Towards Zero-shot Question Answering in CPS-IoT: Large Language Models and Knowledge Graphs

Ozan Baris Mulayim*
omulayim@andrew.cmu.edu
Carnegie Mellon University / Lawrence Berkeley National Laboratory
Pittsburgh, PA, United States

Gabe Fierro
gtfierro@mines.edu
Colorado School of Mines
Golden, CO, United States

Mario Bergés†
marioberges@cmu.edu
Carnegie Mellon University
Pittsburgh, PA, United States

Marco Pritoni
mpritoni@lbl.gov
Lawrence Berkeley National Laboratory
Berkeley, CA, United States

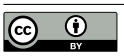
ABSTRACT

Natural language provides an intuitive interface for querying data, yet its unstructured nature often makes precise retrieval of information challenging. Knowledge graphs (KGs), with their structured and relational representations, offer a powerful solution to structuring knowledge, while large language models (LLMs) are capable of interpreting user intent through language. This combination of KGs and LLMs has been explored extensively for Knowledge Graph Question Answering (KGQA), primarily for open-domain or encyclopedic knowledge. Domain-specific KGQA, instead, presents significant opportunities for Cyber-Physical Systems (CPS) and the Internet of Things (IoT), where the extraction of structured metadata is essential for automation and scalability of control and analytics applications.

In this work, we evaluate and improve AutoKGQA, a domain-independent KGQA framework that utilizes LLMs to generate structured queries. Through a case study on KGs of sensor data from buildings, we assess its ability to retrieve time series identifiers, which are a requirement for extracting time series data from large sensory databases. Our results demonstrate that while AutoKGQA performs well in certain cases, its domain-agnostic approach leads to systematic failures particularly in complex queries requiring implicit knowledge. We show that domain-specific prompting significantly enhances query accuracy, allowing even smaller LLMs to perform on par with larger ones. These findings highlight the impact of domain-adapted prompting in KGQA (DA-KGQA) and suggest a path toward more efficient, scalable, and interpretable AI-driven metadata retrieval for CPS-IoT applications.

*Corresponding Author

†Mario Bergés holds concurrent appointments as a Professor of Civil and Environmental Engineering at Carnegie Mellon University and as an Amazon Scholar. This paper describes work at Carnegie Mellon University and is not associated with Amazon.



FMSys '25, May 6–9, 2025, Irvine, CA, USA

ACM ISBN /2025/05.
https://doi.org/10.1145/3722565.3727197

CCS CONCEPTS

• **Information systems → Language models**; **Question answering**; • **Computing methodologies → Knowledge representation and reasoning**.

KEYWORDS

Large Language Models, Knowledge Graphs, Time series Extraction

ACM Reference Format:
Ozan Baris Mulayim, Gabe Fierro, Mario Bergés, and Marco Pritoni. 2025. Towards Zero-shot Question Answering in CPS-IoT: Large Language Models and Knowledge Graphs. In *The 2nd International Workshop on Foundation Models for Cyber-Physical Systems Internet of Things (FMSys '25), May 6–9, 2025, Irvine, CA, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3722565.3727197

1 INTRODUCTION

Foundation Models (FMs) aim to demonstrate versatility and generalization capabilities across a wide range of tasks, leveraging self-supervised learning to adapt to diverse domains with minimal fine-tuning [6]. However, in the context of Cyber-Physical Systems and the Internet of Things (CPS-IoT), FMs often assume that time-series data will be readily available in labeled and structured forms, limiting their ability to handle real-world scenarios. We argue that CPS-IoT FMs should instead be capable of performing critical operations such as time-series retrieval, processing, planning, and execution to achieve true generalization.

For CPS-IoT, planning has been explored through methods such as Time Series Foundation Models (TSFMs) [22], and execution has been demonstrated in robotics FMs [16]. However, information retrieval remains an understudied yet essential component. This capability is crucial for enabling models to autonomously navigate sensor databases, identify relevant *contextual* and *sensor* data sources. By bridging the gap between high-level task descriptions and meaningful insights, information retrieval allows FMs to act as dynamic intermediaries between user/agent intent and complex, unstructured data repositories.

The integration of FMs with structured data such as knowledge graphs (KGs) holds significant promise, enabling general-purpose models to achieve domain-specific reasoning and contextual awareness [31]. This potential has motivated efforts to develop Graph

FMs [31], Large Language Model (LLM)-augmented KGs (e.g., KG Question Answering (KGQA) [3]), KG-enhanced LLMs, and hybrid LLM-KG systems [25]. These integrations have shown promise in applications like recommendation systems, knowledge discovery, and search engines [14, 23, 33]. However, their potential in the CPS-IoT domain, particularly for contextual and sensor information retrieval, has not been sufficiently explored. Bridging this gap is critical for enabling FMs to autonomously navigate complex data landscapes, retrieve relevant sensor inputs, and generate actionable insights for real-world applications.

