicits abstract entity types, augmented by contextual information derived from the graph structure, as detailed below. Relations ($r \in R$) are abstracted into phrases that capture their semantic essence, such as transforming "participated in" into "engage in," "attend," and "involve in." The LLM generates these outputs in a structured format, which we parse into lists of phrases, forming the mappings $\phi(v)$, $\phi(e)$, and $\psi(r)$ to the concept set C.

The abstraction process operates in batches, processing B_s elements simultaneously. The input prompts are tokenized to a maximum length L_{tok}, and the LLM generates responses under controlled parameters (e.g., temperature τ and top-p sampling with probability p) to balance creativity and coherence. The resulting phrases are stored alongside their corresponding elements, ensuring traceability and enabling subsequent analysis.

B.2.2 Contextual Enhancement for Entities

As shown in Figure 6, to enhance the accuracy of entity abstraction, we incorporate contextual information extracted from the knowledge graph. For each entity $e \in V_N$, we examine its neighboring nodes—predecessors and successors—along with their associated relations. A subset of these neighbors, limited to N_{ctx} (e.g., one predecessor and one successor), is randomly sampled to construct a context string. This string concatenates the neighbor's identity and relation (e.g., "neighbor1 relation1, relation2 neighbor2"), providing the LLM with additional semantic cues. For instance, an entity "Black Mountain College" with context "started by John Andrew Rice" might yield phrases like "college," "school," and "liberal arts college." This contextual enrichment ensures that the abstracted types are grounded in the entity's role within the graph, improving the schema's relevance and specificity.

Abstract Entity Phrase Generation

I will give you an ENTITY. You need to give several phrases containing 1-2 words for the ABSTRACT ENTITY of this ENTITY. You must return your answer in the following format: phrases1, phrases2, phrases3,... You can't return anything other than answers. These abstract intention words should fulfill the following requirements:
1. The ABSTRACT ENTITY phrases can well represent the ENTITY, and it could be the type of the ENTITY or the related concepts of the ENTITY.
2. Strictly follow the provided format, do not add extra characters or words.
3. Write at least 3 or more phrases at different abstract level if possible.
4. Do not repeat the same word and the input in the answer.
5. Stop immediately if you can't think of any more phrases, and no explanation is needed.

Examples:
ENTITY: Soul CONTEXT: premiered BFI London Film Festival, became highest-grossing Pixar release Your answer: movie, film
ENTITY: Thinkpad X60 CONTEXT: Richard Stallman announced he is using Trisquel on a Thinkpad X60 Your answer: Thinkpad, laptop, machine, device, hardware, computer, brand
ENTITY: Harry Callahan CONTEXT: bluffs another robber, tortures Scorpio Your answer: person, Amarican, character, police officer, detective
ENTITY: Black Mountain College CONTEXT: was started by John Andrew Rice, attracted faculty Your answer: college, university, school, liberal arts college
ENTITY: 1st April CONTEXT: Utkal Dibas celebrates Your answer: date, day, time, festival
ENTITY: [ENTITY] CONTEXT: [CONTEXT] Your answer:

Figure 6: This figure shows the conceptualization prompts for entities enhanced with context.

Events and relations, as shown in Figure 5 and Figure 7, by contrast, rely solely on their textual descriptions without additional context, as their abstraction focuses on inherent semantics rather than graph connectivity. This distinction reflects the differing roles of nodes and edges in the knowledge graph structure.

B.2.3 Implementation Details

The schema induction pipeline processes a graph G serialized from the triple extraction phase, typically stored in a binary format and loaded into memory. The elements are partitioned into batches, with the option to apply slicing (dividing the workload into S_{total} slices and processing the S_{slice}-th portion) for distributed computation. If a sample size N_{sample} is specified, a random subset of batches is

Abstract Relation Phrase Generation

I will give you a RELATION. You need to give several phrases containing 1-2 words for the ABSTRACT RELATION of this RELATION. You must return your answer in the following format: phrases1, phrases2, phrases3,... You can't return anything other than answers. These abstract intention words should fulfill the following requirements:
1. The ABSTRACT RELATION phrases can well represent the RELATION, and it could be the type of the RELATION or the simplest concepts of the RELATION.
2. Strictly follow the provided format, do not add extra characters or words.
3. Write at least 3 or more phrases at different abstract level if possible.
4. Do not repeat the same word and the input in the answer.
5. Stop immediately if you can't think of any more phrases, and no explanation is needed.

