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, The Hong Kong University of Science and Technology
yning092@connect.hkust-gz.edu.cn liuh@ust.hk

Abstract

Urban knowledge graph has recently worked as an emerging building block to distill critical knowledge from multi-sourced urban data for diverse urban application scenarios. Despite its promising benefits, urban knowledge graph construction (UrbanKGC) still heavily relies on manual effort, hindering its potential advancement. This paper presents **UrbanKGent**, a unified large language model agent framework, for urban knowledge graph construction. Specifically, we first construct the knowledgeable instruction set for UrbanKGC tasks (such as relational triplet extraction and knowledge graph completion) via heterogeneity-aware and geospatial-infused instruction generation. Moreover, we propose a tool-augmented iterative trajectory refinement module to enhance and refine the trajectories distilled from GPT-4. Through hybrid instruction fine-tuning with augmented trajectories on Llama 2 and Llama 3 family, we obtain UrbanKGC agent family[2], consisting of UrbanKGent-7/8/13B version. We perform a comprehensive evaluation on two real-world datasets using both human and GPT-4 self-evaluation. The experimental results demonstrate that UrbanKGent family can not only significantly outperform 31 baselines in UrbanKGC tasks, but also surpass the state-of-the-art LLM, GPT-4, by more than 10% with approximately 20 times lower cost. Compared with the existing benchmark, the UrbanKGent family could help construct an UrbanKG with hundreds of times richer relationships using only one-fifth of the data. Our data and code are available at https://github.com/usail-hkust/UrbanKGent.

1 Introduction

Urban Knowledge Graph (UrbanKG) aims to model intricate relationships and semantics within a city by extracting and organizing urban entities (e.g., POIs, road networks, etc.) into a multi-relational heterogeneous graph [1]. As an emerging building block, multi-sourced urban data are widely used to construct an UrbanKG to provide critical knowledge for various knowledge-enhanced urban downstream tasks, such as traffic management, pollution monitoring, and emergency response [2, 3, 4, 5]. UrbanKG has gradually become an essential tool of the modern smart city.

In prior literature, many efforts have been devoted to urban knowledge graph construction (UrbanKGC) using massive urban data sources. In particular, one commonly used approach [6, 7, 8] is to extract entities from structured urban data (e.g., geographic data, city sensor data, and traffic data) and define the relationships between obtained urban entities based on manually designed rules or patterns. However, these approaches suffer heavy reliance on a deep understanding of the application domain and are labor-intensive. Recently, inspired by the success of the Large Language Models (LLMs)

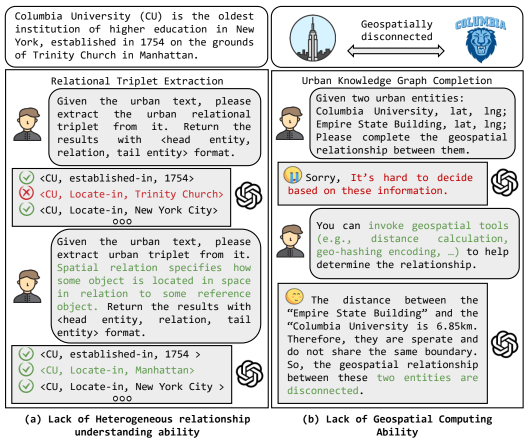
*Corresponding author.

[2]https://huggingface.co/usail-hkust/UrbanKGent-7B, https://huggingface.co/usail-hkust/UrbanKGent-8B and https://huggingface.co/usail-hkust/UrbanKGent-13B

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

in knowledge graph construction [9, 10, 11], the LLMs have been adopted to improve UrbanKGC. For instance, GeoLM [12] is pretrained on crowdsourced geographical corpus for geospatial entity recognition and relation extraction. K2 [13] retrains Llama-2-7B model on manually processed and filtered geoscience text corpus for geospatial relation extraction. Nevertheless, these works rely on high quality **corpus annotation** and computational extensive **model retraining**, which may discourage researchers from adopting UrbanKG for their own work.

LLM agent [14, 15] has recently emerged and shown remarkable zero-shot capability for autonomous domain-specific task completion. For example, Voyager [16] is a LLM-powered agent for zero-shot game exploration without re-training, and LLMLight [17] is a traffic signal control agent with zero-shot LLM reasoning ability. These studies motivate us to construct tailored LLM agents to address the aforementioned limitations in UrbanKG construction.


Columbia University (CU) is the oldest institution of higher education in New York, established in 1754 on the grounds of Trinity Church in Manhattan.

Relational Triplet Extraction

Given the urban text, please extract the urban relational triplet from it. Return the results with <head entity, relation, tail entity> format.

✓ <CU, established-in, 1754>
✗ <CU, Locate-in, Trinity Church>
✓ <CU, Locate-in, New York City>
ooo

Given the urban text, please extract urban triplet from it. Spatial relation specifies how some object is located in space in relation to some reference object. Return the results with <head entity, relation, tail entity> format.

✓ <CU, established-in, 1754 >
✓ <CU, Locate-in, Manhattan>
✓ <CU, Locate-in, New York City >
ooo

Urban Knowledge Graph Completion

Given two urban entities: Columbia University, lat, lng; Empire State Building, lat, lng; Please complete the geospatial relationship between them.

Sorry, It’s hard to decide based on these information.

You can invoke geospatial tools (e.g., distance calculation, geo-hashing encoding, …) to help determine the relationship.

The distance between the “Empire State Building” and the “Columbia University is 6.85km. Therefore, they are sperate and do not share the same boundary. So, the geospatial relationship between these two entities are disconnected.

Geospatially disconnected
COLUMBIA
(a) Lack of Heterogeneous relationship understanding ability
(b) Lack of Geospatial Computing Ability

Figure 1: Illustrative example of urban relational triplet extraction and knowledge graph completion. (a) The heterogeneous relationship understanding limitation of LLMs can be addressed by injecting prior urban knowledge into instruction. (b) The geospatial computing limitation of LLMs can be alleviated by invoking external geospatial tools.

In fact, constructing an LLM agent compatible with various UrbanKGC tasks is a non-trivial problem due to the following two challenges: *(1) Challenges 1: How to adapt LLMs for UrbanKGC?* LLMs may not align well with the specific task due to the gap [18] between the natural language processing corpus for training LLMs and the domain-specific corpus in urban domain [19]. For example, the urban text data is usually heterogeneous and contains multifaceted urban knowledge (e.g., spatial, temporal, and functional aspects) [13]. As shown in Figure 1(a), the text description of *"Columbia University"* reflects its geographic spatial locations (i.e., spatial relationship), construction timelines (i.e., temporal relationship), and how it provides educational service for the city (i.e., functional relationship). LLMs may require a **tailored alignment to understand heterogeneous urban relationships** to extract these urban spatial, temporal, and functional relations accurately. *(2) Challenges 2: How to improve the capacity of LLMs for UrbanKGC?* The effectiveness of LLMs for urban knowledge graph construction is restricted by their feeble numerical computation capacity [20, 21], leading to their disability in complex geospatial relationship extraction [22, 23]. However, the urban geospatial relationship plays a vital role in urban semantic modeling [12] and has been widely incorporated in previous UrbanKGs [8, 24]. As can be seen in Figure 1, extracting *"disconnected"* relation between the geo-entity *"Columbia University"* and *"Empire State Building"* is useful for urban geo-semantic modeling. Accurately extracting such geospatial relationship demands necessary geospatial computing (e.g., utilizing latitude and longitude for distance calculation) and reasoning (i.e., deriving calculation results for geospatial relation reasoning) capabilities. It is appealing to improve the **geospatial computing and reasoning ability** of LLMs to satisfy the UrbanKGC task requirement.

To address the aforementioned challenges, in this study, we propose **UrbanKGent**, a unified LLM agent framework for automatic UrbanKG construction. Figure 2 illustrates the overview of UrbanKGent. For a given city, we first generate a knowledgeable instruction set for UrbanKGC tasks (relational triplet extraction and knowledge graph completion) from urban geographic and text data sources. By heterogeneity-aware and geospatial-infused instruction generation, as shown in Figure 1(a), various urban spatiotemporal relationship knowledge can be encoded into the instruction, which facilitates alignment between LLMs with the target UrbanKGC tasks. Moreover, we propose a tool-augmented iterative trajectory refinement module to enhance and refine the trajectory derived by distilling GPT-4 with the above constructed instructions. Based on geospatial tool augmentation and self-refinement, the deficiency of LLMs in geospatial computing and reasoning could be alleviated, and unfaithful trajectories could be filtered out. Finally, we perform hybrid instruction fine-tuning based on the enhanced and refined trajectories on Llama 2-7/13B and Llama 3-8B variants [25] by using LoRA [26]. The obtained UrbanKGent agent including 7/8/13B version, is feasible for completing multiple UrbanKGC tasks cost-effectively without extra GPT-API cost.

We conduct comprehensive experiments on two UrbanKGC tasks in two metropolises (New York City and Chicago) using both human evaluation and GPT-4-based self-evaluation. The empirical results validate the effectiveness of the proposed LLM agent for completing various UrbanKGC tasks. Moreover, the obtained UrbanKGent family (7/8/13B version) could help extract the same scale of triplets and entities of existing UrbanKG benchmark [8] using only one-fifth of data, and even expand the types of relations by hundreds of times.

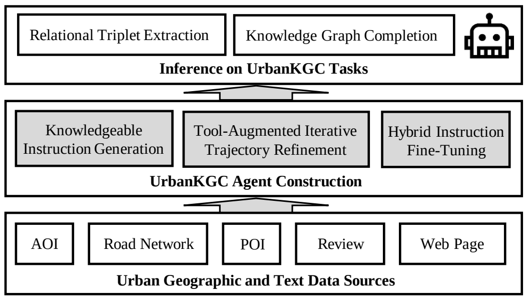

Relational Triplet Extraction
Knowledge Graph Completion
Inference on UrbanKGC Tasks
Knowledgeable Instruction Generation
Tool-Augmented Iterative Trajectory Refinement
Hybrid Instruction Fine-Tuning
UrbanKGC Agent Construction
AOI
Road Network
POI
Review
Web Page
Urban Geographic and Text Data Sources

Figure 2: The framework of UrbanKGent.

Our contributions are summarized as follows: (1) We propose the first UrbanKGC agent framework **UrbanKGent** and UrbanKGent family to provide real-world UrbanKGC service, offering new opportunities to advance UrbanKG studies. (2) We propose a knowledgeable instruction generation module and a tool-augmented iterative trajectory refinement method, which align LLMs to UrbanKGC tasks and compensate for their geospatial computing and reasoning inability. (3) Extensive experiments on two real-world datasets validate the effectiveness of proposed framework and uncover its exceptional performance across UrbanKGC tasks.

2 UrbanKGC Data Description

2.1 Data Collection

We first acquire urban knowledge for two large cities New York City and Chicago from two data sources. Table 1 summarizes the statistics of the raw datasets.

2.1.1 Geographic Data

The geographic data provides critical urban spatial structure information and functional semantics, which has been widely used in previous UrbanKG studies [8, 24, 27, 28].

Area-Of-Interst (AOI) Data. AOI data describes the urban spatial area structure, including urban commercial areas (e.g., shopping centers), residential areas (e.g., communities), and so on. In this work, we first follow UUKG [8] to acquire the AOI name and geometry value from NYC Gov [3] and CHI Gov [4]. Next, we use the AOI name to search their text description from Wikipedia and C4 [5] dataset. Each AOI record contains an AOI name, a polygon geometry value, and a text description. For example, *["Jamaica Bay", polygon (-73.86 40.58, ...), "Jamaica Bay is an estuary ..."]* is the record of the AOI *"Jamaica Bay"* with geometry value and text description.