Buildings serve as an ideal case study of CPS-IoT systems, as they generate vast amounts of time-series data by monitoring the operation of thousands of sensors and actuators with various functions. Moreover, the use of KGs to represent relationships between different building systems and components has been extensively explored [27]. Yet, the integration of FMs with KGs for building-related applications remains underexplored [24], with limited empirical analysis beyond preliminary studies [5].

To address this gap, we systematically evaluate a domain-independent, zero-shot KGQA tool—AutoKGQA [3]—both in its original form and with domain-specific modifications. Our contributions include: (1) constructing a preliminary evaluation set of building-related questions with varying levels of complexity and ambiguity, (2) comparing the performance of AutoKGQA before and after introducing domain-aware prompting, and (3) analyzing the accuracy-cost tradeoffs of using different LLMs for KGQA. Our findings demonstrate that minimal prompt engineering can significantly enhance query accuracy while reducing reliance on larger, more expensive models, paving the way for scalable and efficient KGQA applications in smart buildings and beyond. Furthermore, through concrete examples, we highlight scenarios where domain-specific knowledge is indispensable and where the general knowledge contained in LLMs falls short, providing insights into the limitations and opportunities for future improvements in LLM-KG integration.

1.1 Background

Several researchers explored the interaction between KG and FM [8, 21, 26, 30]. For instance, KG reasoning focuses on deriving new knowledge through logical inference, with notable works including [12, 32]. KG completion aims to predict missing triples or entities, enhancing the structure and completeness of KGs [37]. In our work, we focus on the ability of FM to extract sensor and contextual requirements provided in natural language and retrieve relevant answers from an existing building KG, a problem that closely aligns with KGQA. While text-to-SPARQL approaches have tackled similar challenges, we concentrate on KGQA, as many of its implementations inherently include a text-to-SPARQL subtask.

To establish the necessary background, we first introduce the *Brick Schema*, the ontology used to construct the building KG we used in our study. We then review existing KGQA methods, evaluating their applicability and limitations in CPS-IoT contexts.

1.1.1 Brick Schema. Brick Schema is an open-source ontology standardizing metadata representation in commercial buildings. It defines a structured vocabulary for building components, such as sensors, HVAC systems, and zones, along with their interconnections, enabling interoperability and reasoning over structured data. Its adoption is supported by a strong research community [4], industry implementation [10], and alignment with ASHRAE 223 standards [2]. A key distinction must be made between KGs and ontologies. A KG encodes specific knowledge by representing entities, relationships, and properties within a domain [15], while an ontology defines the formal structure of that domain, specifying entity types, hierarchies, and constraints to ensure consistency and enable inference. In the buildings domain, KGs are referred as semantic models or building models (i.e., *Brick model*) and they describe metadata of a specific building. On the other hand *ontologies* such as *Brick schema* are ontologies that standardizes classes, relationships, and constraints, improving consistency between KGs.

1.1.2 Knowledge Graph Question Answering. KGQA methods fall into two categories: retrieval-based and semantic parsing-based approaches [7]. Retrieval-based methods extract relevant entities efficiently but struggle with complex, multi-hop reasoning [35]. Semantic parsing methods translate natural language into SPARQL, enabling precise retrieval but requiring costly annotations and domain-specific adaptation. Given the complexity of CPS-IoT queries, retrieval-based methods can be insufficient, while semantic parsing offers a more suitable framework for structured KG queries.

Recent approaches for semantic parsing integrate LLMs for reasoning-driven SPARQL generation, aiming to reduce the training requirements on KG-specific question-answer pairs. SGPT learns patterns for query generation but still requires question-answer pairs [29], making it impractical for ontologies like Brick. SPARQLGEN introduced a one-shot SPARQL framework with GPT-3, incorporating contextual information to improve query quality [17], but its reliance on ground truth queries and dataset-specific examples limits generalization. Other methods, such as structured extraction using a BERT-CRF pipeline, improve disambiguation but would generate excessive number of entities in building KGs and require fine-tuning using a training dataset [35]. To address these challenges, AutoKGQA [3] enables zero-shot KGQA by selecting relevant ontology terms and instances as structured context for LLMs. While other zero-shot KGQA tools like Tree-KGQA [28] exist, they do not support arbitrary KGs. HybGRAG [19] is a recent related work, focusing on hybrid questions, but its implementation was unavailable at the time of this submission.