Examples:
RELATION: participated in Your answer: become part of, attend, take part in, engage in, involve in
RELATION: be included in Your answer: join, be a part of, be a member of, be a component of
RELATION: [RELATION] Your answer:

Figure 7: This figure shows the conceptualization prompts for relations enhanced with context.

selected to reduce processing time during experimentation.

The LLM, configured with a precision setting (e.g., float16) and optimized with acceleration techniques, operates on a GPU to handle the batched inference efficiently. Prompts are formatted using a model-specific chat template, T_{chat}, ensuring compatibility with the LLM's input-output conventions. The generated phrases are written to a CSV file, with each row recording the original element, its abstracted phrases, and its type (event, entity, or relation). Post-processing aggregates these phrases to compute the unique concepts in C, providing statistics on the schema's coverage, such as the number of distinct event types, entity types, and relation types.

This approach yields a flexible and automated schema, mapping each node $v \in V$ and relation $r \in R$ to a subset of concepts in C via ϕ and ψ. By abstracting specific instances into general types, the induced schema enhances the knowledge graph's adaptability, supporting downstream applications across varied domains without requiring manual curation.

C Experiment Settings of Schema Accuracy

C.1 Datasets

Entity Typing. We conduct experiments on the typed entities of two real-world knowledge graphs, FB15kET (Bordes et al., 2013) and YAGO43kET (Moon et al., 2017a), which are the subsets of Freebase (Bollacker et al., 2008) and YAGO (Suchanek et al., 2007), respectively. The types of entities are collected from (Moon et al., 2017b). There are 3,584 and 45,182 entity types in FB15kET and YAGO43kET, respectively. We utilize the entities in the testing sets of these two datasets with their types as ground truths to validate the entity induction performance of our schema induction method.

Event Typing. We conduct experiments on the typed events of wikiHow (Koupaee and Wang, 2018), which is an online community contains a collection of professionally edited how-to guideline articles. The types of events are collected by P2GT (Chen et al., 2020). There are 625 event types among 12,795 events. We utilize the events in the testing set of wikiHow with their types as ground truths to validate the event induction performance of our schema induction method.

Relation Typing. There are no datasets designed for the relation typing task, so here we make use of the domain segments separated by "/" in FB15kET (Bordes et al., 2013) to extract the types. These domain segments serve as ground truth types, with the last domain component functioning as the relation itself. There are 607 relation types among 1,345 relations in FB15kET. We utilize the relations in the testing set of FB15kET with their types as ground truths to validate the relation induction performance of our schema induction method.

C.2 Metrics

We employ **BertScore-Recall** and **BertScore-Coverage** as the evaluation metrics, which are denoted as **BS-R** and **BS-C** respectively. They are used to calculate how many types in each instance or entire testing set are recalled by our schema induction method. The BertScore (Zhang et al., 2019), which is denoted as **BS**, between each pair of type and induced schema are calculated as follows:

$$\text{BertRecall} = \frac{1}{|t|} \sum_{t_i \in t} \max_{\hat{t}_j \in \hat{t}} \mathbf{x}_{t_i}^{\top} \mathbf{x}_{\hat{t}_j}, \tag{1}$$

$$\text{BertPrec} = \frac{1}{|\hat{t}|} \sum_{\hat{t}_i \in \hat{t}} \max_{t_j \in t} \mathbf{x}_{t_j}^{\top} \mathbf{x}_{\hat{t}_i}, \tag{2}$$

$$\text{BS}(t, \hat{t}) = 2 \frac{\text{BertRecall} \cdot \text{BertPrec}}{\text{BertRecall} + \text{BertPrec}}, \tag{3}$$

where t and $\hat{t}$ represents the tokens of a ground truth type and induced schema, respectively. The embedding vector of each token t_i or $\hat{t}_j$ of a type t or induced schema $\hat{t}$ is denoted as $\mathbf{x}_{t_i}$ and $\mathbf{x}_{\hat{t}_j}$, which are obtained with RoBERTa (Liu et al., 2019). Then the BS-R and BS-C can be calculated as follows:

$$\text{BS-R}(\mathcal{T}, \hat{\mathcal{T}}) = \frac{1}{|\hat{\mathcal{T}}|} \sum_{\hat{t} \in \hat{\mathcal{T}}} \max_{t \in \mathcal{T}} \text{BS}(t, \hat{t}), \tag{4}$$

$$\text{BS-C}(\mathcal{S}_t, \mathcal{S}_{\hat{t}}) = \frac{1}{|\mathcal{S}_{\hat{t}}|} \sum_{\hat{t} \in \mathcal{S}_{\hat{t}}} \max_{t \in \mathcal{S}_t} \text{BS}(t, \hat{t}), \tag{5}$$

where $\mathcal{T}$ and $\hat{\mathcal{T}}$ represent a set of ground truth types and induced schemas in each testing instance, respectively. Similarly, $\mathcal{S}_t$ and $\mathcal{S}_{\hat{t}}$ denote the set of ground truth types and induced schemas across the entire testing set, respectively.