Road Network Data. Road data describes the urban spatial network, including urban motorways, overpasses, and so on. We first follow [8] to obtain the road name and geometry value from Open Street Map (OSM [6]. Then, following the same text acquisition operation in AOI data, we crawl the textual description of each road record from Wikipedia. Each road record contains a road name, a linestring geometry value, a road type and a text description. For example, *["Central Park Avenue", linestring (-73.87 40.90, ...), primary, "Central Park Avenue is a boulevard in ..."]* describes the primary road named *"Central Park Avenue"* with a linestring geometry value and its textual description.

Point-Of-Interest (POI) Data. POI data represents different urban functions (e.g., residential and commercial), which have been widely adopted in many recent UrbanKG works [24, 6, 8]. We first follow [8] to obtain the POI name, and geometry value from OSM. Then the textual description of each POI record could be crawled from Wikipedia following the similar process. Each POI record contains a POI name, a coordinate geometry, a POI type, and a text description. For example,

[3]https://www.nyc.gov/
[4]https://www.chicago.gov/
[5]https://huggingface.co/datasets/allenai/c4
[6](https://www.openstreetmap.org/

["Trump World Tower", coordinate (-73.96 40.75), residential, "Trump World Tower is a residential condominium ..."] is the record of the POI *"Trump World Tower"*.

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

2.1.2 Text Data

The text data provides rich contextual knowledge of the city space from different perspectives (e.g., the spatial context) [13], and it plays an important role in geospatial understanding. In this work, we collect two types of text corpus.

Review Data. The review of urban places provides commercial information that citizens use to make business decisions [29], playing a critical role in urban knowledge distillation. We collect review data from Google Map [7]. Specifically, we first manually split the city into multiple rectangular regions, then we utilize the Google Map API to query the places contained within each region and their reviews. Each review record contains a place name, a coordinate geometry value, a rating, and a text review. For example, *["Lifestyles Academy Inc", coordinate (-87.87 41.65), 4.9, "Very nice organization and ..."]* is the review record of place *"Lifestyles Academy Inc"*.

Web Page Data. The web page data works as the general text corpus for the city, and it contains rich geoscience knowledge that has been utilized in recent urban entity and relation extraction studies [13]. We collect web page data from the Google search engine. Specifically, we first input the name of the crawled AOI, Road, and POI record into Google. Then we concatenate the textual sentences of the top 10 retrieved web pages. Each web page record contains a long urban text description.

2.2 Data Preprocessing

Before constructing the UrbanKGC dataset, we first preprocess the raw datasets. We filter out AOIs, roads, POIs, reviews, and web pages whose crawled textual descriptions are null value, too short (e.g., less than ten word description) or meaningless (e.g., just repeating the POI name). In addition, we remove irrelevant information from the text description, such as non-English characters, non-ASCII gibberish, website addresses, and so on. More details can be found in Appendix A.

3 Preliminary

This section presents the UrbanKGC task definition and provides task analysis.

3.1 Task Definition and Problem Formulation

Before diving into the technical details, we first introduce the definition of UrbanKG:

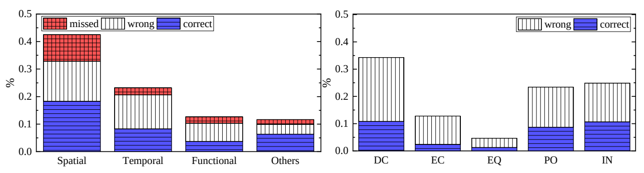
Definition 1 ***UrbanKG***. *The UrbanKG is defined as a multi-relational graph $\mathcal{G} = (\mathcal{E}, \mathcal{R}, \mathcal{F})$, where $\mathcal{E}$, $\mathcal{R}$ and $\mathcal{F}$ is the set of urban entities, relations and facts, respectively. In particular, facts are defined as $\mathcal{F} = \{\langle h, r, t\rangle \mid h, t \in \mathcal{E}, r \in \mathcal{R}\}$, where each triplet $\langle h, r, t\rangle$ describes head entity h is connected with tail entity t via relation r.*

The UrbanKG encodes diverse urban semantic knowledge by connecting urban entities into a multi-relational graph. This work aims to construct an UrbanKG from collected unstructured text data. We decompose the UrbanKG construction (UrbanKGC) process into two sequential knowledge graph construction tasks, namely relational triple extraction [10] and knowledge graph completion [11]. We first provide the basic definition for these two subtasks, and then introduce the problem formulation of this work.

3.1.1 Task Definition

Relational Triplet Extraction (RTE). Given the unstructured texts, this task achieves joint extraction of entities and their relations [10] which are in the form of a triplet $\langle h, r, t\rangle$. For instance, given the

[7]https://www.google.com/maps



(a) The performance of GPT-4 on heterogeneous relations in RTE task

(b) The performance of GPT-4 on five geospatial relations in KGC task

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.

urban text sentence *"Columbia University is a private Ivy league research university in New York City."*, this task aims to identify two entities *"Columbia University"* and *"New York City"* and their relation *"locate-in"*, described as triplet <*Columbia University, locate-in, New York City*>.

Knowledge Graph Completion (KGC). Given a head entity h and a tail entity t, this task is to predict the missing relation between them [11]. For instance, given the head entity *"Columbia University"* and the tail entity *"Empire State Building"*, this task is to predict that their missing relation, e.g., *"disconnected"*, described as triplet <*Columbia University, disconnected, Empire State Building*>.

3.1.2 Problem Formulation

Given the urban unstructured text data, the desired output is an UrbanKG $\mathcal{G}$. In this paper, this problem is decomposed into two sequential subtasks: (1) **Relational Triplet Extraction**: the first task extracts relational triplet $\langle h, r, t\rangle$ from the urban text data. The output of RTE task is $\mathcal{G}_1$ =($\mathcal{E}$, $\mathcal{R}_1$, $\mathcal{F}_1$), where $\mathcal{E}$ and $\mathcal{R}_1$is the set of extracted entities and relations, while $\mathcal{F}_1$ is the set of extracted triplets. (2) **Knowledge Graph Completion**: for the given head entity h and tail entity t in $\mathcal{G}_1$, the second task is to predict the geospatial relationship[8] between them. The output of this task is $\mathcal{G}_2$ =($\mathcal{E}$, $\mathcal{R}_2$, $\mathcal{F}_2$), where $\mathcal{R}_2$ and $\mathcal{F}_2$ is the set of completed relations and triplets. By sequentially completing the above two tasks, we can obtain the constructed UrbanKG $\mathcal{G}$ = ($\mathcal{E}$, $\mathcal{R}_1 \cup \mathcal{R}_2$, $\mathcal{F}_1 \cup \mathcal{F}_2$).

3.2 Quantitative Task Analysis

As shown in Figure 1, we qualitatively find that LLMs lack urban heterogeneous relationship understanding ability and experience in geospatial computing and reasoning difficulty when adopting it for UrbanKGC tasks. This subsection presents a quantitative analysis of these two challenges.

Heterogenous Relationship Understanding. The ability to understand heterogeneous relationships is ubiquitous in distilling knowledge from the massive urban corpus. For example, the text description in Figure 1 illustrates a place from spatial location, temporal time, and functional aspects. Capturing these heterogeneous semantics is important for urban knowledge distillation. We perform quantitative analysis by randomly sampling 50 urban text data and then prompt GPT-4 to complete relational triplet extraction by providing only the basic task description. As shown in Figure 3(a), we find the LLMs experience serious misjudgment (i.e., extract wrong triplets or miss the triplet) on urban spatial, temporal, and functional triplet extraction. This indicates the limited capacity of LLMs to understand heterogeneous relationships.

Geospatial Computing and Reasoning. Geospatial computing and reasoning techniques are widely used in many previous UrbanKG studies [8, 24] for urban geospatial relation extraction. In recent works [23, 31], the geospatial skills of LLMs have also been demonstrated to lack geospatial awareness and reasoning ability [22]. To identify potential limitations, we quantitatively investigate how LLMs can perform on geospatial relation completion tasks. Specifically, we construct 100 head and tail entity pairs, covering five geospatial relations in the KGC task, and then prompt GPT-4 to predict with basic task description and geospatial relation candidates. As shown in Figure 3(b), we find that GPT-4 performs poorly on five geospatial relation completion. This further validates the disability of LLMs in geospatial computing and reasoning.

4 UrbanKGC Agent Construction

This section presents the proposed UrbanKGC agent construction framework.

[8]We follow GeoLM [12] to provide five RCC relationship [30] candidates: Disconnection (DC), external connection (EC), equality (EQ), partial overlap (PO), and tangential and non-tangential proper parts (IN). Details are in Appendix A.

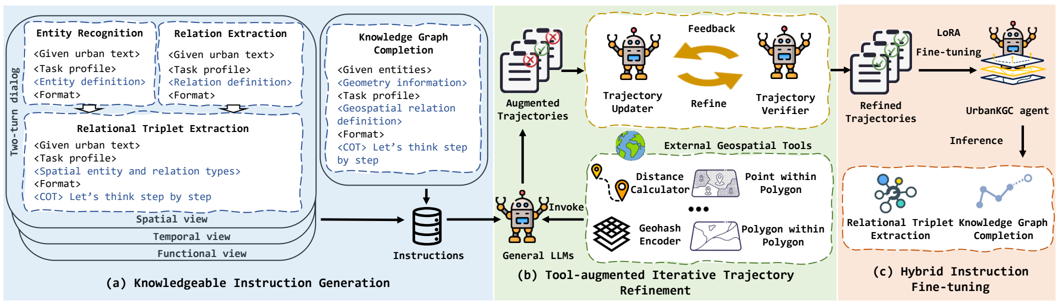


Figure 4: An overview of UrbanKGent Construction.

4.1 Overview

The overall pipeline of the UrbanKGent framework is illustrated in Figure 4. *(1) Knowledgeable Instruction Generation* consists of the heterogeneity-aware and geospatial-infused instruction generation modules for aligning LLMs to UrbanKGC tasks. *(2) Tool-augmented Iterative Trajectory Refinement* proposes geospatial tool interface invocation and iterative self-refinement mechanisms to enhance and refine generated trajectory. *(3) Hybrid Instruction Fine-tuning* fine-tune LLMs based on the refined trajectories for cost-effectively completing diverse UrbanKGC tasks.

4.2 Knowledgeable Instruction Generation

We first construct the knowledgeable instruction to adopt LLMs for two UrbanKGC tasks, including relational triplet extraction (RTE) and knowledge graph completion (KGC). Figure 4(a) illustrates the overview of the instruction construction process of these two tasks.

Heterogeneity-aware Instruction Generation for Relational Triplet Extraction. As discussed in Section 3, the urban text contains diverse heterogeneous relationships, thus we consider multiple views with both urban entity and relation definition for relational triplet extraction. In particular, we construct a multi-view instruction template for the urban relational triplet extraction, including spatial view, temporal view, and functional view. Each view is a multi-turn question-answer dialog [32] consisting of entity recognition, relation extraction, and triplet extraction module.

For the spatial view, we devise a two-turn dialog to align LLMs for spatial triplet extraction. In the first turn, we inject spatial entity and relation definition into the instruction template to guide LLMs to understand spatial characteristics and then extract potential spatial entities (e.g., *University*) and relations types (e.g., *locate-in*). In the second turn, the extracted types are explicitly fed into the instruction template for spatial triplet extraction. Intuitively, the spatial view allocates dedicated urban knowledge for LLMs to extract urban spatial relationships. Similarly, we construct the temporal view and functional view for corresponding temporal and functional triplet extraction, independently.

Geospatial-infused Instruction Generation for Knowledge Graph Completion. Despite heterogeneity-aware instruction enabling LLMs to extract urban triplets from various perspectives, the geospatial relationship between geospatial entities cannot be directly extracted. Therefore, we introduce a geospatial-infused instruction generation module to guide LLMs to complete missing geospatial relationships.

