NEURAL INFORMATION PROCESSING SYSTEMS

2024 Conference on Neural Information Processing Systems

UrbanKGent: A Unified Large Language Model Agent Framework for Urban Knowledge Graph Construction

Yansong NING[1], Hao LIU[1,2]

[1]AI Thrust, The Hong Kong University of Science and Technology (Guangzhou)
[2]CSE, he Hong Kong University of Science and Technology

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

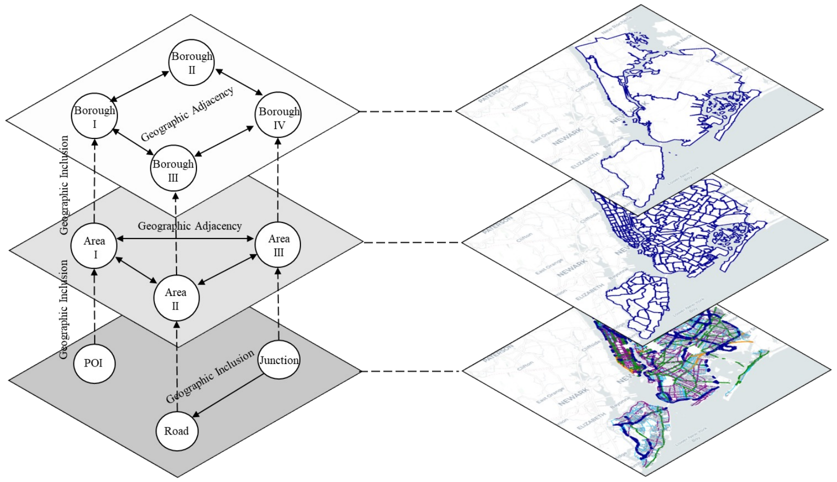
OUTLINE

- **Introduction**
- Our Proposed Method – UrbanKGent
- Experiments
- Conclusion and Future Work

Background

☐ Urban Knowledge Graph (UrbanKG)

- ■ Organize urban entities into a multi-relational heterogeneous graph to model intricate relationships and semantics.
- ■ UrbanKG provides critical knowledge for various knowledge-enhanced urban downstream tasks.



An illustrative UrbanKG example

Urban downstream tasks

Research Gap

☐ Previous UrbanKG Construction (UrbanKGC) Studies

- ■ Manually designed methods: 1) rely on deep understanding of the application domain; 2)labor-intensive.
- ■ Language model based methods: 1) rely on annotated corpus; 2) need model retraining.

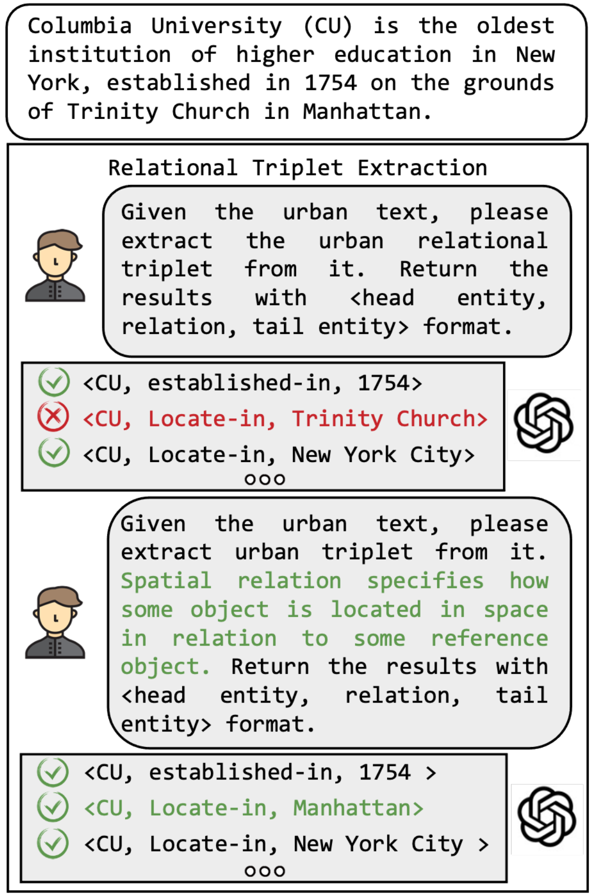
☐ Motivation

- ■ Leverage the remarkable zero-shot capability of LLM in autonomous domain-specific task completion.
- ■ Construct tailored LLM agent compatible with various UrbanKGC tasks to address the aforementioned limitations in UrbanKGC.