D Case Study Examples

Figures 9 and 10 demonstrate specific cases where events and concepts are crucial for effective knowledge graph utilization in retrieval-augmented generation. Figure 9 illustrates how event nodes provide essential contextual information that entity-only representations miss, while Figure 10 showcases how concept nodes establish semantic bridges across otherwise disconnected subgraphs, enabling more comprehensive reasoning for complex questions.

E Algorithm for RAG

We include all the algorithms used in our RAG evaluation on various graphs constructed by AutoSchemaKG. Algorithm 1 presents the Think-on-Graph reasoning method that leverages our knowledge graphs for multi-hop question answering. For adapting our entity-event-concept graphs at different scales, we implemented two variants of HippoRAG2: Algorithm 2 for smaller, more focused graph traversal, and Algorithm 3 for large-scale graph exploration with optimized memory management. These adaptations enable efficient navigation of the rich semantic structures in our ATLAS knowledge graphs.

Algorithm 1 Think on Graph (ToG) (Sun et al., 2024a) for Question Answering

```
Require: Knowledge Graph $G$, Query $q$, Top-$N$ parameter, Maximum depth $D_{max}$
Ensure: Answer to query $q$
 1: Extract entities from query $q$ using NER
 2: Retrieve top-$k$ initial nodes from $G$ based on entity similarity
 3: Let $P \leftarrow$ set of paths, each containing a single initial node
 4: $D \leftarrow 0$                                              ▷ Current search depth
 5: while $D \leq D_{max}$ do
 6:     $P \leftarrow$ Search$(q, P, G)$                          ▷ Expand paths by one hop
 7:     $P \leftarrow$ Prune$(q, P, N)$                           ▷ Keep top-$N$ most relevant paths
 8:     if Reasoning$(q, P)$ determines paths sufficient then
 9:         return Generate$(q, P)$                               ▷ Generate answer using paths
10:     end if
11:     $D \leftarrow D + 1$
12: end while
13: return Generate$(q, P)$                       ▷ Generate answer using best available paths
14: procedure SEARCH$(q, P, G)$
15:     $P_{new} \leftarrow \emptyset$
16:     for each path $p \in P$ do
17:         $e_{tail} \leftarrow$ last entity in path $p$
18:         $S \leftarrow$ successors of $e_{tail}$ in $G$ not already in $p$
19:         $R \leftarrow$ predecessors of $e_{tail}$ in $G$ not already in $p$
20:         if $S = \emptyset$ and $R = \emptyset$ then
21:             $P_{new} \leftarrow P_{new} \cup \{p\}$                ▷ Keep dead-end paths
22:         else
23:             for each node $n \in S$ do
24:                 $r \leftarrow$ relation from $e_{tail}$ to $n$ in $G$
25:                 $P_{new} \leftarrow P_{new} \cup \{p + [r, n]\}$       ▷ Extend path forward
26:             end for
27:             for each node $n \in R$ do
28:                 $r \leftarrow$ relation from $n$ to $e_{tail}$ in $G$
29:                 $P_{new} \leftarrow P_{new} \cup \{p + [r, n]\}$       ▷ Extend path backward
30:             end for
31:         end if
32:     end for
33:     return $P_{new}$
34: end procedure
35: procedure PRUNE$(q, P, N)$
36:     Score each path in $P$ using LLM relevance assessment (1-5 scale)
37:     Sort paths by decreasing score
38:     return top-$N$ highest scoring paths
39: end procedure
40: procedure REASONING$(q, P)$
41:     Extract triples from paths in $P$
42:     Ask LLM if triples are sufficient to answer $q$ (Yes/No)
43:     return True if answer is "Yes", False otherwise
44: end procedure
45: procedure GENERATE$(q, P)$
46:     Extract triples from paths in $P$
47:     Prompt LLM with triples and query $q$ to generate answer
48:     return generated answer
49: end procedure
```