First, we incorporate the geometry information (i.e., the latitude and longitude) of geo-entities into instruction, so that the LLMs can utilize these geospatial values for relation inference. Second, we add the geospatial relationship definition to the instruction to guide LLMs in understanding the geospatial relationship definition. Intuitively, LLMs can refer to geospatial knowledge and make practical solutions for the KGC task. We provide the detailed instruction template in the Appendix B.

4.3 Tool-augmented Iterative Trajectory Refinement

4.3.1 Trajectory Generation

With the initial UrbanKGC instructions constructed, the following step is to generate reasoning trajectories [33], which will be used to fine-tune LLMs tailored to UrbanKGC task. Specifically, we follow FireAct [34] and use Chain-of-Thought (CoT) [35], a gradient-free technique, to prompt

GPT-4 (i.e., add prompt trigger *"Let's think step by step"* at the end of RTE and KGC instructions template) to generate the reasoning trajectories for UrbanKGC tasks.

The generated CoT trajectories could provide a step-by-step reasoning solution for UrbanKGC tasks. Nevertheless, the complex geospatial relationships cannot be easily extracted as discussed in Section 3 and recent geospatial reasoning works [31, 23, 22]. Therefore, we introduce a tool invocation module to guide LLMs to invoke tailored external geospatial tools [36] to enhance their geospatial computing and reasoning capacity for UrbanKGC tasks.

4.3.2 Tool Invocation for Trajectory Augmentation

We conduct two sequential procedures: tool invocation for geospatial computing support and trajectory deliberation for reasoning enhancement.

Tool Invocation. First, we construct a geospatial reasoning toolkit (e.g., distance calculation, eight interfaces in total shown in Table 6) by prompting GPT-4 for self-programming. Then, we construct tailored prompts to guide LLMs to invoke these interfaces. Specifically, the prompt is concatenated with an illustrative description of the function of each geospatial tool and a task instruction (i.e., *"Which types of tool interface you need"*). Intuitively, the external tool allocates calculation results for LLMs to infer missing geospatial relation. The toolkit description can be found in Appendix C.

Trajectory Deliberation. After manipulation with external tools, we prompt LLMs to refine uncertain reasoning steps based on these obtained manipulation results. Specifically, we construct the prompt by concatenating with manipulation results (e.g., the distance and geohash value of geo-entity) and a task instruction (i.e., *"Please refine your reasoning process"*). After feeding the prompt into GPT-4, the enhanced trajectory is obtained. Detailed prompt information can be found in Appendix C.

4.3.3 Iterative Trajectory Self-refinement

Despite tool-augmented deliberation improving the geospatial computing and reasoning ability of LLMs, enhanced trajectories may not all be faithful [37]. To alleviate potential error and ensure the trajectory quality [38], we refine these trajectories via an iterative self-refinement mechanism [39]. Specifically, we iterate two sequential blocks: (i) Trajectory verifier: given the trajectory, the verifier aims to provide feedback for refining the reasoning process; (ii) Trajectory updater: given the trajectory and feedback, the updater will further refine the current trajectory based on the feedback.

Trajectory Verifier. We construct a tailored prompt to ask LLMs to generate feedback. Specifically, we use a simple but effective trigger (*"Judge whether all extracted triplets are correct and provide improvement suggestion"*) to prompt LLMs to provide feedback. If the trajectory no longer requires modification, we let LLMs respond with *"This is a faithful trajectory"*. Such a verification step lets LLMs make reflections and improve the correctness of the trajectory.

Trajectory Updater. The updater utilizes provided feedback to refine the current trajectory via prompt trigger *"Follow suggestion to refine the reasoning process"*. Intuitively, the feedback may address multiple aspects (e.g., missed triplet in the RTE task or unfaithful reasoning process in the KGC task) of the unfaithful trajectories.

We iterate the trajectory verifier and updater until the predefined stopping condition is satisfied. The stopping condition is determined by either meeting the maximum number of iterations (we set it at three to avoid excessive cost) or when the verifier confirms all trajectories are faithful. Upon meeting the stopping condition, we use the last refined trajectory for further fine-tuning. Detailed prompt information can be found in Appendix C.

4.4 Hybrid Instruction Fine-Tuning

To construct a cost-effective UrbanKGC agent, we further utilize trajectories (generated by GPT-4) to fine-tune a smaller open-source LLM for faster inference speed and lower cost (i.e., prompting GPT-4 for UrbanKGC is expensive). Specifically, we finetune the LLM via the mixed-task instruction-tuning strategy [33]. The goal is to enhance the LLMs' capabilities in diverse UrbanKGC tasks.

Mixture Training. Set the base language model as $\mathcal{M}$, and $\mathcal{P}_{\mathcal{M}}(y \mid x)$ represents the probability distribution of response y given instruction x. We consider the trajectory set on two UrbanKGC tasks, i.e., $\mathcal{D}_{RTE}$ and $\mathcal{D}_{KGC}$. Since both the instruction and the target output are formatted in natural

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

language, we can unify the training into an end-to-end sequence-to-sequence way. Formally, the optimization process aims to minimize the loss of language model $\mathcal{M}$ as follows:

$$\mathcal{L} = \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{RTE}}}\left[\log \mathcal{P}_{\mathcal{M}}(y \mid x)\right] + \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{KGC}}}\left[\log \mathcal{P}_{\mathcal{M}}(y \mid x)\right], \qquad (1)$$

where x and y represent the instruction input and instruction output in the trajectory, respectively.

Training Setup. We choose the chat version of open-sourced Llama 2-7/13B and Llama-3-8B as our backbone models, and fine-tune Llama using LoRA strategy [26].

4.5 Inference on UrbanKGC Task

Via hybrid instruction fine-tuning, the obtained LLM UrbanKGent can be trained to follow the instructions to finish the UrbanKGC task. We prompt UrbanKGent to complete UrbanKGC tasks by following the pipeline shown in Figure 4. For the RTE task, we sequentially execute entity recognition, relation extraction, and relational triplet instruction generation, iterative self-refinement and output the extracted triplets. For the KGC task, we sequentially execute KGC instruction generation, external tool augmentation, iterative self-refinement block, and finally output the completed triplets.

5 Experiments

5.1 Experimental Settings

Dataset. In this work, two sequential tasks (i.e., RTE and KGC) of UrbanKGC are within an open-world setting (i.e., no predefined ontology) [40, 41]. We construct the RTE and KGC datasets of NYC and CHI by sampling uniformly from five raw data in Table 1, respectively. As shown in Table 2, we first construct two small datasets (i.e., NYC-Instruct and CHI-Instruct) for instruction fine-tuning and two middle datasets (i.e., NYC and CHI) to validate the performance of the constructed UrbanKGC agent. The remaining data works as the large-scale UrbanKGC dataset (i.e., NYC-Large and CHI-Large) in real-world scenarios shown in Table 5. The three types of datasets are non-overlapping to prevent data leakage. More dataset construction details are in Appendix A.

Baseline Methods. We provide a comprehensive comparison of our method with existing paradigms: **(1) End-to-end Models:** For the zero-shot RTE task, we utilize the end-to-end generation model RelationPrompt [42] and PRGC [10]. For the KGC task, we fine-tune KG-BERT [43] and KG-T5 [44] with the QA pairs constructed from the self-instruct dataset. **(2) LLMs-based Zero-shot Reasoning [45]:** We directly prompt the LLMs with basic task definitions to get the answer without training. **(3) LLMs-based In-context Learning [35]:** We sample 3-shot QA pairs as demonstrations from the self-instruct dataset as examples and get the answers from the LLMs without training. **(4) Vanilla Fine-tuning [11]:** We directly fine-tune the LLMs using the QA pairs constructed from the self-instruct dataset, and then prompt the LLMs with basic task definition without demonstrations. **(5) UrbanKGent Inference:** We directly prompt the LLMs using the UrbanKGgent inference pipeline in Section 4.5. The prompt templates of the above baseline methods are in Appendix B.

Implementation and Detail Settings. In our experiment, we select Vicuna [46], Alpaca [47], Mistrial [48], Llama-2 [49], Llama-3 [50], GPT-3.5 [51] and GPT-4 [51] as the backbone LLM $\mathcal{M}$. All experiments are conducted on eight NVIDIA A800 GPUs. For the GPT-3.5 and GPT-4, we adopt the gpt-3.5-turbo-16k-0613 API and gpt-4-0613 API.

Evaluation Protocol. Since UrbanKGC tasks in this work follow an open-world setting where labels are not visible, the classical metric (e.g., F1 and Hits@10) is not applicable. In this work, we regard evaluation as the binary classification, i.e., if the extracted triplet in RTE task is correct and if the completed relation in KGC task is correct. We follow recent LLMs-based KGC works [11] to employ accuracy as an evaluation metric. To make a comprehensive evaluation of the experimental results, we employ both of the human evaluation and GPT evaluation, which has been widely used in many LLM studies [52, 45]. For **Human Evaluation**, we employ human annotators to evaluate the results on

Table 3: The main results of relational triplet extraction (RTE) and knowledge graph completion (KGC). We report the accuracy (acc) and confidence for GPT evaluation on two datasets, and report accuracy (acc) for the Human evaluation approach. The best baseline performance is underlined.

Type	Models	NYC				CHI			
		GPT (acc/confidence)		Human (acc)		GPT (acc/confidence)		Human (acc)	
		RTE	KGC	RTE	KGC	RTE	KGC	RTE	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	0.35/4.27	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%

200 random samples. As for the **GPT Evaluation**, we use GPT-4 to evaluate the model performance on the full data to escape intensive labor. In this work, the GPT-4's evaluation has been demonstrated to be consistent with the human evaluation. Detail is in Appendix D.

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

5.2 Main Result

The performance results are reported in Table 3. As can be seen, the constructed agent outperforms all thirty-one baseline models on two UrbanKGC datasets. Specifically, the UrbanKGent-13B achieves (15.56%, 14.29%, 14.89%, and 11.90%) improvements compared with the state-of-the-art GPT-4 with the same inference pipeline on NYC. The improvements on CHI are (15.22%, 17.07%, 13.46%, and 13.95%), respectively. Moreover, the UrbanKGent-7/8B also achieve comparable performance compared with the GPT-4.

Meanwhile, we observe that the zero-shot LLMs perform poorly in the UrbanKGC tasks, even using GPT-4. In addition, although the demonstrations provided by In-context-learning can incorporate the UrbanKGC task information, the performance gain is limited. Besides, we find that fine-tuning LLMs can make obvious improvements in the overall performance. Through vanilla fine-tuning, the Llama-2-7/13B and Llama-3-8B could achieve comparable performance with GPT-3.5 under the ZSL settings.

Moreover, although the various LLM backbones using the UrbanKGent inference pipeline perform slightly worse than the vanilla fine-tuning method, they could obtain better performance compared with zero-shot reasoning and In-context learning paradigms. Such results demonstrate the benefit of knowledgeable instruction design and external tool innovation, but also indicate its performance bottleneck. As a deeper exploration, our work fills this gap through hybrid instruction fine-tuning, and the fine-tuned UrbanKGC agents, whether 7B, 8B or 13B, can achieve state-of-the-art performance in UrbanKGC tasks. We provide an in-depth analysis of the proposed UrbanKGent framework in Appdendix E.2.

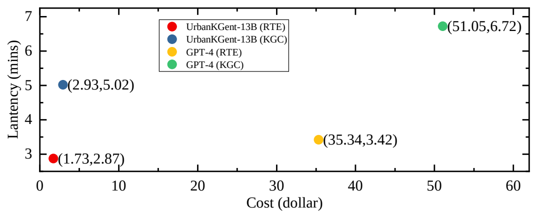


Figure 5: The model latency and cost of constructed UrbanKGent-13B and GPT-4 in UrbanKGC. We report the total inference time and cost of 1,000 RTE and KGC tasks.