Challenges

☐ How to adapt LLMs for UrbanKGC?

- The gap between natural language processing corpus for training LLMs and the domain-specific corpus in urban domain.
- Urban text data is usually heterogeneous and contains multifaceted urban knowledge (e.g., spatial, temporal, and functional aspects).
- LLMs may require a tailored alignment to understand heterogeneous urban relationships to extract these urban relations accurately.

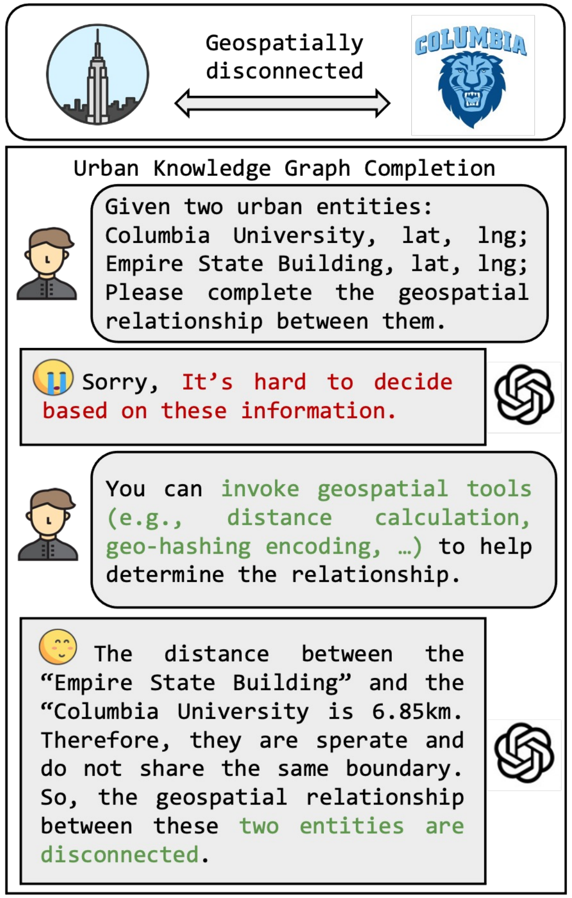


(a) Lack of Heterogeneous relationship understanding ability

Challenges

☐ How to improve the capacity of LLMs for UrbanKGC?

- ■ The effectiveness of LLMs for UrbanKGC is restricted by their feeble numerical computation capacity.
- ■ Lead to disability in complex geospatial relationship extraction.
- ■ To improve the geospatial computing and reasoning ability (e.g., invoking external tools for calculation) of LLMs to satisfy the UrbanKGC task requirement.