Algorithm 2 HippoRAG2 (Gutiérrez et al., 2025)
General algorithm follows the original implementation, while we modify the initialization of graph and embeddings.

```
 1: function INIT(graph_type)
 2:     if graph_type is entity then
 3:         graph, embedding ← $Graph(entity), Embeddings(entity)$
 4:     else if graph_type is entity+event then
 5:         graph, embedding ← $Graph(entity, event), Embeddings(entity, event)$
 6:     else if graph_type is entity+event+concept then
 7:         graph, embedding ← $Graph(entity, event, concept), Embeddings(entity, event, concept)$
 8:     end if
 9: end function
10: function QUERY2EDGE(query, topN)
11:     $Q_{emb}$ ← Retriever(query)
12:     $S = Q \cdot W_e$                    ▷ Calculate similarity scores with precomputed edge embeddings
13:     $E$ = argsort$_i(S)[: N]$                    ▷ Select topN edges based on scores
14:     filtered_edges ← LLM_filter($E$)            ▷ Filter edges using Large Language Model
15:     mapped_edges ← Map_edges(filtered_edges)    ▷ Map filtered edges to original edges
16:     return_node_scores ← Calculate_node_scores(mapped_edges)
17:     return return_node_scores
18: end function
19: function QUERY2PASSAGE(query, weight_adjust)
20:     $Q_{pass}$ ← Encode($query$)                ▷ Encode query into passage representation
21:     $S_{text}$ ← Similarity_Scores($Q_{pass}$, text_embeddings)
22:     return Scores_Dictionary($S_{text}$)
23: end function
24: function RETRIEVE_PERSONALIZATION_DICT(query, topN)
25:     node_dict ← query2edge($query, topN$)
26:     text_dict ← query2passage($query$, weight_adjust)
27:     return node_dict, text_dict
28: end function
29: function RETRIEVE_PASSAGES(query, topN)
30:     node_dict, text_dict ← retrieve_personalization_dict($query, topN$)
31:     if node_dict is empty then
32:         return TopN_Text_Passages(text_dict)
33:     else
34:         personalization_dict ← {node_dict, text_dict}
35:         page_rank_scores ← PageRank(personalization_dict)
36:         return TopN_Passages(page_rank_scores)
37:     end if
38: end function
```

Algorithm 3 LargeKGRetriever
A variant of HippoRAG2 (Gutiérrez et al., 2025), optimized with dynamic graph sampling and common word filtering

```
1:  function INIT(graph_type)
2:      keyword ← Default_graph_keyword                    ▷ Keyword can be cc, pes2o, wiki
3:      Initialize_resources(keyword)
4:      Load_node_and_edge_indexes()
5:  end function
6:  function RETRIEVE_TOPK_NODES(query, top_k_nodes)
7:      entities ← LLM_NER(query)
8:      KG_entities ← Encode_and_Search(entities, FAISS_index)
9:      filtered_keywords ← LLM_filter(KG_entities)
10:     return filtered_keywords
11: end function
12: function RETRIEVE_PERSONALIZATION_DICT(query, number_of_source_nodes)
13:     topk_nodes ← retrieve_topk_nodes($query$, number_of_source_nodes)
14:     if topk_nodes == {} then
15:         return {}
16:     end if
17:     Update personalization dictionary with topk_nodes
18:     return Personalization dictionary
19: end function
20: function PAGERANK(personalization_dict, topN, sampling_area)
21:     $G_{\text{Sample}}$ ← Random Walk with Restart Sampling
22:     $Scores$ = PageRank($G_{\text{Sample}}$, personalization_dict)
23:     topN_nodes = argsort$_i(Scores)[: N]$
24:     for node in topN nodes do
25:         Connected_Passage += node.score
26:     end for
27:     return TopN_Ranked_Passages
28: end function
29: function RETRIEVE_PASSAGES(query, topN, number_of_source_nodes, sampling_area)
30:     personalization_dict ← retrieve_personalization_dict(query, number_of_source_nodes)
31:     if personalization_dict is empty then
32:         return {}, [0]
33:     end if
34:     topN_passages ← pagerank(personalization_dict, topN, sampling_area)
35:     return topN_passages
36: end function
```

F The Recall Metrics in Opendomain QA Tasks

We also use Retrieval Quality metrics at $k \in \{2, 5\}$: PR@$k = |D_k \cap S|/|S|$ where PR@k (Partial Recall) measures the fraction supporting document is in top-k, D_k is the set of top-k retrieved documents, and S is the set of supporting documents. For multi-hop QA datasets (HotpotQA, 2WikiMultihopQA, MuSiQue), these retrieval metrics are crucial as they measure how effectively our system retrieves the evidence needed for multi-step reasoning.