5.3 Agent Application

We first derive UrbanKGent-13B for initial UrbanKGs acquisition in New York City and Chicago. After proper filtering and merging of the triplets, we obtain two large-scale UrbanKGs shown in Table 4. Compared with existing UrbanKG benchmark [8], we only use roughly one-fifth of the data for constructing the UrbanKGs with the same scale of triplets and entities, and even expanding the variety of relationships to a hundred times the original types. Moreover, we also provide efficiency analysis in Figure 5. As can be seen, UrbanKGent-13B achieves lower inference speed in latency and reduce the cost by roughly 20 times in both of RTE and KGC tasks. More details is in Appendix E.3.

6 Related work

Domain-Oriented Agent Construction. The concept of language agent [34] has become very popular recently, and a variety of LLM agents targeting different domains have been proposed. For example, Voyager [16] is constructed for automated game exploration, WebGPT [17] is an HTML agent for diverse document understanding tasks, LLMLight [53] constructs a language agent for transportation domain, K2 [13], GeoGalactica [19] and GeoLLM [12] propose to re-train language agent for geospatial semantic understanding. In addition, many recent works like Auto-GPT [54] and CAMEL [55] aim at proposing an autonomous agent framework for agent construction. Nevertheless, there is still no UrbanKGC agent construction framework for the urban computing domain.

LLMs for Knowledge Graph Construction. Recently, the advent of LLMs [56] invigorated the field of NLP. Many studies have begun to explore the potential of LLMs in the domain of KG construction. For example, [32, 57] finds that transforming the NER and RE task into a multi-turn question-answering dialog could improve the model performance. [9] explicitly derive syntactic knowledge to guide LLMs to think, which could develop the performance of NER. Despite these LLM-driven KG construction methods [58, 40] in general domains being widely investigated, KG construction in urban domain still remains an open challenge [59].

Urban Knowledge Graph. Urban knowledge graph has been proven useful in various urban tasks, such as traffic flow prediction [60, 61, 27, 62], mobility prediction [6], site selection [7], city profiling [63], crime prediction and so on [8, 64, 65]. Their common approach involves manually extracting urban entities and defining urban relations to construct an urban knowledge graph. For example, [6] construct a dedicated spatiotemporal knowledge graph regarding trajectory and timestamp as entities to improve trajectory prediction and [7] construct user check-in relations to help mobility prediction. Nevertheless, existing UrbanKGs heavily rely on manual design, leading to high labor costs.

7 Conclusion

In this work, we proposed UrbanKGent, the first automatic UrbanKG construction agent framework with LLMs. We first constructed a knowledgeable instruction set to adopt LLMs for different UrbanKGC tasks. Then, we proposed a tool-augmented iterative trajectory refinement module to facilitate the instruction tuning of various large language models. Extensive experimental results demonstrate the advancement of UrbanKGent in improving UrbanKGC tasks. The obtained UrbanKGent agent family, consisting of 7/8/13B version, with lower latency and cost compared with derving GPT-4 for UrbanKG construction. We hope the open-source UrbanKGent can foster future urban knowledge graph research and broader smart city applications.

Acknowledgments and Disclosure of Funding

This work was supported by the National Natural Science Foundation of China (Grant No.62102110, No.92370204), National Key R&D Program of China (Grant No.2023YFF0725004), Guangzhou-HKUST(GZ) Joint Funding Program (Grant No.2023A03J0008), Education Bureau of Guangzhou Municipality.

References

[1] Weijia Zhang, Jindong Han, Zhao Xu, Hang Ni, Hao Liu, and Hui Xiong. Urban foundation models: A survey. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 6633–6643, 2024.

[2] Fangzhou Sun, Abhishek Dubey, and Jules White. Dxnat—deep neural networks for explaining non-recurring traffic congestion. In *2017 IEEE International Conference on Big Data (IEEE BigData 2017), Boston, MA, USA, December 11-14, 2017*, pages 2141–2150, 2017.

[3] Quanjun Chen, Xuan Song, Harutoshi Yamada, and Ryosuke Shibasaki. Learning deep representation from big and heterogeneous data for traffic accident inference. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pages 338–344, 2016.

[4] Weiyu Cheng, Yanyan Shen, Yanmin Zhu, and Linpeng Huang. A neural attention model for urban air quality inference: Learning the weights of monitoring stations. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, AAAI 2018, USA, February 2-7, 2018*, pages 2151–2158, 2018.

[5] Ying Sun, Hengshu Zhu, Lu Wang, Le Zhang, and Hui Xiong. Large-scale online job search behaviors reveal labor market shifts amid covid-19. *Nature Cities*, 1(2):150–163, 2024.

[6] Huandong Wang, Qiaohong Yu, Yu Liu, Depeng Jin, and Yong Li. Spatio-temporal urban knowledge graph enabled mobility prediction. pages 184:1–184:24, 2021.

[7] Yu Liu, Jingtao Ding, and Yong Li. Developing knowledge graph based system for urban computing. In *Proceedings of the 1st ACM SIGSPATIAL International Workshop on Geospatial Knowledge Graphs*, pages 3–7, 2022.

[8] Yansong Ning, Hao Liu, Hao Wang, Zhenyu Zeng, and Hui Xiong. Uukg: unified urban knowledge graph dataset for urban spatiotemporal prediction. *Advances in Neural Information Processing Systems*, 36, 2023.

[9] Tingyu Xie, Qi Li, Jian Zhang, Yan Zhang, Zuozhu Liu, and Hongwei Wang. Empirical study of zero-shot ner with chatgpt. *arXiv preprint arXiv:2310.10035*, 2023.

[10] Hengyi Zheng, Rui Wen, Xi Chen, Yifan Yang, Yunyan Zhang, Ziheng Zhang, Ningyu Zhang, Bin Qin, Ming Xu, and Yefeng Zheng. Prgc: Potential relation and global correspondence based joint relational triple extraction. *arXiv preprint arXiv:2106.09895*, 2021.

[11] Liang Yao, Jiazhen Peng, Chengsheng Mao, and Yuan Luo. Exploring large language models for knowledge graph completion. *arXiv preprint arXiv:2308.13916*, 2023.

[12] Zekun Li, Wenxuan Zhou, Yao-Yi Chiang, and Muhao Chen. Geolm: Empowering language models for geospatially grounded language understanding. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5227–5240, 2023.

[13] Cheng Deng, Tianhang Zhang, Zhongmou He, Qiyuan Chen, Yuanyuan Shi, Le Zhou, Luoyi Fu, Weinan Zhang, Xinbing Wang, Chenghu Zhou, et al. Learning a foundation language model for geoscience knowledge understanding and utilization. *arXiv preprint arXiv:2306.05064*, 2023.

[14] Wangchunshu Zhou et al. Agents: An open-source framework for autonomous language agents. *ArXiv*, abs/2309.07870, 2023.

[15] Zhiheng Xi et al. The rise and potential of large language model based agents: A survey. *ArXiv*, abs/2309.07864, 2023.

[16] Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*, 2023.

[17] Izzeddin Gur, Hiroki Furuta, Austin Huang, Mustafa Safdari, Yutaka Matsuo, Douglas Eck, and Aleksandra Faust. A real-world webagent with planning, long context understanding, and program synthesis. *arXiv preprint arXiv:2307.12856*, 2023.

[18] Yao Fu, Hao Peng, Litu Ou, Ashish Sabharwal, and Tushar Khot. Specializing smaller language models towards multi-step reasoning. *arXiv preprint arXiv:2301.12726*, 2023.

[19] Zhouhan Lin, Cheng Deng, Le Zhou, Tianhang Zhang, Yi Xu, Yutong Xu, Zhongmou He, Yuanyuan Shi, Beiya Dai, Yunchong Song, et al. Geogalactica: A scientific large language model in geoscience. *arXiv preprint arXiv:2401.00434*, 2023.

[20] Simon Frieder, Luca Pinchetti, Ryan-Rhys Griffiths, Tommaso Salvatori, Thomas Lukasiewicz, Philipp Christian Petersen, Alexis Chevalier, and Julius Berner. Mathematical capabilities of chatgpt. *arXiv preprint arXiv:2301.13867*, 2023.

[21] Beichen Zhang, Kun Zhou, Xilin Wei, Wayne Xin Zhao, Jing Sha, Shijin Wang, and Ji-Rong Wen. Evaluating and improving tool-augmented computation-intensive math reasoning. *arXiv preprint arXiv:2306.02408*, 2023.

[22] Prabin Bhandari, Antonios Anastasopoulos, and Dieter Pfoser. Are large language models geospatially knowledgeable? In *Proceedings of the 31st ACM International Conference on Advances in Geographic Information Systems*, pages 1–4, 2023.

[23] Peter Mooney, Wencong Cui, Boyuan Guan, and Levente Juhász. Towards understanding the geospatial skills of chatgpt: Taking a geographic information systems (gis) exam. In *Proceedings of the 6th ACM SIGSPATIAL International Workshop on AI for Geographic Knowledge Discovery*, pages 85–94, 2023.

[24] Yu Liu, Jingtao Ding, Yanjie Fu, and Yong Li. Urbankg: An urban knowledge graph system. *ACM Transactions on Intelligent Systems and Technology*, pages 1–25, 2023.

[25] Hugo Touvron, Louis Martin, and Kevin Stone et al. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288, 2023.

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

[27] Jia Liu, Tianrui Li, Shenggong Ji, Peng Xie, Shengdong Du, Fei Teng, and Junbo Zhang. Urban flow pattern mining based on multi-source heterogeneous data fusion and knowledge graph embedding. *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[28] Shiyuan Zhang, Tong Li, Shuodi Hui, Guangyu Li, Yanping Liang, Li Yu, Depeng Jin, and Yong Li. Deep transfer learning for city-scale cellular traffic generation through urban knowledge graph. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 4842–4851, 2023.

[29] Eunjoon Cho, Seth A Myers, and Jure Leskovec. Friendship and mobility: user movement in location-based social networks. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1082–1090, 2011.

[30] Inderjeet Mani, Christy Doran, Dave Harris, Janet Hitzeman, Rob Quimby, Justin Richer, Ben Wellner, Scott Mardis, and Seamus Clancy. Spatialml: annotation scheme, resources, and evaluation. *Language Resources and Evaluation*, pages 263–280, 2010.

[31] Rohin Manvi, Samar Khanna, Gengchen Mai, Marshall Burke, David Lobell, and Stefano Ermon. Geollm: Extracting geospatial knowledge from large language models. *arXiv preprint arXiv:2310.06213*, 2023.

[32] Xiang Wei, Xingyu Cui, Ning Cheng, Xiaobin Wang, Xin Zhang, Shen Huang, Pengjun Xie, Jinan Xu, Yufeng Chen, Meishan Zhang, et al. Zero-shot information extraction via chatting with chatgpt. *arXiv preprint arXiv:2302.10205*, 2023.

[33] Aohan Zeng, Mingdao Liu, Rui Lu, Bowen Wang, Xiao Liu, Yuxiao Dong, and Jie Tang. Agenttuning: Enabling generalized agent abilities for llms. *arXiv preprint arXiv:2310.12823*, 2023.

[34] Baian Chen, Chang Shu, Ehsan Shareghi, Nigel Collier, Karthik Narasimhan, and Shunyu Yao. Fireact: Toward language agent fine-tuning. *arXiv preprint arXiv:2310.05915*, 2023.

[35] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. pages 24824–24837, 2022.

[36] Tianle Cai, Xuezhi Wang, Tengyu Ma, Xinyun Chen, and Denny Zhou. Large language models as tool makers. *arXiv preprint arXiv:2305.17126*, 2023.

[37] Ruixin Hong, Hongming Zhang, Hong Zhao, Dong Yu, and Changshui Zhang. Faithful question answering with monte-carlo planning. *arXiv preprint arXiv:2305.02556*, 2023.

[38] Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. Large language models cannot self-correct reasoning yet. *arXiv preprint arXiv:2310.01798*, 2023.