2 METHODOLOGY

This section outlines our methodology for evaluating KGQA systems in building automation. We describe the evaluation set, which categorizes queries by complexity and ambiguity, and details of how AutoKGQA was extended with domain-specific prompting.

2.1 Evaluation Set

In KGQA, ambiguity in entity and relation references presents a major challenge [34]. Question formulation directly impacts retrieval accuracy, particularly when references vary in specificity [18]. We categorize questions based on two factors: (1) *complexity* of queries using the number of hops[1] required, and (2) *ambiguity* of the entities and relations. Each query level (shown as L) is denoted as

[1]We consider a query to be *zero-hop* if it only involves type-based filtering using `rdf:type` and `rdfs:subClassOf*`, without traversing relationships between entities.

$m.n$, where m represents the number of hops[2], and n represents the ambiguity, increasing as the query elements become less explicit.

The questions generated using this approach can be seen in Table 1. They were generated based on the queries required to build the applications available in Mortar [11]. Explicit and partially explicit class mappings are highlighted in green, while explicit relationships are shown in blue.

Linguistic *ambiguity* exists along a continuum, and our objective here is to extract representative samples from its underlying distribution. Some queries explicitly specify an entity or relation (e.g., building electrical meter), while others require contextual interpretation (e.g., energy consumption) to map them to the corresponding KG term (e.g., `Building_Electrical_Meter`). Further, controlled natural language (SQUALL [9]) queries ($L = 0.0$) serve as the starting point, due to being less natural but closer to the true SPARQL query [20]. In addition to ambiguity in classes and relationships, we distinguish question formulations: $L\{0.0, 0.1, 0.3, 0.5\}$ include direct questions that explicitly ask for an entity or relation (e.g., *what is*), while $L\{0.2, 0.4\}$ include indirect requests that imply a request for information without directly stating it (e.g., *retrieve*, *list*, *show*). As query *complexity* increases, retrieval requires traversing multiple relations, introducing additional ambiguity such as missing links[3] ($L = 2.2$). Zero-hop queries involve type-based filtering without explicitly traversing relationships between entities ($L\{0.0 - 0.5\}$), while multi-hop queries ($L\{1.0 - 2.2\}$) require reasoning over intermediate entities and relations, making the task more challenging.

It is important to clarify that this evaluation set is not designed to be exhaustive, as constructing a fully comprehensive dataset would be prohibitively expensive [20]. Prior work has addressed this challenge by using semi-automated methods to generate large datasets, given the high cost and effort of manual curation [20]. In our case, we intentionally keep the dataset limited, ensuring that each level introduces increasing complexity without redundancy. This allows for a more focused analysis of failure cases. In addition, compared to the traditional KGQA methods, dataset size naturally decreases in a zero-shot setting because (1) only test data are needed, significantly reducing data requirements, and (2) testing with LLMs incurs substantial computational and monetary costs, making smaller evaluation sets more practical. For example, authors of AutoKGQA used 15 queries for validation [3]. To define our test set, we iteratively develop cases for each level of complexity and ambiguity until all tested methods fail, revealing the limitations of current approaches. While this provides valuable insights, larger test sets are necessary for more conclusive evaluations of KGQA performance in this domain.

2.2 Experimental Setting

AutoKGQA was selected as the primary baseline for its zero-shot KGQA capability. We adapt its prompt with domain-specific information and incorporate a tree-of-thought approach [36], resulting in our Domain Adapted AutoKGQA (DA-KGQA). This strategy improves performance through prompt engineering while preserving the fully zero-shot nature of the task. Unlike AutoKGQA, which samples five completions from an LLM using a single example, DA-KGQA prompts the model to simulate multiple expert agents, each proposing a SPARQL query using a different strategy and level of complexity (see prompt in Figure 1, right). This structured diversity encourages more varied and progressively complex reasoning paths, increasing the likelihood of producing a correct and precise query. The distinction between the two approaches is illustrated in Figure 1. Our code is available to enable reproducibility[4] .

