Knowledge Graph Construction: From Semantic Richness to Task-Aware Optimization

Bridging Intentions, Schemas, and Downstream Utility in Modern KG Systems

Jiaxin Bai
Assistant Professor
Hong Kong Baptist University
April 2026

Why Knowledge Graph Construction Matters Now

Knowledge Graphs (KGs) have become the **semantic backbone** of many modern AI systems, powering search engines, recommendation systems, and question answering.

Yet, the fundamental question of **HOW** to build them effectively remains an active and evolving research frontier.

This talk traces a research arc across three recent works that collectively address the core challenges:

- **What knowledge to represent?**
- **How to organize it without human bottlenecks?**
- **How to ensure the resulting graph is actually useful for downstream tasks?**

Roadmap: Three Pillars of Modern KG Construction

The talk is organized around three interconnected **questions**:

1. Semantic Richness

How do we capture not just entities, but intentions, events, and their complex relationships? ***(Intention KG / RIG)***

2. Autonomous Construction

How do we build KGs at web scale without predefined schemas? ***(AutoSchemaKG / ATLAS)***

3. Task-Aware Optimization

How do we ensure the KG is not just "good" but "useful" for downstream tasks? ***(AutoGraph-R1)***

From Manual Schemas to Autonomous, Task-Optimized Construction

Era 1: Manual/Crowdsourced KGs (Freebase, ConceptNet, ATOMIC)
High quality but limited scale, requires human experts.

Era 2: LLM-Assisted Extraction (FolkScope, COSMO)
LLMs generate knowledge, but still rely on predefined schemas.

Era 3: Autonomous Construction (AutoSchemaKG)
No predefined schemas, dynamic schema induction.

Era 4: Task-Optimized Construction (AutoGraph-R1)
RL directly optimizes KG for downstream utility.

This talk covers the transition from Era 2 to Era 4.

The Unifying Thread: Conceptualization as a Bridge

A key technique shared across all three works is **conceptualization** — the process of abstracting specific instances into broader semantic categories.

Intention KG

"vampire costume" is abstracted to "costumes" and "Halloween."

Purpose: Connecting related intentions.

AutoSchemaKG

"Black Mountain College" is conceptualized into "College," "School," "Institution."

Purpose: Inducing schemas and enabling cross-domain reasoning.

Conceptualization transforms sparse, disconnected facts into a coherent knowledge ecosystem.

Part I: Capturing the "Why" Behind User Behavior

Intention Knowledge Graph Construction for User Intention Relation Modeling

Jiaxin Bai, Zhaobo Wang, Junfei Cheng, Dan Yu, et al.

EACL 2026

Existing KGs Miss the Deliberate Nature of User Behavior

Traditional e-commerce KGs (Amazon Product Graph, AliCG, AliCoCo) effectively represent **item properties** but fail to capture **user intentions**.

Recent works like FolkScope and COSMO incorporate **intentions through LLMs**, but they treat intentions as isolated nodes — they **lack inter-intention relationships**.

> **Example: Preparing for Halloween**
>
> A user has interconnected intentions: dressing as a vampire, creating decorations, hosting a party. These intentions are **temporally**, **causally**, and **conceptually** linked.

This connects to Daniel Kahneman's **System I vs. System II** reasoning distinction — online shopping often reflects deliberate, goal-oriented behavior.

Can We Model How User Intentions Relate to Each Other?