(b) Lack of Geospatial Computing Ability

OUTLINE

Introduction

Our Proposed Method – UrbanKGent

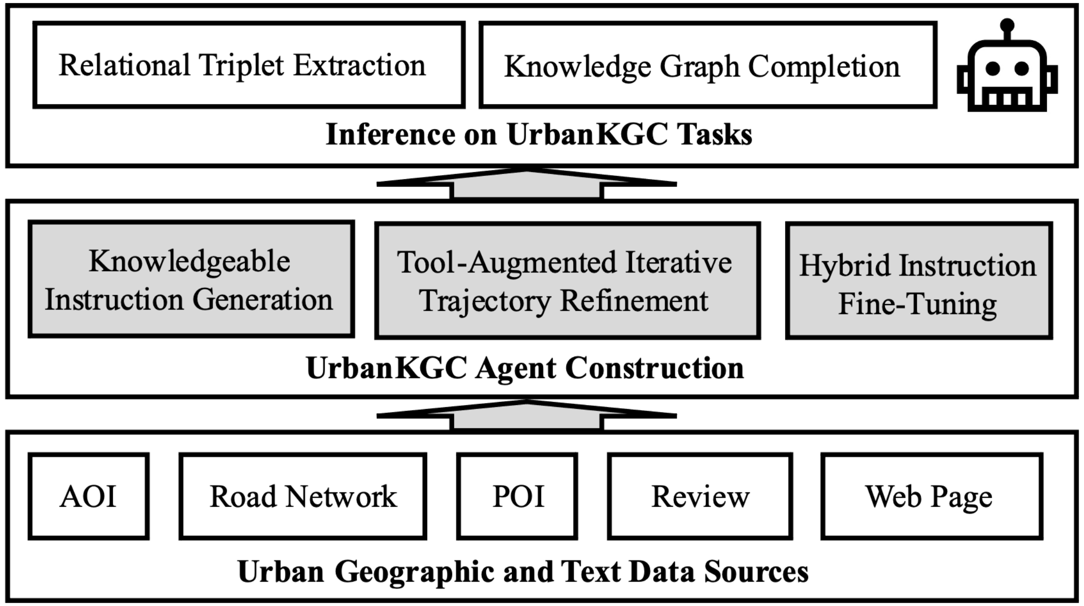
Experiments

Conclusion and Future Work

The Proposed Method: UrbanKGent

☐ Overview

- A unified LLM agent framework for automatic UrbanKG construction.
- Three steps: 1) urban data collection; 2) UrbanKGC agent construction; 3) Inference on UrbanKGC task.



The Proposed Method: UrbanKGent

☐ Data Collection

- Acquire geographic data and text data for two large cities New York City and Chicago.
- Geographic and text data: Area-Of-Interest (AOI), Road network, Point-Of-Interest (POI), Review and Web page.

Table 1: The statistics of raw datasets.

Dataset	Description	New York City	Chicago
Geographic Data	# of AOI	192	136
	# of road	6,765	2,241
	# of POI	5,872	5,877
Text Data	# of review	16,360	13,627
	# of web page	11,596	7,283

The Proposed Method: UrbanKGent

□ Quantitative Analysis on UrbanKGC task

- UrbanKGC tasks: 1) Relational Triplet Extraction (RTE; 2)Knowledge Graph Completion (KGC).
- Limited capacity of LLM to understand heterogenous urban relationship.
- Disability of LLMs in geospatial computing and reasoning.

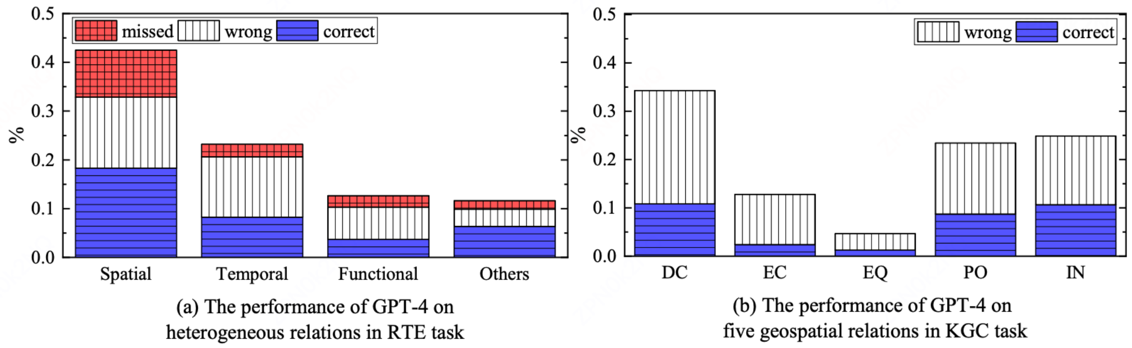
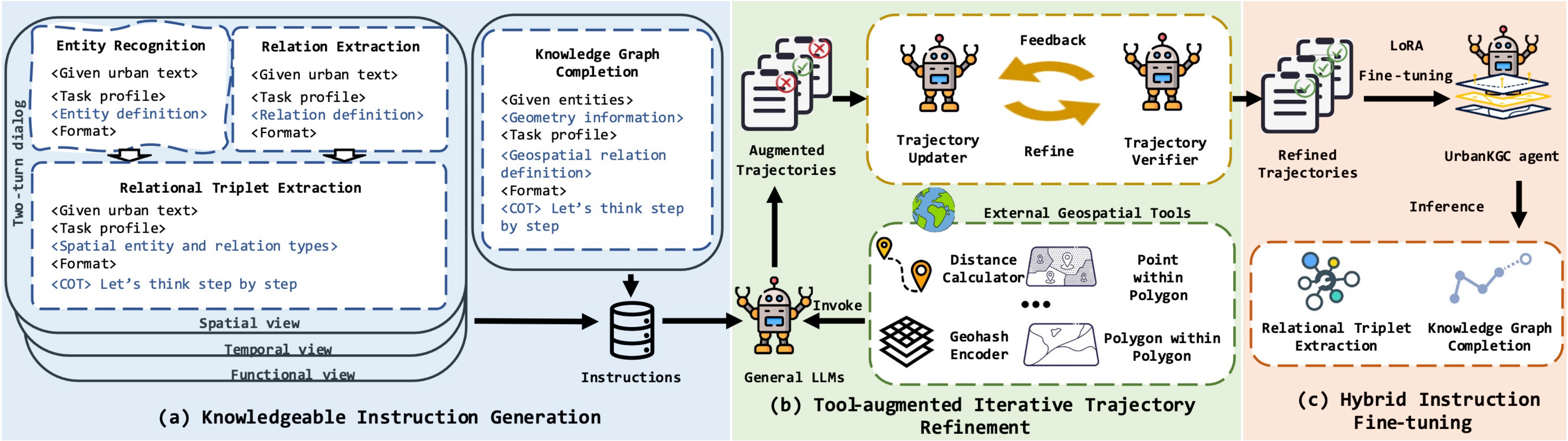