It is important to clarify that KGQA retrieves *point names*, not the time series data itself. The KG embeds point names, which a client program can use to fetch data from a building management system—following the approach of systems like Energon [13] and Mortar [11]. This design differs from traditional KGQA assumptions, where the graph contains all necessary information to answer a query. Instead, the generated query retrieves point names, leaving data retrieval and computation to external systems which take these point names as input. For our experiment, we use the KG of a real-world building (vm3a) from Mortar repository [11]. This KG, constructed with the Brick Schema, contained 1,609 ontology terms and 61,366 instances. The KG also represents hierarchical equipment structures, including AHUs, Variable Air Volume (VAV) systems, and sensors, where sensors can be associated with multiple levels of hierarchy. Since the model was developed with a previous version of the ontology, we updated it to align with the latest Brick Schema version (1.4.0).

To ensure consistency, we set the temperature parameter to zero for all LLM calls, recognizing that LLMs remain inherently stochastic. Nevertheless, our initial tests showed consistent results across multiple runs, likely due to the five-query generation step, which tends to converge on a consistent approach. Since AutoKGQA performs sampling in this step, it retained a non-zero temperature of 0.5 to allow variability. Other parameters, such as relevance threshold (0.25) and max hits rate (0.005), were determined through trial and error to balance retrieval precision and token constraints. All other parameters remained identical to the original AutoKGQA implementation.

3 RESULTS

Table 1 compares the performance of our approach against AutoKGQA. To assess the impact of different sensor types, we generated two sets of zero-hop questions (e.g., damper position versus electrical meter) and found no significant differences in performance. We then proceeded to more complex 1-hop and 2-hop queries. The LLM models used—Llama 3.3 70B, GPT-4o, and GPT-4o-mini—are denoted as `L70b`, `4o`, and `4o-mini`, respectively.

To evaluate performance, we execute the generated SPARQL queries and compare their outputs against ground truth queries. Our primary metric is based on whether the result sets are semantically equivalent. When a query returns more than 10 rows, we compare the number of returned rows as a proxy for correctness. For smaller result sets (fewer than 10 rows), we manually analyze whether the returned point names match those of the ground truth. Additionally, we assess whether the output includes all required variable bindings. For instance, if a question asks for both zone air temperature sensors

[2]While `rdfs:subClassOf*` may span multiple edges, within the class hierarchy, we treat subclass reasoning as background inference rather than part of query traversal.
[3]We define a *missing link* as a condition in which a question appears to imply a direct (one-hop) relationship between two entities, but the graph requires traversal through an intermediate node (two-hop) to retrieve the answer.
[4]https://github.com/ozanbarism/DA-KGQA

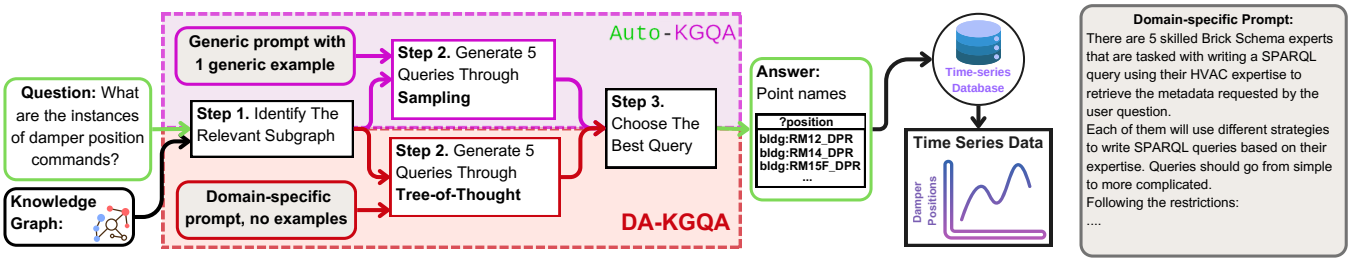


Figure 1: Comparison of AutoKGQA and DA-KGQA with its' domain-specific prompt.