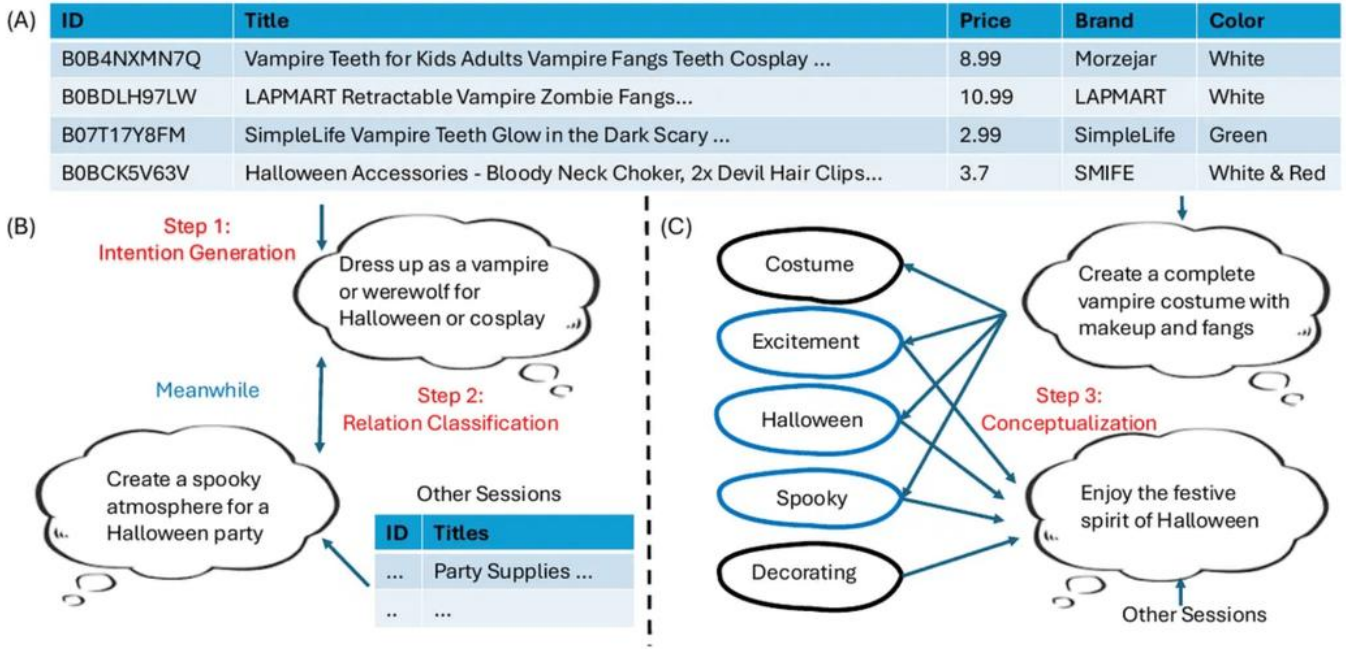
The key insight is that understanding user behavior requires modeling not just **WHAT** users intend, but **HOW** their intentions connect.

Two mechanisms drive these connections:

1. **Discourse-based Commonsense Knowledge** — Planning a Halloween party naturally entails costume selection AND decoration preparation (temporal/causal links).
2. **Abstract Conceptualization** — A "vampire costume" intention connects to other costume-related intentions through shared concepts.

This leads to the IGC-RC framework: Intention Generation, Conceptualization, and Relation Classification.

IGC-RC: A Three-Step Framework for Intention KG Construction



Step 1: Intention Generation — Use GPT-3.5 to generate plausible user intentions from session data.

Step 2: Relation Classification — Template-based assertion generation + Vera model to classify temporal and causal relationships.

Step 3: Conceptualization — Abstract intentions into higher-level concepts using LLMs.

Extracting User Intentions from Browsing Behavior

Using the Amazon M2 dataset (1.2M English sessions), the system extracts product attributes and uses GPT-3.5 to generate 4.3 million diverse user intentions.

The approach improves upon FolkScope by using a more capable LLM and removing restrictive relation constraints, resulting in more natural intentions.

Examples of generated intentions:

- "Purchase a construction dump truck toy for a 2-year-old"
- "Relieve discomfort and soothe itching caused by hemorrhoids"
- "Purchase unscented baby wipes for sensitive skin"

Human evaluation shows plausibility of 0.9552 vs. FolkScope's 0.6116.

Classifying Temporal, Causal, and Co-occurrence Relations

Three-step approach to establish connections:

1. **Template-based Assertion Generation** — Transform intention pairs into natural language assertions expressing five relationship types: precedence, succession, simultaneity, cause, and result.
2. **Plausibility Estimation** — Fine-tuned Vera model assesses plausibility.
3. **Threshold-based Edge Selection** — Retain only high-confidence relationships (VERA-Verification score > 0.9).

	Precedence			Succession			Simultaneous			Cause			Result			Overall		
	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
Vera	0.73	0.52	0.61	0.78	0.66	0.72	0.75	0.83	0.79	0.75	0.73	0.74	0.76	0.92	0.83	0.75	0.73	0.74
+ Fine-tuning	**0.86**	**0.97**	**0.91**	**0.88**	**0.92**	**0.90**	**0.87**	**0.93**	**0.90**	**0.84**	**0.91**	**0.88**	**0.81**	**0.99**	**0.89**	**0.85**	**0.94**	**0.89**

Abstracting Intentions into Semantic Concepts

Each intention is mapped to broader concepts following three principles: relevance, unambiguity, and complementarity.

Intention	Concepts
Purchase a construction dump truck toy for a 2-year-old	playtime, construction, gift
Enjoy multiplayer gameplay with friends	socializing, competition, fun
Personalize their drawstring bags	personalization, gift, accessorizing
Perform sanding and grinding tasks	smoothing, surface prep, precision

Conceptualization performance: **86.60%** correctness with **77.19%** inter-annotator agreement.

RIG: 351 Million Edges Capturing Intention Dynamics

The constructed Relational Intention Graph (RIG) operates at **massive scale**:

# Sessions	# Concepts	# Intentions	# Ses.-Int.	# Int.-Con.	# Int.-Int.	# Nodes	# Edges
1,176,296	110,741	2,956,195	5,115,587	5,115,212	341,649,216	4,243,232	351,880,015

Six edge types: session-to-intention, intention-to-concept, and three types of intention-to-intention (asynchronous, synchronous, causality).