Figure 3: Quantitative performance analysis of prompting GPT-4 for UrbanKGC tasks. The result is obtained by comparing 50 GPT-4's outputs with the human's annotation.

The Proposed Method: UrbanKGent

☐ UrbanKGC Agent Construction

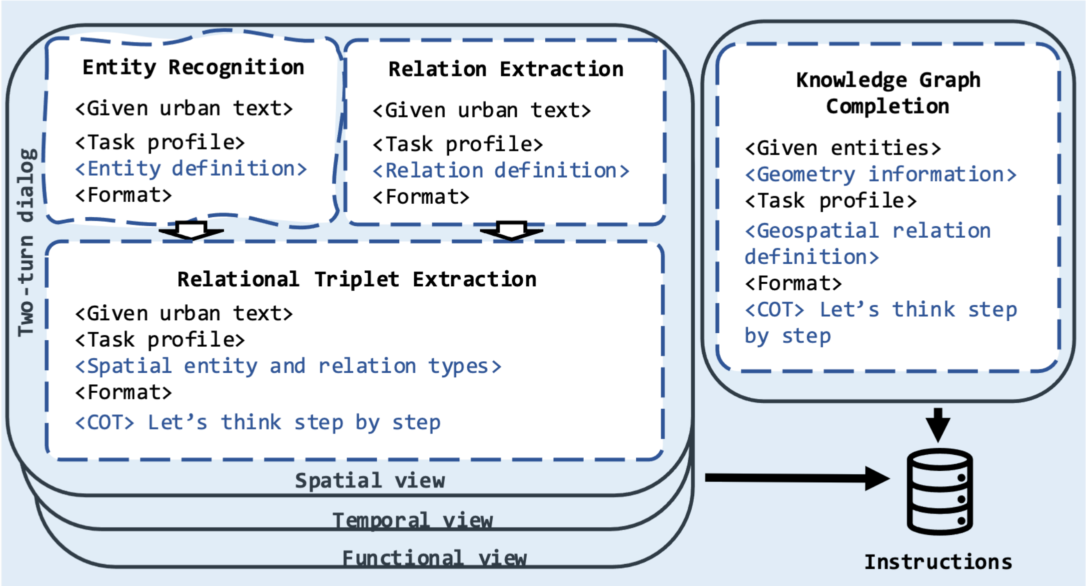
- Knowledgeable instruction generation for aligning LLM to UrbanKGC tasks;
- Tool-augmented iterative trajectory refinement to enhance and refine generated trajectory;
- Hybrid Instruction Fine-tuning for cost-effectively completing UrbanKGC tasks.