[39] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. *arXiv preprint arXiv:2303.17651*, 2023.

[40] Keming Lu, Xiaoman Pan, Kaiqiang Song, Hongming Zhang, Dong Yu, and Jianshu Chen. Pivoine: Instruction tuning for open-world information extraction. *arXiv preprint arXiv:2305.14898*, 2023.

[41] Hongbin Ye, Ningyu Zhang, Hui Chen, and Huajun Chen. Generative knowledge graph construction: A review. *arXiv preprint arXiv:2210.12714*, 2022.

[42] Yew Ken Chia, Lidong Bing, Soujanya Poria, and Luo Si. Relationprompt: Leveraging prompts to generate synthetic data for zero-shot relation triplet extraction. *arXiv preprint arXiv:2203.09101*, 2022.

[43] Liang Yao, Chengsheng Mao, and Yuan Luo. Kg-bert: Bert for knowledge graph completion. *arXiv preprint arXiv:1909.03193*, 2019.

[44] Apoorv Saxena, Adrian Kochsiek, and Rainer Gemulla. Sequence-to-sequence knowledge graph completion and question answering. *arXiv preprint arXiv:2203.10321*, 2022.

[45] Cunxiang Wang, Sirui Cheng, Zhikun Xu, Bowen Ding, Yidong Wang, and Yue Zhang. Evaluating open question answering evaluation. *arXiv preprint arXiv:2305.12421*, 2023.

[46] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna. lmsys. org (accessed 14 April 2023)*, 2(3):6, 2023.

[47] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. Stanford alpaca: An instruction-following llama model, 2023.

[48] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

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

[50] AI@Meta. Llama 3 model card. 2024.

[51] OpenAI. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[52] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *arXiv preprint arXiv:2306.05685*, 2023.

[53] Siqi Lai, Zhao Xu, Weijia Zhang, Hao Liu, and Hui Xiong. Large language models as traffic signal control agents: Capacity and opportunity. *arXiv preprint arXiv:2312.16044*, 2023.

[54] Significant-gravitas/auto-gpt: An experimental open-source attempt to make gpt-4 fully autonomous. 2023.

[55] Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. Camel: Communicative agents for" mind" exploration of large scale language model society. *arXiv preprint arXiv:2303.17760*, 2023.

[56] Zheng Gong and Ying Sun. Graph reasoning enhanced language models for text-to-sql. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2447–2451, 2024.

[57] Guozheng Li, Peng Wang, and Wenjun Ke. Revisiting large language models as zero-shot relation extractors. *arXiv preprint arXiv:2310.05028*, 2023.

[58] Chenhan Yuan, Qianqian Xie, and Sophia Ananiadou. Zero-shot temporal relation extraction with chatgpt. *arXiv preprint arXiv:2304.05454*, 2023.

[59] Yueling Zeng and Li-C Wang. Domain knowledge graph construction via a simple checker. *arXiv preprint arXiv:2310.04949*, 2023.

[60] Chengbiao Yang and Guilin Qi. An urban traffic knowledge graph-driven spatial-temporal graph convolutional network for traffic flow prediction. In *Proceedings of the 11th International Joint Conference on Knowledge Graphs*, pages 110–114, 2022.

[61] Jiyuan Tan, Qianqian Qiu, Weiwei Guo, and Tingshuai Li. Research on the construction of a knowledge graph and knowledge reasoning model in the field of urban traffic. *Sustainability*, page 3191, 2021.

[62] Ling Zhao, Hanhan Deng, Linyao Qiu, Sumin Li, Zhixiang Hou, Hai Sun, and Yun Chen. Urban multi-source spatio-temporal data analysis aware knowledge graph embedding. *Symmetry*, page 199, 2020.

[63] Zhilun Zhou, Yu Liu, Jingtao Ding, Depeng Jin, and Yong Li. Hierarchical knowledge graph learning enabled socioeconomic indicator prediction in location-based social network. In *Proceedings of the ACM Web Conference*, pages 122–132, 2023.

[64] Ruiyi Yang, Flora D Salim, and Hao Xue. Sstkg: Simple spatio-temporal knowledge graph for intepretable and versatile dynamic information embedding. *arXiv preprint arXiv:2402.12132*, 2024.

[65] Jindong Han, Hao Liu, Hengshu Zhu, and Hui Xiong. Kill two birds with one stone: A multi-view multi-adversarial learning approach for joint air quality and weather prediction. *IEEE Transactions on Knowledge and Data Engineering*, 35(11):11515–11528, 2023.

[66] Qinyuan Cheng, Tianxiang Sun, Wenwei Zhang, Siyin Wang, Xiangyang Liu, Mozhi Zhang, Junliang He, Mianqiu Huang, Zhangyue Yin, Kai Chen, et al. Evaluating hallucinations in chinese large language models. *arXiv preprint arXiv:2310.03368*, 2023.

[67] Jinlan Fu, See-Kiong Ng, Zhengbao Jiang, and Pengfei Liu. Gptscore: Evaluate as you desire. *arXiv preprint arXiv:2302.04166*, 2023.

[68] Jerrold H Zar. Spearman rank correlation. *Encyclopedia of biostatistics*, 2005.

Table 5: The detailed statistic of RTE datasets. We report the maximum length, minimum length, and average length of urban text in the RTE dataset.

Dataset	Max Length	Min Length	Avg Length	# Records
NYC-Instruct	1,747	68	437	232
CHI-Instruct	1,120	25	408	122
NYC	2,708	51	433	2,089
CHI	1,883	32	445	1,102
NYC-Large	4,598	20	1,179	40,480
CHI-Large	4,597	36	825	28,868

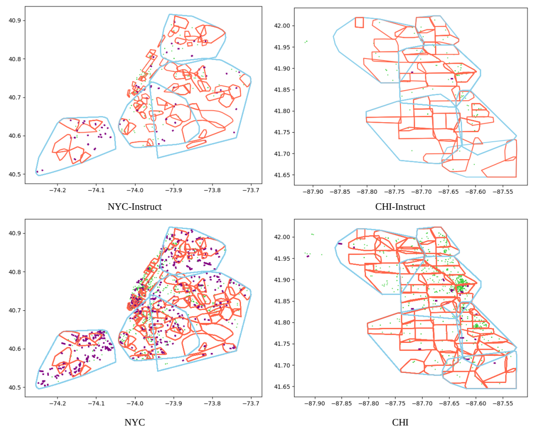


Figure 7: The geometry range visualization of the head entity and tail entity of four KGC datasets. The horizontal and vertical coordinates are longitude and latitude, respectively. The blue and red polygons stand for entities with the polygon geometry, the purple line string stands for the entities with linestring geometry and the green point is for the coordinate entities.

A UrbanKGC Construction

This section presents detailed UrbanKGC dataset statistic information for relational triplet extraction (RTE) and knowledge graph completion (KGC) tasks. Since two sequential tasks (i.e., RTE and KGC) of UrbanKGC are within an open-world setting (i.e., no predefined ontology) [40, 41]. Therefore, for the RTE task, every data record is an urban text without the triplet label. For the KGC task, every data record is a quadruple (i.e., head entity name, head entity geometry, tail entity name, tail entity geometry) without the geospatial relation label.

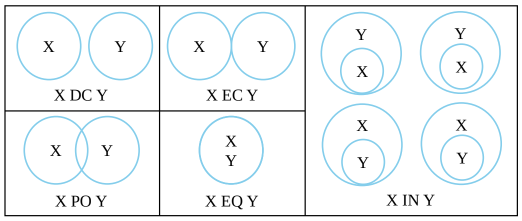


Figure 6: Given two geo-entities X and Y, the illustrative visualization of five types of RCC relationships. In this work, we consider entities with coordinate geometry to a circles with very small radii.

RTE Dataset. To facilitate the understanding of the constructed RTE dataset, we summarize the distribution of urban textual corpus in the six RTE datasets. As shown in Table 5, the entity distribution and textual statistics of the instruct dataset and test data are similar. In addition, the record in dataset (i.e., NYC-Instruct and CHI-Instruct) used for instruction tuning is not overlapping with that in test dataset (i.e., NYC and CHI) and the real-world application dataset (i.e., NYC-Large and CHI-Large). This can avoid potential data leakage issues.

KGC Dataset. As for the KGC dataset, we provide illustrative visualization of the five RCC relationships [30] in Figure 6 for better understanding. Specifically, the disconnected (DC), externally connected (EC), partially overlapping (PO), equal (EQ), tangential and non-tangential proper parts

Table 6: The detailed geospatial tool name, description, interface input, and output. Each tool interface is implemented by python and it is self-programmed by GPT-4.

Tool name	Tool description	Input	Output
Geohash	Geohash encoding	Geometry	Geohash code (8-bit)
Distance	Calculate the distance between two geo entities.	Geometry 1, Geometry 2	Distance value (km)
Point2Polygon	Identify if a point belongs to a polygon	Point geometry, polygon geometry	True/False
Point4Linestring	Identify if a point intersects a linestring	Point geometry, linestring geometry	True/False
Linestring2Polygon	Identify if a linestring belongs to a polygon	Linestring geometry, polygon geometry	True/False
Linestring4Polygon	Identify if a linestring intersects a polygon	Linestring geometry, polygon geometry	True/False
Polygon2Polygon	Identify if a polygon belongs to a polygon	Polygon geometry, polygon geometry	True/False
Polygon4Polygon	Identify if a polygon intersects a polygon	Polygon geometry, polygon geometry	True/False

(IN) together depict the basic geospatial relationship between urban entities. Moreover, to facilitate the understanding of the constructed KGC dataset, we visualize the geometry range of the head entity and the tail entity in four small KGC datasets. Due to the large amount of data and the display overlapping between entities and entities during visualization, we will not show the visualization results of NYC-Large and CHI-Large dataset, but the pattern is similar. As can be seen in Figure 7, the entity distribution in the instruct dataset (i.e., NYC-Instruct and CHI-Instruct) and test dataset (i.e., NYC and CHI) are similar. The pattern in NYC-Large and CHI-Large is similar. Due to the KGC task follows a zero-shot setting, we are unable to provide accurate distribution proportions of the 5 RCC relationships in the four datasets. However, it is intuitive can be seen that all four datasets contain the five RCC relationships shown in Figure 6, which provides a guarantee for the practical significance of KGC task in this work.

B Instruction Template

B.1 Instruction Template in UrbanKGent

This section presents the detailed instruction template of the proposed UrbanKGent framework. Specifically, Figure 8(a-b) provides the detailed instruction template of knowledgeable instruction generation module and tool-based trajectory augmentation. The iterative trajectory self-refinement module is achieved by trajectory verifier and updater in Figure 8(c).

B.2 Instruction Template in Baselines

We provide the detailed instruction template of all the baseline models in this work.

LLMs-based Zero-shot Reasoning Methods. We only provide task descriptions for the zero-shot reasoning method. The detailed instruction template is shown in Figure 9.