Human Evaluation Confirms High Quality at Massive Scale

Compared with existing commonsense KGs:

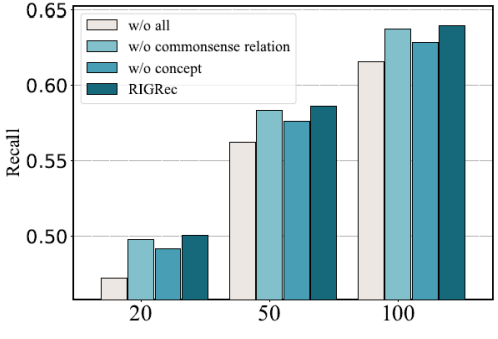
Knowledge Graph	Acceptance	Size
Atomic2020	**86.8**	0.6M
Atomic10x	78.5	6.5M
Atomic NOVA	-	2.1M
RIG (ours)	81.2	**341.6M**
- Asynchronous Relation	80.6	100.2M
- Synchronous Relation	82.8	112.6M
- Causality Relation	80.4	128.7M

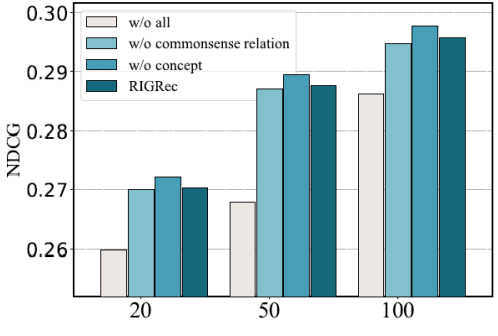
RIG achieves 81.2% acceptance rate while being 500x larger than ATOMIC 2020. The graph maintains quality at scale — a critical achievement for practical deployment.

RIG Significantly Enhances Recommendation Performance

RIGRec, a recommendation model leveraging the intention KG, outperforms 14 baselines across all metrics on the Amazon M2 dataset.

Datasets	Metric	FPMC	GRU4Rec	BERT4Rec	SASRec	SASRecF	CORE	SR-GNN	GCE-GNN	DIF-SR	FEARec	DGNN	ISRec	Satori	KA-MemNN	RIGRec
M2 (UK)	Recall@5	0.2523	0.2792	0.1899	0.3075	0.2957	0.2990	0.2928	0.3130	0.3128	0.3088	0.3021	0.3073	0.2973	0.2932	**0.3342***
	Recall@10	0.3121	0.3469	0.2641	0.3964	0.3713	0.3949	0.3678	0.4001	0.3990	0.3941	0.3882	0.3981	0.3821	0.3781	**0.4229***
	Recall@20	0.3696	0.4108	0.3349	0.4723	0.4406	0.4768	0.4381	0.4726	0.4739	0.4691	0.4563	0.4754	0.4436	0.4419	**0.5003***
	Recall@50	0.4389	0.4865	0.4197	0.5621	0.5245	0.5697	0.5171	0.5542	0.5598	0.5552	0.5584	0.5676	0.5348	0.5295	**0.5863***
	Recall@100	0.4841	0.5346	0.4744	0.6159	0.5771	0.6223	0.5662	0.6032	0.6171	0.6100	0.6072	0.6201	0.5931	0.5847	**0.6398***
	NDCG@5	0.1933	0.2118	0.1260	0.2121	0.2208	0.1673	0.2195	0.2214	0.2171	0.2138	0.2201	0.2207	0.2189	0.2163	**0.2214**
	NDCG@10	0.2126	0.2327	0.1501	0.2406	0.2432	0.1985	0.2418	0.2441	0.2451	0.2415	0.2431	0.2438	0.2425	0.2386	**0.2503***
	NDCG@20	0.2272	0.2499	0.1682	0.2598	0.2634	0.2193	0.2616	0.2626	0.2641	0.2605	0.2637	0.2633	0.2623	0.2591	**0.2703***
	NDCG@50	0.2411	0.2648	0.1848	0.2679	0.2801	0.2379	0.2784	0.2807	0.2812	0.2777	0.2797	0.2795	0.2793	0.2737	**0.2877***
	NDCG@100	0.2484	0.2718	0.1937	0.2862	0.2887	0.2472	0.2865	0.2871	0.2891	0.2866	0.2867	0.2881	0.2864	0.2843	**0.2957***





Ablation studies show both conceptualization and commonsense relations contribute uniquely to the performance gains.

Lessons from Intention KG Construction

Three key takeaways:

1. **Beyond entities:** KGs should capture higher-level semantic units (intentions, goals) — not just items and their properties.
2. **Relations matter:** Modeling HOW knowledge nodes connect (temporal, causal, conceptual) is as important as the nodes themselves.
3. **LLMs as knowledge generators:** LLMs can effectively generate and classify commonsense knowledge at scale.