The Proposed Method: UrbanKGent

☐ Knowledgeable Instruction Generation

- ■ Heterogeneity-aware instruction generation for relational triplet extraction.
    - Sequential entity recognition, relation extraction and triplet extraction.
- ■ Geospatial-infused instruction generation for knowledge graph completion.

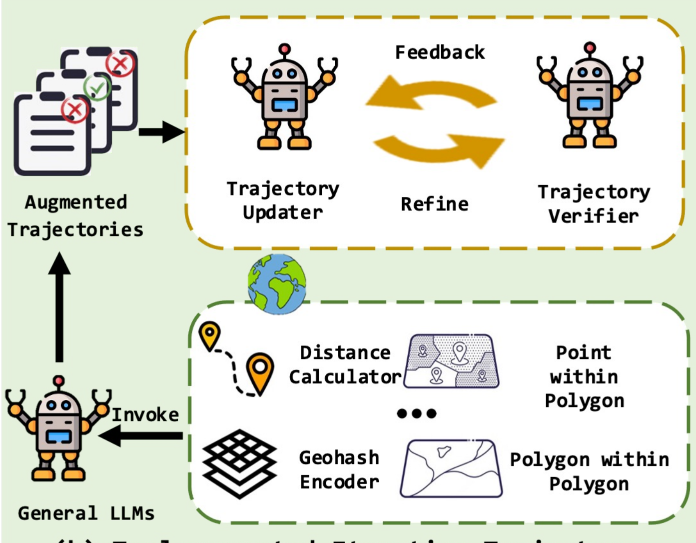


(a) Knowledgeable Instruction Generation

The Proposed Method: UrbanKGent

☐ Tool-augmented Iterative Trajectory Refinement

- Trajectory generation: distill from GPT-4.
- Geospatial tool invocation for trajectory augmentation.
  - Distance calculator, geohash encoder, …
- Iterative trajectory self-refinement to ensure trajectory quality.
  - Trajectory updater and verifier

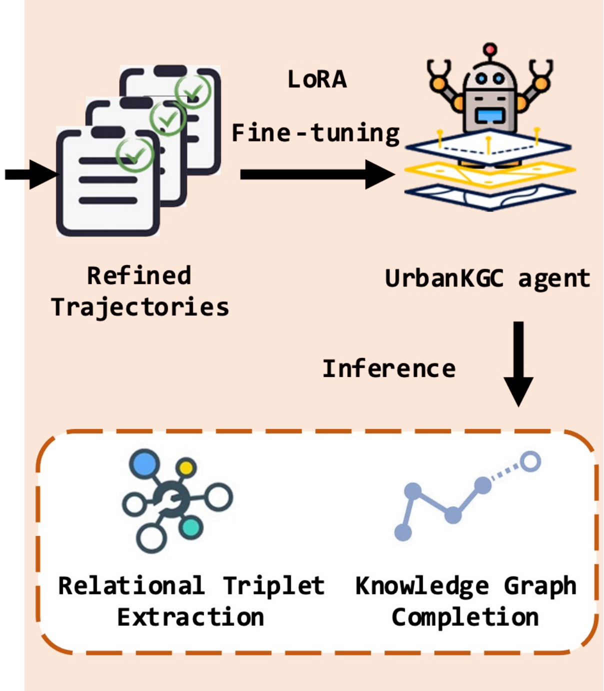


(b) Tool-augmented Iterative Trajectory Refinement

The Proposed Method: UrbanKGent

☐ Hybrid Instruction Fine-Tuning

- Utilize distilled trajectories to fine-tune a smaller open-source LLM for faster inference speed and lower cost.
- Mixture training on two UrbanKGC tasks (i.e., relational triplet extraction and knowledge graph completion).



(c) Hybrid Instruction Fine-tuning

OUTLINE

Introduction

Our Proposed Method – UrbanKGent

Experiments

Conclusion and Future Work

Experiments

◻ Dataset, Baselines and Metric

- Construct the RTE and KGC datasets for fine-tuning and validation;
- Five types of paradigms: 1) Pretrained language model methods; 2) LLMs-based zero-shot reasoning methods; 3) LLMs-based In-context learning methods; 4) Vanilla fine-tuning methods; 5) UrbanKGent Inference method;
- Evaluations: employ accuracy as metric on both of human evaluation and GPT evaluation.

Table 2: The statistics of constructed UrbanKGC dataset.