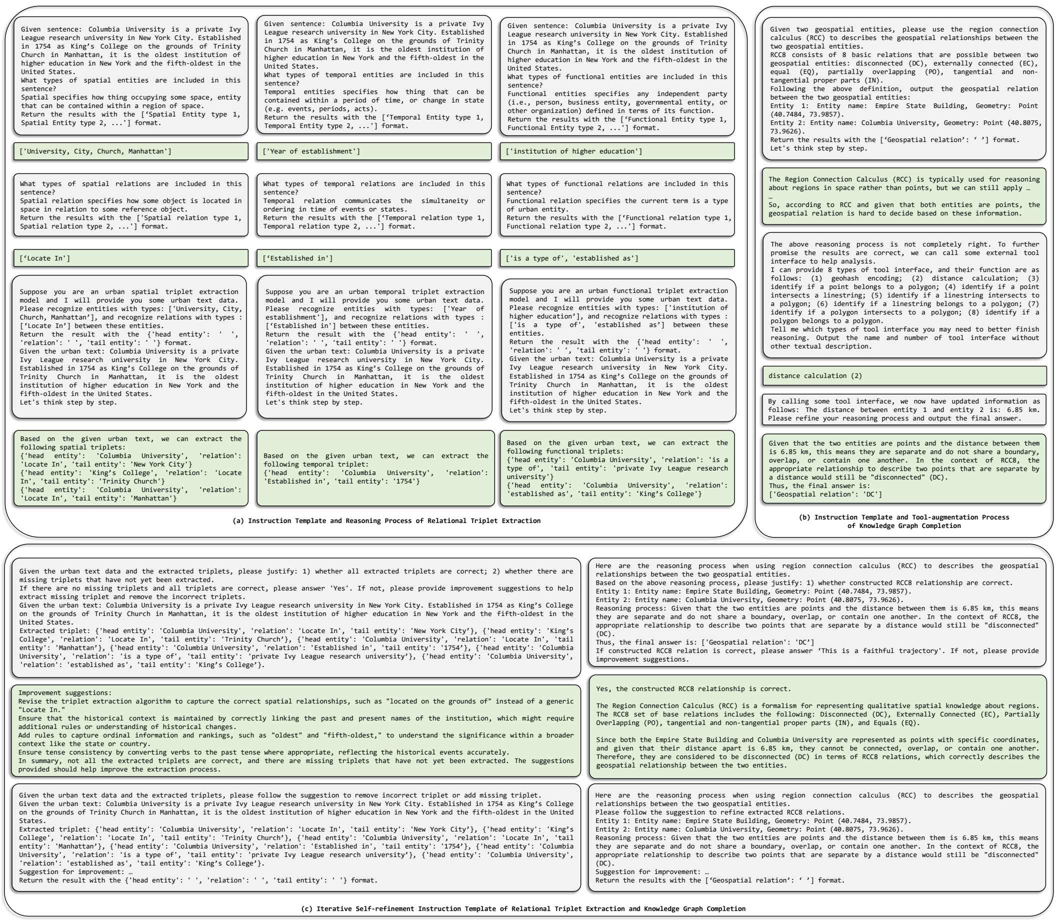
LLMs-based In-context Learning Methods. We first construct several few-shot demonstrations via the chain-of-thought prompting techniques [35], which is popular for automatic demonstration generation. The detailed instruction template is shown in Figure 9. Then, we add these demonstrations before the test question, and the detailed instruction template is shown in Figure 9.

Vanilla Fine-tuning Methods. We only provide task descriptions for vanilla fine-tuning methods without demonstrations. The detailed instruction template is shown in Figure 9.

UrbanKGent Inference Methods. These baseline models follow the same inference pipeline shown in Figure 8.

C Geospatial Toolkit

This work constructs a geospatial reasoning toolkit (eight interfaces in total in Table 6) by asking GPT-4 for self-programming. We obtained geospatial toolkit for point, linestring, and polygon geometry objects, supporting distance calculation and Geohash encoding for two geographical objects, and can also determine basic spatial relationships between two geometric objects, such as containment and intersection. For each interface, we provide its name and function description (e.g., *geohash encoding* in Table 6 and Figure 8b) into the instruction template.



Given sentence: Columbia University is a private Ivy League research university in New York City. Established in 1754 as King’s College on the grounds of Trinity Church in Manhattan, it is the oldest institution of higher education in New York and the fifth-oldest in the United States.
What types of spatial entities are included in this sentence?
Spatial specifies how thing occupying some space, entity that can be contained within a region of space.
Return the results with the [‘Spatial Entity type 1, Spatial Entity type 2, ...'] format.

['University, City, Church, Manhattan']

What types of spatial relations are included in this sentence?
Spatial relation specifies how some object is located in space in relation to some reference object.
Return the results with the ['Spatial relation type 1, Spatial relation type 2, ...'] format.

[‘Locate In']

Suppose you are an urban spatial triplet extraction model and I will provide you some urban text data. Please recognize entities with types: ['University, City, Church, Manhattan’], and recognize relations with types : [‘Locate In’] between these entities.
Return the result with the {'head entity': ' ', 'relation': ' ', 'tail entity': ' '} format.
Given the urban text: Columbia University is a private Ivy League research university in New York City. Established in 1754 as King’s College on the grounds of Trinity Church in Manhattan, it is the oldest institution of higher education in New York and the fifth-oldest in the United States.
Let's think step by step.

Based on the given urban text, we can extract the following spatial triplets:
{'head entity': 'Columbia University', 'relation': 'Locate In', 'tail entity': 'New York City'}
{'head entity': 'King’s College', 'relation': 'Locate In', 'tail entity': 'Trinity Church'}
{'head entity': 'Columbia University', 'relation': 'Locate In', 'tail entity': 'Manhattan'}

Given sentence: Columbia University is a private Ivy League research university in New York City. Established in 1754 as King’s College on the grounds of Trinity Church in Manhattan, it is the oldest institution of higher education in New York and the fifth-oldest in the United States.
What types of temporal entities are included in this sentence?
Temporal entities specifies how thing that can be contained within a period of time, or change in state (e.g. events, periods, acts).
Return the results with the [‘Temporal Entity type 1, Temporal Entity type 2, ...'] format.

['Year of establishment']

What types of temporal relations are included in this sentence?
Temporal relation communicates the simultaneity or ordering in time of events or states.
Return the results with the [‘Temporal relation type 1, Temporal relation type 2, ...'] format.

[‘Established in']

Suppose you are an urban temporal triplet extraction model and I will provide you some urban text data. Please recognize entities with types: ['Year of establishment'], and recognize relations with types : [‘Established in'] between these entities.
Return the result with the {'head entity': ' ', 'relation': ' ', 'tail entity': ' '} format.
Given the urban text: Columbia University is a private Ivy League research university in New York City. Established in 1754 as King’s College on the grounds of Trinity Church in Manhattan, it is the oldest institution of higher education in New York and the fifth-oldest in the United States.
Let's think step by step.

Based on the given urban text, we can extract the following temporal triplet:
{'head entity': 'Columbia University', 'relation': 'Established in', 'tail entity': '1754'}

Given sentence: Columbia University is a private Ivy League research university in New York City. Established in 1754 as King’s College on the grounds of Trinity Church in Manhattan, it is the oldest institution of higher education in New York and the fifth-oldest in the United States.
What types of functional entities are included in this sentence?
Functional entities specifies any independent party (i.e., person, business entity, governmental entity, or other organization) defined in terms of its function.
Return the results with the [‘Functional Entity type 1, Functional Entity type 2, ...'] format.

['institution of higher education']

What types of functional relations are included in this sentence?
Functional relation specifies the current term is a type of urban entity.
Return the results with the [‘Functional relation type 1, Functional relation type 2, ...'] format.

['is a type of', 'established as']

Suppose you are an urban functional triplet extraction model and I will provide you some urban text data. Please recognize entities with types: ['institution of higher education’], and recognize relations with types : ['is a type of', 'established as’] between these entities.
Return the result with the {'head entity': ' ', 'relation': ' ', 'tail entity': ' '} format.
Given the urban text: Columbia University is a private Ivy League research university in New York City. Established in 1754 as King’s College on the grounds of Trinity Church in Manhattan, it is the oldest institution of higher education in New York and the fifth-oldest in the United States.
Let's think step by step.

Based on the given urban text, we can extract the following functional triplets:
{'head entity': 'Columbia University', 'relation': 'is a type of', 'tail entity': 'private Ivy League research university'}
{'head entity': 'Columbia University', 'relation': 'established as', 'tail entity': 'King’s College'}

(a) Instruction Template and Reasoning Process of Relational Triplet Extraction

Given two geospatial entities, please use the region connection calculus (RCC) to describes the geospatial relationships between the two geospatial entities.
RCC8 consists of 8 basic relations that are possible between two geospatial entities: disconnected (DC), externally connected (EC), equal (EQ), partially overlapping (PO), tangential and non-tangential proper parts (IN).
Following the above definition, output the geospatial relation between the two geospatial entities:
Entity 1: Entity name: Empire State Building, Geometry: Point (40.7484, 73.9857).
Entity 2: Entity name: Columbia University, Geometry: Point (40.8075, 73.9626).
Return the results with the [‘Geospatial relation’: ‘ ’] format.
Let's think step by step.

The Region Connection Calculus (RCC) is typically used for reasoning about regions in space rather than points, but we can still apply …
…
So, according to RCC and given that both entities are points, the geospatial relation is hard to decide based on these information.

The above reasoning process is not completely right. To further promise the results are correct, we can call some external tool interface to help analysis.
I can provide 8 types of tool interface, and their function are as follows: (1) geohash encoding; (2) distance calculation; (3) identify if a point belongs to a polygon; (4) identify if a point intersects a linestring; (5) identify if a linestring intersects to a polygon; (6) identify if a linestring belongs to a polygon; (7) identify if a polygon intersects to a polygon; (8) identify if a polygon belongs to a polygon.
Tell me which types of tool interface you may need to better finish reasoning. Output the name and number of tool interface without other textual description.

distance calculation (2)

By calling some tool interface, we now have updated information as follows: The distance between entity 1 and entity 2 is: 6.85 km. Please refine your reasoning process and output the final answer.

Given that the two entities are points and the distance between them is 6.85 km, this means they are separate and do not share a boundary, overlap, or contain one another. In the context of RCC8, the appropriate relationship to describe two points that are separate by a distance would still be "disconnected" (DC).
Thus, the final answer is:
['Geospatial relation': 'DC']

(b) Instruction Template and Tool-augmentation Process of Knowledge Graph Completion

Given the urban text data and the extracted triplets, please justify: 1) whether all extracted triplets are correct; 2) whether there are missing triplets that have not yet been extracted.
If there are no missing triplets and all triplets are correct, please answer 'Yes'. If not, please provide improvement suggestions to help extract missing triplet and remove the incorrect triplets.
Given the urban text: Columbia University is a private Ivy League research university in New York City. Established in 1754 as King’s College on the grounds of Trinity Church in Manhattan, it is the oldest institution of higher education in New York and the fifth-oldest in the United States.
Extracted triplet: {'head entity': 'Columbia University', 'relation': 'Locate In', 'tail entity': 'New York City’}, {'head entity': 'King’s College', 'relation': 'Locate In', 'tail entity': 'Trinity Church’}, {'head entity': 'Columbia University', 'relation': 'Locate In', 'tail entity': 'Manhattan’}, {'head entity': 'Columbia University', 'relation': 'Established in', 'tail entity': '1754’}, {'head entity': 'Columbia University', 'relation': 'is a type of', 'tail entity': 'private Ivy League research university’}, {'head entity': 'Columbia University', 'relation': 'established as', 'tail entity': 'King’s College’}.

Improvement suggestions:
Revise the triplet extraction algorithm to capture the correct spatial relationships, such as "located on the grounds of" instead of a generic "Locate In."
Ensure that the historical context is maintained by correctly linking the past and present names of the institution, which might require additional rules or understanding of historical changes.
Add rules to capture ordinal information and rankings, such as "oldest" and "fifth-oldest," to understand the significance within a broader context like the state or country.
Ensure tense consistency by converting verbs to the past tense where appropriate, reflecting the historical events accurately.
In summary, not all the extracted triplets are correct, and there are missing triplets that have not yet been extracted. The suggestions provided should help improve the extraction process.