However, this work is domain-specific (e-commerce) and relies on predefined relation types. This raises the next question: **Can we build KGs autonomously across ANY domain?**

The Next Challenge: Scaling Beyond Predefined Schemas

The Intention KG demonstrates the power of rich semantic representations, but it operates within a specific domain with predefined relations.

To achieve true web-scale knowledge representation, we must eliminate the human bottleneck of schema design.

Part II: Autonomous KG Construction at Web Scale

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

Jiaxin Bai, Wei Fan, Qi Hu, Qing Zong, et al.

ACL 2026

The Inherent Paradox of Traditional KG Construction

Current KG construction approaches face an inherent paradox: they require predefined schemas created by domain experts, which fundamentally limits their scalability, adaptability, and domain coverage.

- Traditional approaches (Freebase, YAGO) start with a manually designed ontology.
- Even recent LLM-based methods (PiVE, iText2KG, GraphRAG) still rely on fixed prompts or heuristics.

AutoSchemaKG breaks this dependency by enabling fully autonomous construction — the system simultaneously extracts knowledge triples AND induces schemas directly from text.

Real-World Knowledge Is Dynamic, Not Just Static Entities

A critical innovation of AutoSchemaKG is modeling **EVENTS** alongside entities.

Traditional KGs capture static facts ("Paris is the capital of France"), but miss dynamic knowledge — temporal relationships, causality, and procedural knowledge.

AutoSchemaKG treats events as first-class semantic units.

Experiments confirm that with event information, AutoSchemaKG preserves over 90% of original passage content versus just 70% for entity information alone.

This echoes the Intention KG's insight that capturing dynamic, relational knowledge is crucial.

Formalizing Knowledge Graphs with Conceptual Schemas

A Knowledge Graph with Conceptual Schema is defined as **$G = (V, E, C, \varphi, \psi)$**, where:

- **$V = V_E + V_N$** — nodes include event nodes (V_E) and entity nodes (V_N)
- **$E \subseteq V \times V \times R$** — edges connecting entity-entity, entity-event, or event-event nodes
- **C** — set of conceptual categories
- **$\varphi: V \rightarrow P(C)$** — assigns each node a subset of concepts
- **$\psi: R \rightarrow P(C)$** — links each relation type to concept subsets

Key constraint: Every node and relation must have at least one concept assignment. This formalization enables schema induction as a natural part of construction.

Four-Phase Pipeline for Autonomous KG Construction

Phase 1: Input Processing
Source selection (Wikipedia, Semantic Scholar, Common Crawl), document segmentation, batching.

Phase 2: Triple Extraction
Three sequential LLM-based stages: Entity-Entity, Entity-Event, and Event-Event extraction.

Phase 3: Schema Induction
Conceptualize entities, events, and relations into abstract categories using LLMs (no predefined ontology).

Phase 4: KG Construction
Build nodes (entity, event, concept), establish edges, link concepts.

Multi-Stage LLM-Based Triple Extraction

The extraction pipeline uses LLaMA-3-8B-Instruct with optimized precision :

- **Stage 1:** Entity-Entity relationships — e.g., (Paris, capital_of, France)
- **Stage 2:** Entity-Event relationships — e.g., (Napoleon, led, Battle of Waterloo)
- **Stage 3:** Event-Event relationships — e.g., (Battle of Waterloo, caused, Exile to St. Helena)

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

Triple quality across ATLAS-Wiki shows Precision/Recall/F1 >90% for most types.

Dynamic Schema Induction Without Human Intervention

Rather than simply extracting triples, AutoSchemaKG employs a sophisticated abstraction mechanism. For each entity, event, and relation, the LLM generates at least three conceptual phrases at varying abstraction levels.

Examples:

- **Entity:** "Black Mountain College" → College, School, Institution
- **Event:** "A cat chased a prey" → Hunting, Predation, Pursuit
- **Relation:** "participates" → Involvement, Engagement, Participation

Schema induction achieves 95% semantic alignment with human-crafted schemas (BS-R metric) with zero manual intervention.

	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

ATLAS: Billion-Scale KGs from Web Corpora

Processing over 50 million documents from Dolma 1.7 resulted in the ATLAS family of knowledge graphs:

	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**

Total: 900+ million nodes, 5.9 billion edges.

Computational cost: ~78,400 GPU hours.

Events Preserve Over 90% of Original Passage Information

MCQ-based evaluation of information preservation demonstrates the critical importance of events:

Event-level triples preserve dramatically more information than entity-level triples. This validates the design choice of modeling events as first-class citizens in the KG.

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	92.69	94.82
	Event + Entity	**93.30**	**95.13**
ATLAS-Pes2o	[Lower-Upper]	62.32-98.99	75.05-99.49
	Entity	80.00	83.33
	Event	96.97	97.78
	Event + Entity	**97.37**	**97.98**
ATLAS-CC	[Lower-Upper]	56.08-97.29	70.25-99.10
	Entity	76.78	81.01
	Event	94.87	96.78
	Event + Entity	**96.28**	**96.98**

AutoSchemaKG Outperforms Baselines on Multi-Hop QA

With HippoRAG2 integration on three benchmark datasets:

The Full-KG configuration (entities + events + concepts) consistently outperforms entity-only or event-only variants.