Questions in datasets like 2WikiMultihopQA (Ho et al., 2020) and HotpotQA (Yang et al., 2018) tend to be more entity-centric, with relationships and entities more explicitly represented, which aids retrievers in easily locating relevant subgraphs. In contrast, MuSiQue (Trivedi et al., 2022), due to its questions' increased complexity in both description and multi-hop nature, poses greater challenges for retrieval. Additionally, differences in graph construction cause the retrievers to perform differently across datasets.

G Details and Full Results on General Benchmarks

G.1 Implementation and Evaluation Details on FELM

For the evaluation metrics, we follow the original paper (Chen et al., 2023) and use balanced accuracy and F1 score to evaluate the factuality checking capability. For the classification of segments in an instance, we ask the model to generate the ID of false segments, and then get the true positive (TP), false positive (FP), true negative (TN) and false negative (FN) results. The balanced accuracy is calculated as:

$$\text{Balanced Accuracy} = \frac{TP}{TP + FN} + \frac{TN}{TN + FP} \quad (6)$$

Since we use F1 score to evaluate the factual error detection capability, we calculate the F1 score as:

$$\text{F1} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \quad (7)$$

where Precision = $\frac{TN}{TN+FN}$ and Recall = $\frac{TN}{TN+FP}$.

We use the Retrieval-Augmented Generation method with different knowledge bases on 3 domains (world knowledge, science and technology, and writing/recommendation) of FELM benchmark. For the math domain and reasoning domain, we use the vanilla setting and their results are the same across different knowledge bases. The detailed results of the 3 domains are shown in Table 10.

G.2 Implementation and Evaluation Details on MMLU

Table 11 presents our classification for organizing MMLU tasks into distinct subject categories, providing a structured framework for domain-specific performance analysis. Table 12 displays comprehensive results across all MMLU subject areas, revealing an important insight: while retrieval-augmented generation enhances performance in knowledge-intensive domains, it can negatively impact performance on reasoning-focused tasks such as mathematics and logical reasoning. This finding aligns with previous research suggesting that RAG may sometimes interfere with LLMs' inherent reasoning capabilities.

Model/Dataset	**MuSiQue**		**2Wiki**		**HotpotQA**	
Metric	**Recall@2**	**Recall@5**	**Recall@2**	**Recall@5**	**Recall@2**	**Recall@5**
Baseline Retrievers						
Contriever	34.8	46.6	46.6	57.5	58.4	75.3
BM25	32.4	43.5	55.3	65.3	57.3	74.8
LLM Embeddings						
GTE-Qwen2-7B-Instruct	48.1	63.6	66.7	74.8	75.8	89.1
GritLM-7B	49.7	65.9	67.3	76.0	79.2	92.4
NV-Embed-v2 (7B)	52.7	69.7	67.1	76.5	84.1	94.5
Existing Graph-based RAG Methods						
RAPTOR (Llama-3.3-70B-Instruct)	47.0	57.8	58.3	66.2	76.8	86.9
HippoRAG (Llama-3.3-70B-Instruct)	41.2	53.2	71.9	90.4	60.4	77.3
HippoRAG2 (Llama-3.3-70B-Instruct)	56.1	74.7	76.2	90.4	83.5	96.3
AutoSchemaKG (`LLama-3.1-8B-Instruct`*) + HippoRAG1*						
Entity-KG (Llama-3-8B-Instruct)	41.37	51.08	61.72	75.45	51.89	65.95
Entity-Event-KG (Llama-3-8B-Instruct)	41.28	51.12	61.37	74.56	51.31	65.93
Full-KG (Llama-3-8B-Instruct)	40.78	50.36	61.08	71.9	52.8	65.4
AutoSchemaKG (`LLama-3.1-8B-Instruct`*) + HippoRAG2*						
Entity-KG (Llama-3-8B-Instruct)	48.33	72.58	67.34	84.25	77.59	92.16
Entity-Event-KG (Llama-3-8B-Instruct)	48.83	72.7	68.59	85.85	81.26	92.66
Full-KG (Llama-3-8B-Instruct)	49.12	72.48	68.46	84.6	84.17	93.04

Table 8: Recall @ 2 and Recall @ 5.