Given the urban text data and the extracted triplets, please follow the suggestion to remove incorrect triplet or add missing triplet.
Given the urban text: Columbia University is a private Ivy League research university in New York City. Established in 1754 as King’s College on the grounds of Trinity Church in Manhattan, it is the oldest institution of higher education in New York and the fifth-oldest in the United States.
Extracted triplet: {'head entity': 'Columbia University', 'relation': 'Locate In', 'tail entity': 'New York City’}, {'head entity': 'King’s College', 'relation': 'Locate In', 'tail entity': 'Trinity Church’}, {'head entity': 'Columbia University', 'relation': 'Locate In', 'tail entity': 'Manhattan’}, {'head entity': 'Columbia University', 'relation': 'Established in', 'tail entity': '1754’}, {'head entity': 'Columbia University', 'relation': 'is a type of', 'tail entity': 'private Ivy League research university’}, {'head entity': 'Columbia University', 'relation': 'established as', 'tail entity': 'King’s College’}.
Suggestion for improvement: …
Return the result with the {'head entity': ' ', 'relation': ' ', 'tail entity': ' '} format.

Here are the reasoning process when using region connection calculus (RCC) to describes the geospatial relationships between the two geospatial entities.
Based on the above reasoning process, please justify: 1) whether constructed RCC8 relationship are correct.
Entity 1: Entity name: Empire State Building, Geometry: Point (40.7484, 73.9857).
Entity 2: Entity name: Columbia University, Geometry: Point (40.8075, 73.9626).
Reasoning process: Given that the two entities are points and the distance between them is 6.85 km, this means they are separate and do not share a boundary, overlap, or contain one another. In the context of RCC8, the appropriate relationship to describe two points that are separate by a distance would still be "disconnected" (DC).
Thus, the final answer is: ['Geospatial relation': 'DC’]
If constructed RCC8 relation is correct, please answer ‘This is a faithful trajectory'. If not, please provide improvement suggestions.

Yes, the constructed RCC8 relationship is correct.

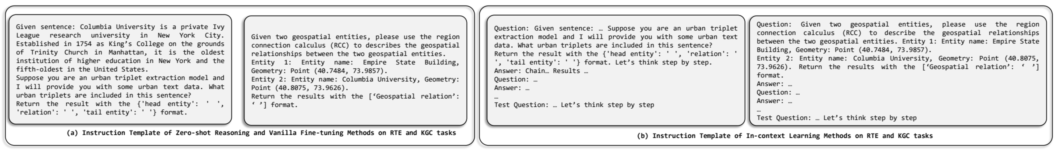
The Region Connection Calculus (RCC) is a formalism for representing qualitative spatial knowledge about regions. The RCC8 set of base relations includes the following: Disconnected (DC), Externally Connected (EC), Partially Overlapping (PO), tangential and non-tangential proper parts (IN), and Equals (EQ).

Since both the Empire State Building and Columbia University are represented as points with specific coordinates, and given that their distance apart is 6.85 km, they cannot be connected, overlap, or contain one another. Therefore, they are considered to be disconnected (DC) in terms of RCC8 relations, which correctly describes the geospatial relationship between the two entities.

Here are the reasoning process when using region connection calculus (RCC) to describes the geospatial relationships between the two geospatial entities.
Please follow the suggestion to refine extracted RCC8 relations.
Entity 1: Entity name: Empire State Building, Geometry: Point (40.7484, 73.9857).
Entity 2: Entity name: Columbia University, Geometry: Point (40.8075, 73.9626).
Reasoning process: Given that the two entities are points and the distance between them is 6.85 km, this means they are separate and do not share a boundary, overlap, or contain one another. In the context of RCC8, the appropriate relationship to describe two points that are separate by a distance would still be "disconnected" (DC).
Suggestion for improvement: …
Return the results with the [‘Geospatial relation’: ‘ ’] format.

(c) Iterative Self-refinement Instruction Template of Relational Triplet Extraction and Knowledge Graph Completion

Figure 8: The reasoning process of UrbanKGent-13B and the detailed instruction template for UrbanKent inference pipeline. The content in the gray box is the instruction, and the content in the green box is the agent’s response.



Given sentence: Columbia University is a private Ivy League research university in New York City. Established in 1754 as King’s College on the grounds of Trinity Church in Manhattan, it is the oldest institution of higher education in New York and the fifth-oldest in the United States.
Suppose you are an urban triplet extraction model and I will provide you with some urban text data. What urban triplets are included in this sentence?
Return the result with the {'head entity': ' ', 'relation': ' ', 'tail entity': ' '} format.

Given two geospatial entities, please use the region connection calculus (RCC) to describes the geospatial relationships between the two geospatial entities.
Entity 1: Entity name: Empire State Building, Geometry: Point (40.7484, 73.9857).
Entity 2: Entity name: Columbia University, Geometry: Point (40.8075, 73.9626).
Return the results with the [‘Geospatial relation’: ‘ ’] format.

(a) Instruction Template of Zero-shot Reasoning and Vanilla Fine-tuning Methods on RTE and KGC tasks

Question: Given sentence: … Suppose you are an urban triplet extraction model and I will provide you with some urban text data. What urban triplets are included in this sentence?
Return the result with the {'head entity': ' ', 'relation': ' ', 'tail entity': ' '} format. Let’s think step by step.
Answer: Chain… Results …
Question: …
Answer: …
…
Test Question: … Let’s think step by step

Question: Given two geospatial entities, please use the region connection calculus (RCC) to describe the geospatial relationships between the two geospatial entities. Entity 1: Entity name: Empire State Building, Geometry: Point (40.7484, 73.9857).
Entity 2: Entity name: Columbia University, Geometry: Point (40.8075, 73.9626). Return the results with the [‘Geospatial relation’: ‘ ’] format.
Answer: …
Question: …
Answer: …
…
Test Question: … Let’s think step by step

(b) Instruction Template of In-context Learning Methods on RTE and KGC tasks

Figure 9: The instruction template of zero-shot reasoning, In-context learning and vanilla fine-tuning baselines.

Table 7: Illustrative RTE evaluation example when we utilize human evaluation and GPT evaluation method. We calculate the accuracy by counting the proportion of true triplets. The label for GPT evaluation method is invisible.

Type	Urban text	Results	Label	Number of the true triplet	Number of the false triplet
Human Evaluation	Columbia University is a private Ivy league research university in New York City.	<Columbia University, locate-in, New York City >	<Columbia University, locate-in, New York City >	1	0
GPT Evaluation			-	1	0

D Evaluation

This section presents the detailed evaluation process and examples to help the reader have a better understanding: **(1) Human Evaluation.** We employ human annotators to evaluate the results on 200 random samples. For the relational triplet extraction task, we first manually annotate the triplet label for each sample. Then, we manually evaluate the correctness of each triplet [40] based on annotation and calculate the accuracy value. For the knowledge graph completion task, we follow [11] to manually label the response as correct or wrong, and calculate the accuracy. **(2) GPT Evaluation.** Recently, many studies [52, 45] adopt LLM-based evaluation for open-domain tasks and empirically demonstrate that GPT-4's evaluation and human evaluation can be consistent [66]. In this work, we also use GPT-4 to evaluate the model performance on the full data to escape intensive labor. Specifically, given an UrbanKGC instruction and results, we prompt GPT-4 to return the confidence score and the justification (i.e., True/False), which will be further used to calculate the accuracy.

Moreover, we provide a comprehensive analysis to demonstrate why GPT and Human evaluations are highly aligned.

D.1 Human Evaluation Process

For the relational triplet extraction (RTE) task, we provide an evaluation example in Table 7. Given the urban text sentence "Columbia University is a private Ivy league research university in New York City.", the human annotators are required to first label the triplet contained in this urban text, described as triplet <*Columbia University, locate-in, New York City*>. Based on the label, then, they are instructed to evaluate how many true triplets and false triplets in the results from the models. Finally, they will fill out the evaluation form (i.e., the number of the true triplets and the number of the false triplets). We will calculate the accuracy of results based on these annotated forms.

For the knowledge graph completion (KGC) task, given the head entity <*Columbia University, Point (40.8075, 73.9626)*> and the tail entity <*Empire State Building, Point (40.7484, 73.9857)*>, the human annotators are first required to complete their missing geospatial relationship from the five relation candidate (i.e., DC, EC, PO, EQ and IN). Specifically, the annotation could be achieved by manually visualizing the location of two entities given on the map and following the RCC relation rule in Figure 6 to determine their geographical relationships. Finally, they will fill out the evaluation results (i.e., True/False). We will calculate the accuracy based on these evaluation results.

D.2 GPT Evaluation Process

For the relational triplet extraction (RTE) task, we provide an evaluation example in Table 7. Given the urban text sentence "Columbia University is a private Ivy league research university in New York City.", and the model extraction results (i.e., <*Columbia University, locate-in, New York City*>), we directly prompt GPT-4 to fill out the evaluation form (i.e., the number of the true triplets and the number of the false triplets). Then, the accuracy could be obtained based on these self-evaluated results.

For the knowledge graph completion (KGC) task, given the head entity <*Columbia University, Point (40.8075, 73.9626)*> and the tail entity <*Empire State Building, Point (40.7484, 73.9857)*> and completed geospatial relationship, we directly prompt GPT-4 the justify if the results are correct. Specifically, we will first explicitly call eight external geospatial tools in Table 6, and combine all these eight calculation results into the prompt. Then we feed these prompts into GPT-4 to help it

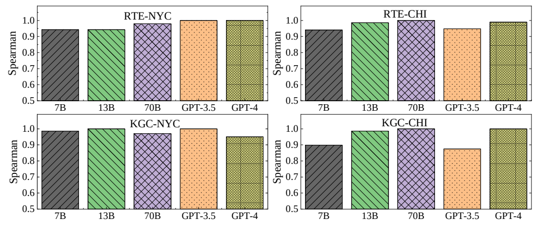


Figure 10: The Spearman correlation between the GPT evaluation and human's evaluation under five different LLM backbones (i.e., Llama-2-7B, Llama-2-13B, Llama-2-70B, GPT-3.5 and GPT-4).

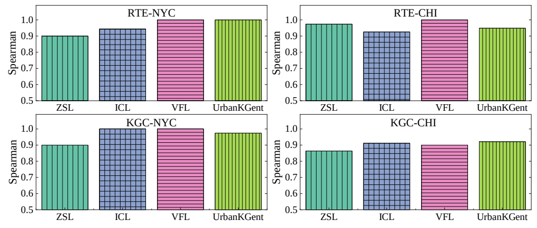


Figure 11: The Spearman correlation between the GPT evaluation and human's evaluation under four LLM paradigms (i.e., Zero-shot learning (ZSL), In-context learning (ICL), Vanilla fine-tuning (VFL) and UrbanKGent inference).

make a comparable accurate evaluation. Finally, we obtain the evaluation results (i.e., True/False) for every record. We will calculate the accuracy based on these evaluation results.

However, such an evaluation process carries risks as we cannot guarantee the infallibility of GPT-4. To ensure the validity of the evaluation, we have conducted extensive experiments to demonstrate that the GPT evaluation method aligns well with human evaluation.

D.3 Evaluation Consistency

We conduct experiments to assess the consistency between GPT-4's evaluation results and human evaluation results. In this work, we follow GPTScore [67] to use Spearman coefficient [68] to investigate the correlation between GPT evaluation and human evaluation. Based on the evaluation results in Tabel 3, we can conduct correlation analysis separately by the types of model and paradigm.

Comparison of LLM Backbones. As shown in Figure 10, we observe that the Spearman correlation coefficients on all LLM backbones are greater than 0.8, which means the human evaluation and GPT evaluation on these five LLM backbones is highly correlated.

Comparison of LLM Paradigm. Moreover, we also conduct correlation under different LLM paradigms for UrbanKGC tasks. As can be seen in Figure 11, all four types of paradigm have a higher Spearman value above 0.85. Therefore, deriving the GPT evaluation method is also applicable to different LLM paradigms when finishing UrbanKGC tasks as its evaluation aligns well with that of the human being.

Analysis of Evaluation Repeat. Nevertheless, the above correlation analysis is based on results in Table 3, which are obtained after only single evaluation. To deeply understand the evaluation mechanism, We repeatedly instruct GPT-4 to generate 1, 3, 5, and 10 evaluation results to understand the variance of the evaluation method. To save the cost, we perform the above analysis using UrbanKGent-7B only on the NYC dataset. The results are shown in Table 8, although the coefficient value will decrease as the number of repeated experiments increases. However its value has always remained within a high correlation range (roughly 0.85).