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**

KGs Complement Parametric Knowledge in LLMs

On the FELM benchmark, ATLAS-Wiki with HippoRAG2 achieves 56.43% balanced accuracy (best among all methods).

On MMLU benchmarks, ATLAS KGs improve performance in knowledge-intensive domains: History (+1.6%), Law (+4.0%), Religion (+1.2%), Philosophy/Ethics (+2.5%).

This addresses a fundamental question: Can KGs enhance LLMs even when built from the same pretraining data? Yes — structured representations offer advantages over raw text.

Corpus	Method	Acc	F1
-	-	54.08	26.79
Text Corpora			
Wikipedia	Random	52.77	25.56
	BM25	56.15	30.43
	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

Lessons from Autonomous KG Construction

Four key takeaways:

1. **Schema induction works:** Automated schemas achieve 95% alignment with human-crafted one
2. **Events are essential:** Modeling events alongside entities preserves 90%+ of passage information vs. 70% for entities alone.
3. **Scale matters:** KGs must reach billions of facts to effectively complement LLM parametric knowledge.
4. **Conceptualization is the bridge:** It connects disparate information, enables cross-domain reasoning, and reduces sparsity.

But a critical question remains: **How to further improve graphs actually optimal for very specific downstream tasks?**

The Critical Gap: "Good" Graphs vs. "Useful" Graphs

Both the Intention KG and AutoSchemaKG are evaluated **primarily on** intrinsic quality metrics (precision, recall, plausibility) and then **separately tested on** downstream tasks, and hopefully, it can work well.

But there is a fundamental disconnect: a graph optimized for intrinsic quality may be **structurally suboptimal** for its intended application. The **construction** process is decoupled from the **application** — the graph, once built, cannot learn from its failures.

How do we close this loop?

Part III: Learning to Build Useful Knowledge Graphs

AutoGraph-R1: End-to-End Reinforcement Learning for Knowledge Graph Construction

Hong Ting Tsang, Jiaxin Bai, Haoyu Huang, et al.

ACL 2026 (HKBU + HKUST + Cornell + Microsoft Research)

Why Intrinsically "Good" Graphs Can Fail Downstream

Consider the Golden Gate Bridge example:

Standard LLM Extraction

Golden Gate Bridge → connects → San Francisco → center of → Northern California → is in → California → has government → Republic.

The reasoning path is 4+ hops — too long and fragile. Retriever fails.

RL-Optimized Extraction

Golden Gate Bridge → is located in → California → has government → Republic.

The reasoning path is 2 hops — short and robust. Retriever succeeds.

Both graphs are factually correct, but the RL-optimized one is structurally superior for the downstream QA task.

Graphs as Knowledge Carriers vs. Knowledge Indices

AutoGraph-R1 recognizes that KGs serve two distinct functions in RAG pipelines:

1. Knowledge Carriers

The graph itself is the information source. Triples and subgraphs are retrieved and used directly for reasoning. (Methods: Think-on-Graph, SubgraphRAG)

Structural need: Relational completeness.

2. Knowledge Indices

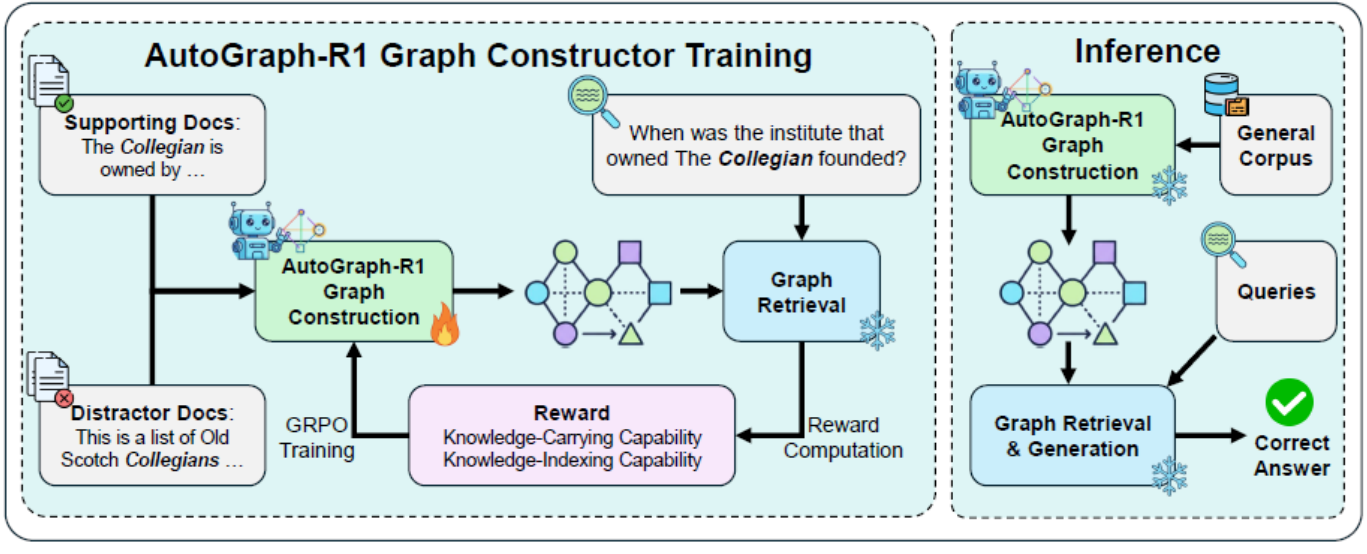
The graph organizes and connects raw text chunks. Graph structure guides retrieval of original passages. (Methods: HippoRAG, HippoRAG2)