Table 9: Recall performance in the knowledge graph created by Llama-3-8B-Instruct shows strong performance that is comparable with the knowledge graph created with 70B model.

Corpus	**Method**	**World Knowledge**				**Science and Technology**				**Writing/Recommendation**			
		P	R	F1	Acc	P	R	F1	Acc	P	R	F1	Acc
-	-	36.67	29.93	32.96	55.10	15.43	24.51	18.94	50.49	25.95	12.73	17.09	52.69
Text Corpora													
Wikipedia	Random	31.78	27.89	29.71	52.52	7.95	11.76	9.49	43.94	21.95	26.97	24.20	53.78
	BM25	26.82	32.65	29.45	49.31	15.23	38.24	21.79	50.48	29.21	48.69	36.52	62.40
	Dense Retrieval	33.93	38.78	36.19	54.97	16.92	43.14	24.31	53.01	25.22	43.45	31.91	58.68
Pes2o-Abstract	Random	27.36	19.73	22.92	49.86	10.13	15.69	12.31	45.64	26.92	28.84	27.85	56.50
	BM25	32.43	32.65	32.54	53.34	11.18	17.65	13.69	46.54	26.75	32.96	29.53	57.34
	Dense Retrieval	31.43	29.93	30.66	52.50	20.51	47.06	28.57	57.55	25.37	32.21	28.38	56.51
Common Crawl	Random	30.14	29.93	30.03	51.72	11.17	21.57	14.72	45.75	23.73	28.09	25.73	54.91
	BM25	27.21	27.21	27.21	49.71	12.21	25.49	16.51	46.68	27.32	40.82	32.73	59.42
	Dense Retrieval	25.50	34.69	29.39	48.00	15.48	36.27	21.70	50.78	24.41	38.95	30.01	57.27
Knowledge Graph													
Freebase	Think on Graph	26.00	8.84	13.20	49.62	23.76	23.53	23.65	55.15	42.86	12.36	19.19	54.51
ATLAS-Wiki	HippoRAG2	33.33	42.18	37.24	54.98	16.45	50.00	24.76	52.75	32.82	32.21	32.51	59.43
ATLAS-Pes2o		39.17	31.97	35.21	56.51	21.35	40.20	27.89	57.13	30.37	15.36	20.40	54.11
ATLAT-CC		33.80	48.98	40.00	56.18	18.06	52.94	26.93	55.42	24.20	25.47	24.82	54.66

Table 10: Factuality results (%) on different domains of FELM benchmark with different Text Corporas and retrieval methods. P, R, F1, and Acc denote Precision, Recall, F1 score, and Balanced Accuracy, respectively.

Subject	Task
History	high school european history, high school us history, high school world history, prehistory
Formal Logic	formal logic, logical fallacies
Law	international law, jurisprudence, professional law
Philosophy and Ethics	philosophy, moral disputes, moral scenarios, business ethics
Religion	world religions
Medicine and Health	clinical knowledge, college medicine, medical genetics, professional medicine, virology, human aging, nutrition, anatomy
Social Sciences	high school geography, high school government and politics, high school psychology, professional psychology, sociology, human sexuality, us foreign policy, security studies
Economics	high school macroeconomics, high school microeconomics, econometrics
Business and Management	management, marketing, professional accounting, public relations
Math	abstract algebra, college mathematics, elementary mathematics, high school mathematics, high school statistics
Natural Sciences	astronomy, college biology, college chemistry, college physics, conceptual physics, high school biology, high school chemistry, high school physics
Computer Science and Engineering	college computer science, high school computer science, computer security, electrical engineering, machine learning
Global Facts	global facts, miscellaneous

Table 11: The correspondence between subjects and tasks.