Dataset		NYC-Instruct	NYC	NYC-Large	CHI-Instruct	CHI	CHI-Large
Records	RTE	232	2,089	40,480	122	1,102	28,868
	KGC	232	2,080	33,534	122	1,101	28,607

Experiments

Main Results

- UrbanKGent-13B/8B/7B outperforms all thirty-one baseline models on two UrbanKGC datasets.
- UrbanKGent inference pipeline perform slightly worse than the vanilla fine-tuning method, but better thant zero-shot reasoning and In-context learning paradigms.

Type	Models	NYC GPT (acc/confidence) RTE	NYC GPT (acc/confidence) KGC	NYC Human (acc) RTE	NYC Human (acc) KGC	CHI GPT (acc/confidence) RTE	CHI GPT (acc/confidence) KGC	CHI Human (acc) RTE	CHI Human (acc) KGC
End-to-end Models	KG-BERT	-	0.24/3.15	-	0.23	-	0.19/4.12	-	0.24
	KG-T5	-	0.21/4.02	-	0.21	-	0.15/3.98	-	0.24
	RelationPrompt	0.12/3.38	-	0.12	-	0.21/3.53	-	0.18	-
	PRGC	0.08/4.01	-	0.06	-	0.13/4.15	-	0.15	-
Zero-shot Reasoning	Vicuna-7B	0.12/2.84	0.19/4.06	0.14	0.16	0.22/4.12	0.14/4.03	0.21	0.18
	Alpaca-7B	0.11/3.75	0.17/3.87	0.15	0.17	0.23/3.96	0.16/4.15	0.20	0.16
	Mistral-7B	0.14/4.12	0.21/4.11	0.17	0.18	0.21/3.75	0.15/3.76	0.19	0.19
	Llama-2-7B	0.14/1.98	0.18/3.75	0.16	0.18	0.26/1.96	0.15/2.83	0.21	0.22
	Llama-3-8B	0.15/4.02	0.15/4.02	0.20	0.21	0.24/3.75	0.15/4.08	0.22	0.22
	Llama-2-13B	0.21/2.07	0.28/3.91	0.19	0.22	0.22/2.19	0.16/2.47	0.22	0.24
	Llama-2-70B	0.25/3.07	0.28/3.75	0.22	0.24	0.27/3.55	0.16/2.47	0.24	0.23
	Llama-3-70B	0.24/4.18	0.29/4.31	0.23	0.24	0.26/3.98	0.17/4.26	0.25	0.23
	GPT-3.5	0.29/4.11	0.36/3.47	0.31	0.23	0.31/3.79	0.31/3.16	0.31	0.29
	GPT-4	0.38/4.03	0.39/3.82	0.41	0.29	0.39/4.08	0.32/4.03	0.43	0.35
In-context Learning	Llama-2-7B	0.18/2.15	0.21/3.96	0.19	0.18	0.25/2.44	0.18/3.27	0.23	0.20
	Llama-3-8B	0.17/4.06	0.18/3.53	0.21	0.22	0.28/4.31	0.17/4.14	0.24	0.21
	Llama-2-13B	0.26/3.52	0.31/3.28	0.23	0.24	0.28/2.65	0.21/2.53	0.25	0.26
	GPT-3.5	0.41/4.65	0.42/4.08	0.42	0.31	0.36/4.24	0.36/4.23	0.39	0.36
Vanilla Fine-tuning	Llama-2-7B	0.32/4.37	0.38/3.65	0.32	0.27	0.29/3.80	0.30/3.65	0.33	0.31
	Llama-3-8B	0.31/4.18	0.35/4.18	0.35	0.26	0.31/4.18	0.29/4.15	0.32	0.34
	Llama-2-13B	0.35/4.26	0.41/3.92	0.39	0.29	0.31/4.14	0.29/3.87	0.37	0.35
UrbanKGent Inference	Vicuna-7B	0.24/3.07	0.24/3.95	0.29	0.23	0.27/4.12	0.22/3.95	0.23	0.25
	Alpaca-7B	0.26/3.85	0.27/3.83	0.26	0.22	0.27/3.83	0.21/4.12	0.27	0.29
	Mistral-7B	0.26/4.15	0.25/4.08	0.28	0.23	0.25/3.61	0.21/4.08	0.25	0.26
	Llama-2-7B	0.27/3.05	0.26/4.12	0.28	0.24	0.27/2.87	0.24/3.54	0.26	0.29
	Llama-3-8B	0.29/4.15	0.31/4.08	0.33	0.26	0.26/3.28	0.24/3.97	0.30	0.31
	Llama-2-13B	0.31/3.87	0.32/3.56	0.35	0.27	0.28/3.24	0.26/3.28	0.31	0.32
	Llama-2-70B	0.33/4.28	0.33/4.27	0.33	0.29	0.29/3.80	0.28/4.01	0.32	0.34
	Llama-3-70B	0.35/4.26	0.36/4.81	0.34	0.28	0.29/4.12	0.29/4.81	0.31	0.35
	GPT-3.5	0.43/4.12	0.46/3.88	0.43	0.34	0.40/4.21	0.39/3.87	0.46	0.41
	GPT-4	0.45/4.08	0.48/4.02	0.47	0.42	0.46/4.17	0.41/4.35	0.52	0.43
UrbanKGent-7B		0.46/4.12 ↑ 2.22%	0.49/3.97 ↑ 2.08%	0.48 ↑ 2.08%	0.44 ↑ 4.76%	0.49/4.28 ↑ 6.52%	0.43/4.58 ↑ 4.88%	0.54 ↑ 3.84%	0.45 ↑ 4.66%
UrbanKGent-8B		0.47/3.97 ↑ 4.44%	0.51/4.15 ↑ 6.25%	0.49 ↑ 4.26%	0.45 ↑ 7.14%	0.49/3.97 ↑ 6.52%	0.44/4.05 ↑ 7.32%	0.55 ↑ 5.77%	0.46 ↑ 6.98%
UrbanKGent-13B		0.52/4.38 ↑ 15.56%	0.56/4.13 ↑ 14.29%	0.54 ↑ 14.89%	0.47 ↑ 11.90%	0.53/4.15 ↑ 15.22%	0.48/4.42 ↑ 17.07%	0.59 ↑ 13.46%	0.49 ↑ 13.95%