Structural need: Effective connectivity to relevant passages.

AutoGraph-R1 designs separate reward functions for each role.

End-to-End RL for KG Construction

AutoGraph-R1 frames KG construction as a reinforcement learning problem:



The training loop: Generate graph → Retrieve from graph → Evaluate answer → Compute reward → Update constructor via GRPO.

The key innovation: the reward comes from downstream task performance, not intrinsic graph quality.

Two Novel Reward Functions for Graph Utility

These rewards are more direct and stable than using final answer F1 scores, which are brittle and noisy.

- **Knowledge-Carrying Reward (R_C):** Measures whether the gold answer is deducible from the constructed KG.

$$R_{\mathrm{C}}(q, y, \mathcal{G}) = \mathbb{I}\big[\mathit{deducible}(q, y \mid \mathcal{G})\big].$$

- **Knowledge-Indexing Reward (R_I):** Measures how effectively the graph indexes relevant text passages (passage recall).

$$R_{\mathrm{I}}(q, \mathcal{D}_{\mathrm{gold}}, \mathcal{G}) = \frac{|\text{Top-}k(\mathcal{G}, q) \cap \mathcal{D}_{\mathrm{gold}}|}{|\mathcal{D}_{\mathrm{gold}}|}$$

Group Relative Policy Optimization for KG Construction

AutoGraph-R1 uses GRPO (Group Relative Policy Optimization) — a memory-efficient alternative to PPO that eliminates the need for a separate value model.

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{\mathbf{s}\sim\mathcal{D},\{\mathbf{a}_i\}_{i=1}^{G}\sim\pi_{\theta_{\text{old}}}(\cdot|\mathbf{s})} \left[\frac{1}{G}\sum_{i=1}^{G}\sum_{t=1}^{|\mathbf{a}_i|} \min\left(r_{i,t}(\theta)\hat{A}_i, \text{clip}(r_{i,t}(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_i \right) \right]$$

For each training sample, the policy generates multiple graph samples. The reward is normalized relative to the group's mean and standard deviation, providing a stable learning signal.

RL-Optimized Graphs Dramatically Improve Graph RAG

When KG acts as a knowledge carrier (Knowledge-Carrying Reward):

Consistent improvements across all retrieval methods (Subgraph, Triples Retriever, ToG) and all model families (Qwen, Llama). Up to +9.69 average F1 gain with Llama-3B.