Corpus	MMLU													
	overall	History	Law	Religion	PaE	MaH	GF	BaM	SS	Logic	Econ	Math	NS	CSaE
None	69.18	76.59	66.86	83.04	63.55	70.38	66.72	72.20	79.74	64.35	68.35	57.31	65.27	66.70
Freebase-ToG	**70.36**	**78.42**	69.00	75.44	65.67	72.65	67.27	**73.67**	76.00	66.03	67.34	60.56	69.39	**68.23**
Random Baseline														
Wikipedia	68.06	76.64	66.82	79.53	59.26	70.34	66.46	67.34	77.78	59.21	65.35	60.91	67.52	61.87
Common Crawl	67.93	74.89	66.52	79.53	61.74	69.82	68.11	67.67	77.52	59.30	64.20	59.80	67.42	62.22
Pes2o-Abstract	68.07	76.24	64.16	80.70	62.01	70.62	66.59	69.27	77.16	62.39	64.70	60.07	66.41	62.18
Text Corpora + DBM25														
Wikipedia	68.99	76.67	67.35	78.36	63.34	69.35	61.98	71.39	76.99	62.30	65.56	61.67	69.31	65.60
Common Crawl	68.33	76.15	66.36	80.12	60.43	69.58	64.67	69.47	76.71	63.18	68.22	62.26	65.55	65.04
Pes2o-Abstract	69.04	78.01	65.89	78.95	63.83	71.01	65.78	68.29	77.07	59.34	**68.87**	61.20	67.73	65.74
Text Corpora + Dense Retrieval														
Wikipedia	69.37	73.59	69.60	79.53	63.58	70.82	62.41	72.57	76.83	62.21	67.35	61.79	**69.81**	65.39
Common Crawl	67.03	74.47	68.98	79.53	60.46	69.29	64.09	68.88	75.21	61.86	62.27	57.13	64.54	64.47
Pes2o-Abstract	69.07	75.79	61.82	78.36	65.15	69.72	66.77	69.02	76.47	63.05	63.07	**63.92**	69.53	67.86
ATLAS + HippoRAG2														
ATLAS-Wiki	68.22	76.73	67.38	84.21	**66.01**	70.82	68.36	72.35	79.16	63.65	64.53	50.09	65.45	62.10
ATLAS-CC	68.26	78.16	**70.85**	83.04	65.60	71.28	63.95	68.84	78.16	65.42	67.51	52.87	63.32	63.40
ATLAS-Pes2o	69.19	77.13	68.41	81.29	65.05	**72.75**	65.67	72.32	81.19	62.98	65.29	54.24	65.41	64.04
ATLAS + ToG														
ATLAS-Wiki	68.29	77.91	66.60	84.21	65.10	70.69	63.85	70.49	78.31	67.08	66.24	54.41	63.65	64.18
ATLAS-CC	68.40	77.07	68.18	83.63	65.24	72.03	66.87	71.21	79.72	66.59	66.42	48.74	63.97	64.22
ATLAS-Pes2o	68.97	77.52	66.95	**84.80**	63.44	71.15	**68.92**	69.98	**81.59**	**67.87**	66.17	55.46	63.35	64.75

Table 12: Performance comparison of our knowledge graph (KG) integrated with HippoRAG2 and ToG against baseline retrieval methods (Random, BM25, Dense Retrieval) across Wikipedia, Common Crawl, and Pes2o-Abstract corpora on MMLU benchmarks. Tasks are classified according to subjects, with bold and underline indicating the highest and the second highest performance. PaE, MaH, GF, BaM, SS, Econ, NS and CSaE represent Philosophy_and_Ethics, Medicine_and_Health, Global_Facts, Business_and_Management, Social_Sciences, Economics, Natural_Sciences and Computer_Science_and_Engineering respectively.

Multiple-Choice Question Generation and Answering

MCQ Generation Prompt:
You are an expert in generating multiple-choice questions (MCQs) from scientific texts. Your task is to generate 5 multiple-choice questions based on the following passage.
Each question should:
- Focus on factual claims, numerical data, definitions, or relational knowledge from the passage.
- Have 4 options (one correct answer and three plausible distractors).
- Clearly indicate the correct answer.

The output should be in JSON format, with each question as a dictionary containing:
- "question": The MCQ question.
- "options": A list of 4 options (e.g., ["A: ..", "B: ..", "C: ..", "D: .."]).
- "answer": The correct answer (e.g., "A").

Output Example:
```
[
  {
    "question": "What is the primary role of a catalyst in a chemical reaction?",
    "options": [
      "A: To make a thermodynamically unfavorable reaction proceed",
      "B: To provide a lower energy pathway between reactants and products",
      "C: To decrease the rate of a chemical reaction",
      "D: To change the overall reaction itself"
    ],
    "answer": "B"
  }
]
```
Passage: {passage}

MCQ Answering Prompt:
Given the contexts or evidences: {contexts}
Here is a multiple-choice question:
Question: {question}
Options: A. {options_0} B. {options_1} C. {options_2} D. {options_3}
Please select the correct answer by choosing A, B, C, or D. Respond with only the letter of your choice.