Experiments

Main Results

- Fine-tuning LLMs could obtain better performance compared with zero-shot reasoning and In-context learning paradigms.
- In-context-learning is limited and even leads to degradation in smaller LLMs (e.g., Llama-2-7B).

	Vicuna-7B	0.12/2.84	0.19/4.06	0.14	0.16	0.22/4.12	0.14/4.03	0.21	0.18
	Alpaca-7B	0.11/3.75	0.17/3.87	0.15	0.17	0.23/3.96	0.16/4.15	0.20	0.16
	Mistral-7B	0.14/4.12	0.21/4.11	0.17	0.18	0.21/3.75	0.15/3.76	0.19	0.19
	Llama-2-7B	0.14/1.98	0.18/3.75	0.16	0.18	0.26/1.96	0.15/2.83	0.21	0.22
Zero-shot Reasoning	Llama-3-8B	0.15/4.02	0.15/4.02	0.20	0.21	0.24/3.75	0.15/4.08	0.22	0.22
	Llama-2-13B	0.21/2.07	0.28/3.91	0.19	0.22	0.22/2.19	0.16/2.47	0.22	0.24
	Llama-2-70B	0.25/3.07	0.28/3.75	0.22	0.24	0.27/3.55	0.16/2.47	0.24	0.23
	Llama-3-70B	0.24/4.18	0.29/4.31	0.23	0.24	0.26/3.98	0.17/4.26	0.25	0.23
	GPT-3.5	0.29/4.11	0.36/3.47	0.31	0.23	0.31/3.79	0.31/3.16	0.31	0.29
	GPT-4	0.38/4.03	0.39/3.82	0.41	0.29	0.39/4.08	0.32/4.03	0.43	0.35
	Llama-2-7B	0.18/2.15	0.21/3.96	0.19	0.18	0.25/2.44	0.18/3.27	0.23	0.20
In-context Learning	Llama-3-8B	0.17/4.06	0.18/3.53	0.21	0.22	0.28/4.31	0.17/4.14	0.24	0.21
	Llama-2-13B	0.26/3.52	0.31/3.28	0.23	0.24	0.28/2.65	0.21/2.53	0.25	0.26
	GPT-3.5	0.41/4.65	0.42/4.08	0.42	0.31	0.36/4.24	0.36/4.23	0.39	0.36
	Llama-2-7B	0.32/4.37	0.38/3.65	0.32	0.27	0.29/3.80	0.30/3.65	0.33	0.31
Vanilla Fine-tuning	Llama-3-8B	0.31/4.18	0.35/4.18	0.35	0.26	0.31/4.18	0.29/4.15	0.32	0.34
	Llama-2-13B	0.35/4.26	0.41/3.92	0.39	0.29	0.31/4.14	0.29/3.87	0.37	0.35