Methods	Simple QA		Multihop QA			Overall
	NQ*	PopQA*	HotpotQA	2WikiQA	Musique	Avg.
Qwen2.5-3B						
Subgraph (Base)	26.43	54.48	39.02	34.15	13.82	33.58
Subgraph (Ours)	**28.03**↑1.60	**59.46**↑4.98	**40.77**↑1.75	**34.71**↑0.56	**15.13**↑1.31	**35.62**↑2.04
Triples Retriever (Base)	30.53	51.67	40.76	32.18	17.81	34.58
Triples Retriever (Ours)	**33.67**↑3.14	**56.76**↑5.09	**46.94**↑6.18	**36.09**↑3.91	**21.41**↑3.60	**38.97**↑4.39
ToG (Base)	26.32	54.92	41.77	43.54	18.21	36.95
ToG (Ours)	**29.27**↑2.95	**61.40**↑6.48	**44.56**↑2.79	**49.33**↑5.79	**18.42**↑0.21	**40.60**↑3.65
Qwen2.5-7B						
Subgraph (Base)	28.07	55.43	41.66	33.97	15.24	34.87
Subgraph (Ours)	**28.54**↑0.47	**60.94**↑5.51	**43.59**↑1.93	**37.43**↑3.46	**15.65**↑0.41	**37.23**↑2.36
Triples Retriever (Base)	33.26	55.56	44.99	35.57	20.43	37.96
Triples Retriever (Ours)	**33.98**↑0.72	**58.02**↑2.46	**48.28**↑3.29	**36.04**↑0.47	**20.56**↑0.13	**39.38**↑1.42
ToG (Base)	25.59	57.53	43.93	46.03	18.46	38.31
ToG (Ours)	**29.36**↑3.77	**62.85**↑5.32	**44.68**↑0.75	**50.20**↑4.17	**19.31**↑0.85	**41.28**↑2.97
Llama-3.2-1B						
Subgraph (Base)	23.17	26.14	27.11	19.36	9.13	20.98
Subgraph (Ours)	**25.72**↑2.55	**47.32**↑21.18	**35.84**↑8.73	**25.05**↑5.69	**11.90**↑2.77	**29.17**↑8.19
Triples Retriever (Base)	21.51	24.55	29.86	18.54	10.79	21.05
Triples Retriever (Ours)	**24.96**↑3.45	**41.23**↑16.68	**36.25**↑6.39	**23.88**↑5.34	**13.89**↑3.10	**28.04**↑6.99
ToG (Base)	20.77	18.17	25.63	15.29	7.79	17.53
ToG (Ours)	19.74	**34.20**↑16.03	**30.70**↑5.07	**21.35**↑6.06	**9.60**↑1.81	**23.12**↑5.59
Llama-3.2-3B						
Subgraph (Base)	25.19	44.07	34.86	26.23	12.26	28.52
Subgraph (Ours)	**27.34**↑2.15	**53.50**↑9.43	**40.89**↑6.03	**33.91**↑7.68	**15.50**↑3.24	**34.23**↑5.71
Triples Retriever (Base)	26.18	43.80	39.11	27.03	15.21	30.27
Triples Retriever (Ours)	**30.93**↑4.75	**51.50**↑7.70	**43.67**↑4.56	**31.49**↑4.46	**18.21**↑3.00	**35.16**↑4.89
ToG (Base)	20.00	31.33	31.93	27.20	11.14	24.32
ToG (Ours)	**23.78**↑3.78	**48.55**↑17.22	**40.60**↑8.67	**39.79**↑12.59	**17.31**↑6.17	**34.01**↑9.69

RL-Optimized Graphs Also Improve Text Retrieval

When KG acts as a knowledge index (Knowledge-Indexing Reward):

Methods	Simple QA		Multihop QA			Overall
	NQ*	PopQA*	HotpotQA	2WikiQA	Musique	Avg.
Qwen2.5-3B						
HippoRAG (Base)	36.28	65.55	53.22	48.97	27.44	46.29
HippoRAG (Ours)	**38.28**↑2.00	**65.93**↑0.38	**55.39**↑2.17	**51.69**↑2.72	**28.11**↑0.67	**47.88**↑1.59
HippoRAG2 (Base)	35.88	65.02	53.70	50.98	25.70	46.25
HippoRAG2 (Ours)	**38.45**↑2.57	**66.23**↑1.21	**56.28**↑2.58	**52.80**↑1.82	**27.93**↑2.23	**48.34**↑2.09
Qwen2.5-7B						
HippoRAG (Base)	37.16	65.95	55.50	53.01	26.03	47.53
HippoRAG (Ours)	**38.80**↑1.64	**67.85**↑1.90	**57.19**↑1.69	**53.60**↑0.59	**26.97**↑0.94	**48.88**↑1.35
HippoRAG2 (Base)	37.02	65.74	57.08	54.99	26.77	48.32
HippoRAG2 (Ours)	**38.68**↑1.66	**67.72**↑1.97	**58.98**↑1.90	**56.46**↑1.47	**27.18**↑0.41	**49.80**↑1.48
Llama-3.2-1B						
HippoRAG (Base)	28.94	39.16	39.37	27.26	18.89	30.72
HippoRAG (Ours)	**35.49**↑6.55	**52.63**↑13.47	**47.31**↑7.94	**38.13**↑10.87	**20.17**↑1.28	**38.75**↑8.03
HippoRAG2 (Base)	29.79	40.64	37.90	29.82	20.21	31.67
HippoRAG2 (Ours)	**35.61**↑5.82	**51.66**↑11.02	**50.97**↑13.07	**41.24**↑11.42	**21.58**↑1.37	**40.21**↑8.54
Llama-3.2-3B						
HippoRAG (Base)	38.35	58.81	50.68	44.26	23.63	43.15
HippoRAG (Ours)	**38.77**↑0.42	**59.62**↑0.81	**54.82**↑4.14	**47.03**↑2.77	**25.26**↑1.63	**45.10**↑1.95
HippoRAG2 (Base)	38.20	60.18	52.86	46.44	23.78	44.29
HippoRAG2 (Ours)	**38.54**↑0.34	**61.05**↑0.87	**54.43**↑1.57	**48.79**↑2.35	23.44	**45.25**↑0.96

Passage recall@5 improves by +15 points for Llama-1B with HippoRAG. The RL framework creates more effective knowledge indices.