E In-Depth Analysis

In this section, we aim to perform more comprehensive analysis of our proposed UrbanKGent agent and hope to answer the following research questions. **(1) RQ1**: How does the constructed

Table 8: The average Spearman correlation value between human evaluations and GPT-4 evaluations. "Repeat X times" refers to instructing GPT-4 to generate judgments X times, and adopting the answer that appears most frequently (e.g., True/False for the KGC task and Number of the true triplet for the RTE task) as the final decision.

Task	Repeat Once	Repeat 3 times	Repeat 5 times	Repeat 10 times
RTE	98.56%	87.31%	86.42%	85.27%
KGC	94.86%	91.23%	90.09%	91.24%

Table 9: The experimental results of relational triplet extraction (RTE) and knowledge graph completion (KGC) on the NYC-Large dataset and CHI-Large dataset. To save the cost, We choose the best approaches in zero-shot-learning (ZSL), In-context-learning (ICL), Vanilla Fine-tuning (VFT), and UrbanKGent Inference setting.

Models	**NYC-Large**				**CHI-Large**			
	GPT(acc/confidence)		**Human(acc)**		**GPT(acc/confidence)**		**Human(acc)**	
	RTE	**KGC**	**RTE**	**KGC**	**RTE**	**KGC**	**RTE**	**KGC**
ZSL	0.36/4.01	0.45/3.95	0.42	0.31	0.38/4.27	0.33/3.67	0.44	0.36
ICL	0.39/4.36	0.48/4.15	0.48	0.39	0.38/4.22	0.35/3.53	0.40	0.35
VFT	0.37/4.05	0.46/3.98	0.45	0.35	0.33/4.25	0.28/3.67	0.38	0.34
UrbanKGent Inference	0.43/4.13	0.51/3.87	0.51	0.43	0.47/4.33	0.43/3.63	0.49	0.42
UrbanKGent-7B	0.44/4.27	0.50/4.07	0.53	0.44	0.48/4.25	0.42/3.85	0.55	0. 46
UrbanKGent-8B	0.43/4.07	0.51/4.16	0.52	0.44	0.49/4.21	0.43/3.77	0.53	0. 44
UrbanKGent-13B	0.46/4.56	0.52/3.67	0.55	0.46	0.55/4.29	0.49/3.24	0.58	0.49

UrbanKGent perform compared with existing paradigms on larger real-world dataset? **(2) RQ2**: How do different components (e.g., the knowledgeable instruction generation) affect the performance? **(3) RQ3**: How the complexity and efficiency of proposed UrbanKGent framework? **(4) RQ4**: How about the UrbanKGent inference trajectories when completing UrbanKGC tasks? **(5) RQ5**: How can the constructed UrbanKGent provide application service to real-world scenarios?

E.1 RQ1: Evaluation on Larger Dataset

As shown in Table 9, we derive UrbanKGent-13B for urban knowledge graph construction using constructed large-scale dataset in NYC and CHI. Specifically, we directly use the text record in NYC-Large and CHI-Large for the relational triplet extraction task. Then, we randomly sample the head-tail entity pairs (both of head and tail entities contain geometry information) from these triplets for knowledge graph completion. Nevertheless, iterating all head-tail entity pairs is a time-consuming task, so we just construct a KGC dataset consistent with the sacle of the RTE dataset. By performing RTE and KGC tasks, we obtain two large-scale UrbanKGs shown in Table 4. Compared with existing dataset and benchmark UUKG [8], we can clearly observe that our agent can only use one-fifth data to construct the UrbanKG with the same scale entities and triplets, but extend the relationship types to a thousand times. It is worth noting that all construction process is completed by a LLM agent without any mannul effort. We think it is the core of this work.

Moreover, we also report the performance of the RTE and KGC in NYC-Large dataset. To save the cost of GPT self-evaluation, we only choose the best approaches in ZSL, ICL, VFT and UrbanKGent Inference. The experimental results are shown in Table 9. As can be seen, the fine-tuned UrbanKGC agent, whether 7B or 13B version, could achieve state-of-art performance on UrbanKGC tasks.

E.2 RQ2: Ablation Studies

We conduct an in-depth analysis of the proposed instruction generation and tool-augmented iterative trajectory refinement module on the NYC dataset. Specifically, for the RTE and KGC task, we validate the effectiveness of each block by comparing the following variants: (1) UrbanKGent-7B$^{\spadesuit}$ removes knowledgeable instruction template in RTE and KGC task; (2) UKGent* removes multi-view design in RTE task; (3) UrbanKGent-7B‡ removes external geospatial tool invocation block; (4)

Table 10: Effect of different blocks.

Models	GPT (acc/confidence)		Human (acc)	
	RTE	KGC	RTE	KGC
UrbanKGent-7B$^\spadesuit$	0.38/4.17	0.42/3.98	0.37	0.34
UrbanKGent-7B*	0.34/4.06	0.45/4.02	0.34	0.39
UrbanKGent-7B‡	0.45/4.32	0.40/3.97	0.45	0.23
UrbanKGent-7B†	0.44/4.10	0.47/3.85	0.46	0.43

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	√	√	√	√

UrbanKGent-7B† removes iterative trajectory self-refinement. We summarize the results in Table 10, and obtain the following observations.

First, knowledgeable instruction generation contributes to the overall performance of both RTE and KGC tasks. We observe a performance degradation by removing the knowledgeable instruction template. Second, the multi-view instruction design provides the most performance gain, which matches our intuition that the UrbanKG text contains heterogeneous relationships that can be effectively extracted by multi-view prompting design. Third, the tool invocation is very important for the KGC task, as we can observe significant performance degradation after removing the tool invocation. In addition, the iterative trajectory self-refinement brings consistent performance gain for both the RTE and KGC tasks.

E.3 RQ3: Complexity and Effiency Analysis

We make a comparison with the four paradigms to demonstrate the advantages of the constructed agent, which is shown in Table 11. Compared with Zero-shot reasoning (ZSL), In-context Learning (ICL), Vanilla Fine-tuning (VFT), and UrbanKGent Inference, UrbanKGent can incorporate extra urban knowledge, invoke external tools and iteratively self-refine to help better complete UrbanKGC tasks.

Moreover, we also provide comprehensive efficiency analysis to show the latency and cost of different models when completing UrbanKGC tasks. Specifically, we report the total inference time and cost[9] of each method completing with 1,000 RTE and KGC tasks. For the cost of GPT-4 service, we first count the number of prompt token and completion token spent on 1,000 RTE and KGC tasks, and then calculate the cost based on billing standards. As for the cost of UrbanKGent-13B, we first count the GPU running time spent on 1,000 RTE and KGC tasks, and then calculate the cost based on the A800 charging standard.

In addition, as reported in table 12, we also provide detailed inference latency of UrbanKGent family when deriving them for constructing different-scale of UrbanKGs.

E.4 RQ4: Case Study

As shown in Figure 8, we present the detailed reasoning process of constructed UrbanKGent-13B when finishing urban relational triplet extraction and knowledge graph completion task. Since the

[9]We subscribe to the NVIDIA A800 computing resources and GPT service from HKUST(GZ). Following the standard price instruction, we could calculate cost of GPT-based baselines or Llama-based baselines.

Table 12: The inference latency comparison of UrbanKGC using UrbanKGent family. We use two middle-size dataset (i.e., NYC and CHI) and two large-scale dataset (i.e., NYC-Large and CHI-Large) for UrbanKG construction.

Dataset	Latency (minutes)			Data Volume
	UrbanKGent-7B	UrbanKGent-8B	UrbanKGent-13B	
NYC	1.19	1.58	3.19	2,089
CHI	0.62	1.13	1.68	1,102
NYC-Large	23.07	30.76	61.93	40,480
CHI-Large	16.55	21.93	44.47	28,868
-	0.57	0.76	1.53	Per 1,000 records

Table 13: The statistic of entity and relation ontology of constructed UrbanKGs on NYC-Large and CHI-Large datset.

UrbanKG Dataset	# Coarse-grained Entity Ontology	# Fine-grained Entity Ontology	# Coarse-grained Relation Ontology	# Fine-grained Relation Ontology	# Entity	# Triplet
NYC-Large	4	6,281	4	2,138	228,928	905,442
CHI-Large	4	2,559	4	1,336	95,813	563,290

iterative self-refinement process contains excessive text, we display the reasoning process of only one iteration.

E.5 RQ5: Agent Application

We have released the UrbanKGent family consisting of 7B, 8B and 13B version in the Huggingface. The opensourced UrbanKGent family offer urban knowledge graph construction service for the researcher in this field. We provide application example in New York and Chicago. Specifically, following the UrbanKGent Inference framework, we sequentially derive UrbanKGent-13B for relational triplet extraction and knowledge graph completion. The obtained initial UrbanKGs encodes diverse urban spatial, temporal and functional knowledge. Then, we propose to use a two-stage triplet filtering and relation merging operation to further improve the quality of constructed UrbanKGs.

In the first stage, low-frequency relations (occur 5 times or less) are merged into high-frequency relations based on relation similarity threshold. The remaining low-frequency triples, whose similarity with any high-frequency relation is below the threshold will be filtered out. In the stage two, we first perform relation clustering based on the embedding of relations. Then, Within each cluster, we prompt LLM to identify semantically similar relations that can be merged into a single relation category, resulting the final set of merged relations.

As shown in Table 13, both urban entity and relation can be pre-categorized into 4 coarse-grained ontologies: spatial, temporal, functional, and others. The entity percentage of spatial, temporal, functional and others in NYC-Large is (68.34%, 17.66%, 10.39%, 3.61%), and in CHI-Large is (63.29%, 16.68%, 12.09%, 7.94%). The relation percentage of spatial, temporal, functional and others in NYC-Large and CHI-Large are (57.36%, 16.38%, 20.89%, 5.37%) and (60.57%, 15.11%, 21.67% and 2.65%), respectively. After multi-view entity recognition and relation extraction (shown in Figure 4(a) in our paper), the fine-grained entity ontologies (6,281 and 2,559 entity types of NYC-Large and CHI-Large UrbanKGs, respectively) and fine-grained relation ontologies (2,138 and 1,366 relation types of NYC-Large and CHI-Large UrbanKGs, respectively) are obtained.

F Limitation and Future Work

This work has limitation on the further application demonstration of construction UrbanKGs, although proposed UrbanKGent family could construct a UrbanKG with a hundreds relationship using only one-fifth of data. In addition, the evaluation method in this work is cost-intensive although GPT evaluation and Human evaluation has been experimentally demonstrated to be consistent. Despite the above limitations, we hope the opensource UrbanKGC agent can foster more extensive UrbanKG research and broad smart city application. In the future, we will derive extra image-modality data to further enrich UrbanKGC.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: In the end of abstract and introduction, the main claims are made.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The limitations are discussed in Appendix F

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification: This paper focus on domain application, no theory assumptions or proofs provided in this paper.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Researcher could follow the provided code to reproduce experimental results.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer:[Yes]

Justification: The code and data are in the supplementary material.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: All training details could be found in the code of the supplementary material.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Necessary statistical indicator of the results are provided in Section 8.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: In the experimental setting section 5.1, detailed computing resources explanation is provided.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: This paper follows the NeurIPS Code Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: In the end of introduction and agent application section 5.3, we discuss how can our proposed LLM agent to be deployed.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The data and models used in this paper are opensourced, there is no critical issue in model and data.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The researcher can follow CC-BY 4.0 license to use the data and code in this paper.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We provide illustrative details for the new dataset construction in Section 2.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [Yes]

    Justification: The human evaluation details could be found in Appendix D

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: [NA]

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.