Figure 8: The prompts for generating and answering MCQ questions for evaluating knowledge retention in knowledge graph.

Question: When did the country the top-ranking Warsaw Pact operatives came from, despite it being headquartered in the country where A Generation is set, agree to a unified Germany inside NATO?

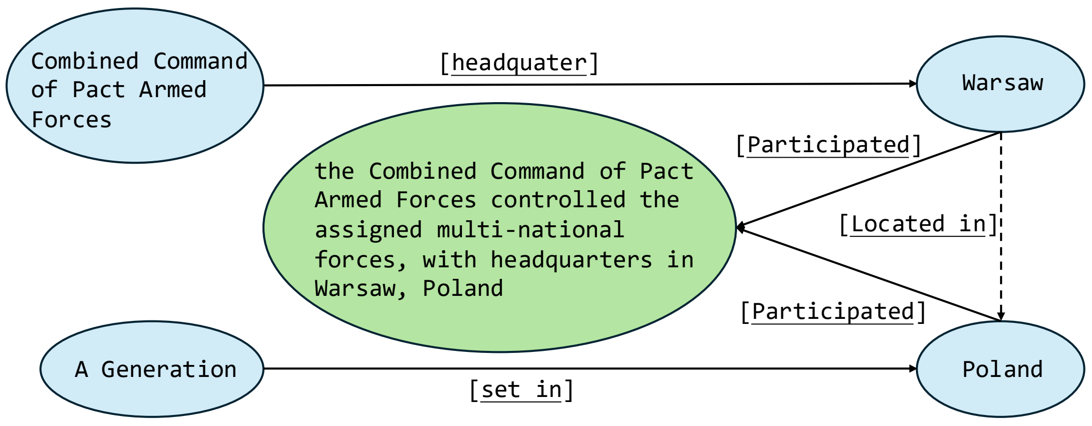
Combined Command of Pact Armed Forces
[headquater]
Warsaw
[Participated]
the Combined Command of Pact Armed Forces controlled the assigned multi-national forces, with headquarters in Warsaw, Poland
[Located in]
[Participated]
A Generation
[set in]
Poland

Figure 9: Event Node (green) offers enriched context over triplets (blue); dotted line indicates missing edge

Question: Who won the Indy Car Race in the largest populated city of the state where the performer of Mingus Three is from?

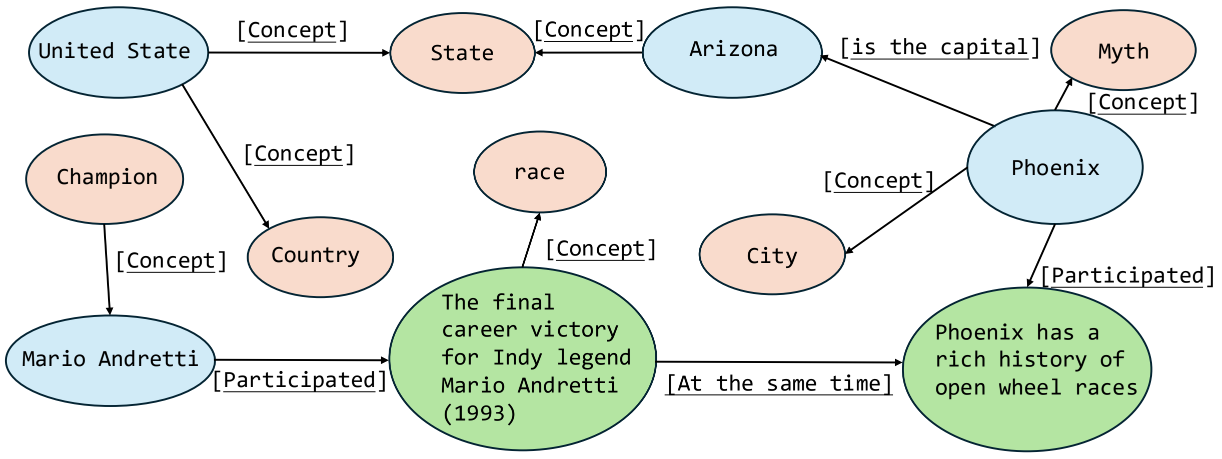

Figure 10: Concept nodes (orange) provide alternate pathways to access information beyond entities and events.