Optimizing for Utility Also Improves Factual Quality

A natural concern: Does optimizing for downstream utility sacrifice intrinsic factual accuracy? The answer is no — it improves both simultaneously.

KG Construction Model	HotpotQA			2WikiMultihopQA			Musique			2021Wiki		
	Acc	Recall	F1	Acc	Recall	F1	Acc	Recall	F1	Acc	Recall	F1
Qwen2.5-7B-Instruct	98.50	93.68	95.65	94.80	91.19	92.68	96.77	95.27	95.73	95.03	91.39	92.92
+ GRPO with Knowledge-Carrying Reward	97.53	**96.66**	**96.71**	95.51	**96.55**	95.25	97.14	**96.75**	**96.45**	96.17	**96.66**	**96.15**
+ GRPO with Knowledge-Indexing Reward	**98.96**	94.81	96.59	**98.35**	94.54	**96.16**	**99.53**	93.14	95.81	**97.44**	95.01	95.99
Qwen2.5-3B-Instruct	94.41	91.00	91.92	83.53	79.34	81.01	92.07	89.52	90.31	87.79	86.01	86.63
+ GRPO with Knowledge-Carrying Reward	96.52	**94.24**	94.80	95.91	**96.28**	**95.80**	**97.01**	**94.74**	**95.22**	96.70	95.58	95.76
+ GRPO with Knowledge-Indexing Reward	**97.11**	93.15	**94.64**	**96.19**	93.66	94.48	96.20	93.87	94.55	**98.22**	**96.04**	**96.85**

RL-trained models produce graphs with higher precision, recall, AND F1 compared to zero-shot baselines. Task-aware optimization aligns with and enhances factual quality.

Lessons from Task-Aware KG Construction

Three key takeaways:

1. **Close the loop:** Directly optimizing KG construction for downstream performance yields significant gains — the disconnect between construction and application can be bridged.

2. **Reward design matters:** Different graph roles (carrier vs. index) require different optimization objectives — one size does not fit all.

3. **Quality and utility are aligned:** Optimizing for extrinsic utility simultaneously improves intrinsic factual quality — there is no trade-off.

> **This represents a paradigm shift: from building "good" graphs to building "useful" ones.**

Connecting the Dots

The three works form a clear research arc addressing increasingly fundamental questions:

Dimension	Intention KG (RIG)	AutoSchemaKG (ATLAS)	AutoGraph-R1
Core Question	What to represent?	How to organize?	How to optimize?
Schema	Predefined (6 types)	Autonomously induced	RL-optimized
Scope	E-commerce	Web-scale, multi-domain	Task-specific
Scale	351M edges	5.9B edges	Task-dependent
Evaluation Focus	Intrinsic + Domain	Intrinsic + General	Extrinsic utility
Key Innovation	Intention relations	Schema induction	RL for construction

Five Cross-Cutting Insights for KG Construction

1. **LLMs are powerful KG constructors** — All three works leverage LLMs as the primary extraction engine, from GPT-3.5 to LLaMA to Qwen.
2. **Beyond entities** — Intentions (Paper 1), events (Paper 2), and task-relevant structures (Paper 3) all demonstrate that entity-only KGs are insufficient.
3. **Conceptualization is universal** — Abstracting specific instances into broader categories is valuable across all three paradigms.
4. **Scale enables capability** — From 351M edges to 5.9B edges, larger KGs unlock new capabilities.
5. **Construction should serve application** — The progression from intrinsic to extrinsic evaluation reflects a maturing understanding of KG utility.

Where Do We Go from Here?

Several open questions emerge from this research line:

- **Multi-domain RL optimization** — Can AutoGraph-R1's approach be extended to optimize for multiple downstream tasks simultaneously?
- **Dynamic KG construction** — How do we update and maintain KGs as knowledge evolves?
- **Efficiency at scale** — AutoSchemaKG requires ~78,400 GPU hours. Can we reduce this while maintaining quality?
- **Cross-lingual construction** — All three works focus primarily on English. How do we extend to multilingual settings?
- **Integration** — Can we combine intention-aware modeling (Paper 1), autonomous schema induction (Paper 2), and RL optimization (Paper 3) into a unified framework?

KG Construction in the Age of LLMs

These three works collectively demonstrate that knowledge graphs remain essential even in the era of powerful LLMs.

Rather than being replaced by LLMs, KGs are being reinvented through them. The key insight is that structured knowledge representations offer advantages that parametric knowledge are not very good at:

- Explainability
- Updatability
- Compositional reasoning

The future of KG construction lies at the intersection of autonomous extraction, dynamic schema induction, and task-aware optimization.

> **As LLMs continue to improve, the quality and scale of automatically constructed KGs will grow in tandem, creating a virtuous cycle between structured and parametric knowledge.**

More about my research:

Thank You

Jiaxin Bai
Assistant Professor
https://bjx.fun