Experiments

☐ Efficiency and Complexity Analysis

- Lower inference speed in latency and reduce the cost by roughly 20 times.
- Compared with ZSL, ICL , VFT, and UrbanKGent Inference, UrbanKGent can incorporate extra urban knowledge, invoke external tools and iteratively self-refine to help better complete UrbanKGC tasks.

Table 11: Comparison among LLM-based UrbanKGC methods in four ways.

Method	Extra Knowledge	Require Fine-tuning	Tool Invokation	Self Refinement
ZSL	×	×	×	×
ICL	√	×	×	×
VFT	√	√	×	×
UrbanKGent Inference	√	×	√	√
UrbanKGent	√	√	√	√

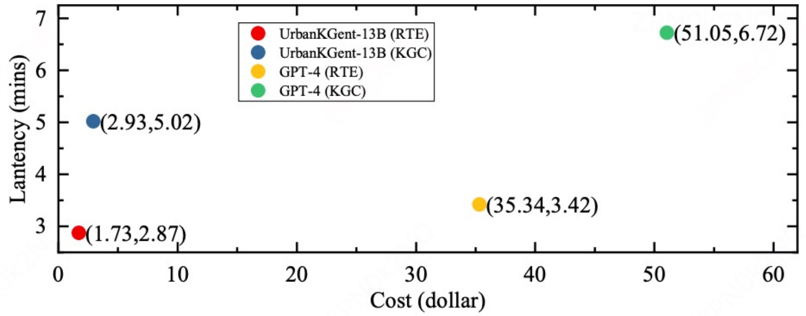


Figure 5: The model latency and cost of constructed UrbanKGent-13B and GPT-4 in UrbanKGC. We report the total inference time and cost of 1,000 RTE and KGC tasks.

Experiments

Agent Application

- Use 1/5 data for constructing the UrbanKGs with the same scale, and even expanding the variety of relationships to a thousand times the original types.
- UrbanKGent agent family consists of 13B/8B/7B is released in Hugging Face.

Table 4: Statistics comparison of constructed UrbanKGs in New York and Chicago between UrbanKGent and existing benchmark.

Dataset	# Entity	# Relation	# Triplet	Data Volume
NYC-Large	228,928	2,138	905,442	40,480
CHI-Large	95,813	1,336	563,290	28,607
NYC-UUKG	236,287	13	930,240	236,277
CHI-UUKG	140,602	13	564,400	140,577

UrbanKGent

The UrbanKGC agent family consisting of UrbanKGent-7/8/13B version.

usail-hkust/UrbanKGent-7B
Updated 14 days ago • 6

usail-hkust/UrbanKGent-8B
Updated 14 days ago • 5

usail-hkust/UrbanKGent-13B
Updated 14 days ago • 12

OUTLINE

Introduction

Our Proposed Method – UrbanKGent

Experiments

Conclusion and Future Work

Conclusion and Future Work

Conclusion

- Propose the first automatic UrbanKG construction agent framework.
- Release the UrbanKGent agent family, with lower latency and cost compared with GPT-4 for UrbanKG construction.

Limitation

- Lack of further application demonstration for constructed UrbanKG, the GPT-based self-evaluation methods is cost-intensive and inconsistent.

Future Work

- Derive extra image-modality data to further enrich UrbanKGC.

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Our Github and Hugging Face repository are continuously updating!

https://github.com/usail-hkust/UrbanKGent

https://huggingface.co/usail-hkust

Thank You!

Q & A

Contact us if you have further questions.

yning092@connect.hkust-gz.edu.cn