Table 1: Performance Across Models and LLMs for Zero-Shot Question Answering

Q_{id}	L	Question	AutoKGQA			DA-KGQA		
			4o-mini	4o	L70b	4o-mini	4o	L70b
1	0.0	Which <meter> is a <Building Electrical Meter>?	✓	✓	✓	✓	✓	✓
2	0.1	What are the instances of building electrical meter?	✓	✓	✓	✓	✓	✓
3	0.2	List all building electrical meters	✓	✓	✓	✓	✓	✓
4	0.3	Which electrical meters are installed in this building?	×	✓	×	✓	✓	✓
5	0.4	Show the electrical meters available in this building.	×	✓	✓	✓	✓	✓
6	0.5	What devices are responsible for measuring energy use in this building?	×	×	×	×	×	×
7	0.0	Which <position> is a <Damper Position Command>?	✓	✓	✓	✓	✓	✓
8	0.1	What are the instances of damper position commands?	✓	✓	✓	✓	✓	✓
9	0.2	Retrieve all damper position commands	✓	✓	✓	✓	✓	✓
10	0.3	What commands adjust the position of dampers?	×	✓	×	✓	✓	✓
11	0.4	List the commands used for damper positioning in this system.	×	✓	✓	✓	✓	✓
12	0.5	What determines how airflows are regulated?	×	×	×	×	×	×
13	1.0	What are the instances of air flow setpoint and the equipment they are a point of?	✓	✓	✓	✓	✓	✓
14	1.1	What are the instances of air flow setpoint and their associated equipment?	✓	×[1]	×[1]	✓	✓	×[1]
15	1.2	What air flow setpoints exist in this system, and what equipment are they related to?	✓	×[1]	×[1]	✓	✓	✓
16	1.3	What are the instances of setpoints of air flow and the equipment they are a point of?	×[1]	✓	×[1]	✓	✓	✓
17	2.0	What are the instances of zone air temperature sensors that are points of VAVs that are fed by AHUs?	×[1]	×[1]	✓	×[2]	×[2]	✓
18	2.1	Which zone air temperature sensors receive input from terminal units connected to AHUs?	×[3]	×[1]	×[4]	×[5]	×[5]	✓[6]
19	2.2	What are the existing zone air temperature sensors and their associated AHUs?	×[7]	×[7]	×	×	×	✓[6]

[1] Asks for the equipment label instead of the instances, [2] Not returning the AHU instances, [3] Using `brick:Terminal_Unit`, [4] Using `brick:Terminal`, [5] Hallucinates relationships such as `brick:hasInput`, `brick:isInputFrom` [6] Does not explicitly assign a class to the intermediary so the query still returns, [7] Missing the link between zone air temperature sensors and AHUs.

and their associated AHUs (e.g., $Q_{id} = 15$), a correct answer must return point names for both variables. However, if the resulting answer has more information (e.g., also returning the VAVs that connect the zone air temperature sensors to the AHUs), we still accept it to be correct as long as number of rows still match those of the ground truth query.

For simpler questions $L\{0.0-0.5\}$, our approach performs identically to AutoKGQA when using larger models, such as `4o`. However, for smaller models like `L70b` and `4o-mini`, our approach consistently outperforms AutoKGQA across both applications (i.e., meters and dampers). This indicates that performance is not dependent on the specific class being queried. Additionally, within the zero-hop range ($L\{0.0-0.5\}$), model performance remained consistent across different surface forms of questions, whether phrased directly $L\{0.0.0.1, 0.3, 0.5\}$ (e.g., *What is...*, *Which...*) or indirectly $L\{0.2.0.4\}$ (e.g., *List*, *Retrieve*, *Show*).

Notably, the clear performance improvement with `4o-mini` suggests that minimal prompt engineering can enable smaller models to achieve results comparable to much larger models, highlighting potential for efficient deployment in edge computing environments. At $L = 0.5$, both methods struggle, which is expected given the elevated ambiguity of these questions. Answering such questions

requires a combination of two key abilities: (1) strong domain-specific reasoning and (2) the capacity to traverse the graph to infer what entities the user might be referring to. This challenge is significant even for human experts unfamiliar with the KG. For instance, when asked about energy consumption, a model must determine whether the question pertains to electricity, natural gas, or another energy source, which depends on the specific systems installed in the building, which was electrical meter in this case. We also find that partially explicit class mappings (e.g., commands used for damper positioning) have minimal impact on performance for larger models like `4o` and `L70b`. Specifically, in DA-KGQA, such mappings did not affect performance at all, even for smaller models. This suggests that users can express class requirements with *some degree of flexibility* without negatively impacting accuracy.

As we move to more complex queries $L\{1.0-1.3\}$, an interesting pattern emerges. Models are generally able to construct more complex queries when classes and relationships are clearly defined (i.e., explicit). However, AutoKGQA exhibits a notable failure mode: it frequently attempts to return equipment labels (i.e., rdf:label values that provide a human-readable description of an instance if exists) instead of their instances. Interestingly, `4o-mini` outperforms its larger counterparts in these cases, likely because it follows a simpler reasoning process and avoids unnecessary query modifications (e.g., trying to extract labels) which results in raising errors when such labels do not exist.

For the most complex two-hop queries ($L\{2.0-2.2\}$), we observe interesting failure patterns. The original AutoKGQA implementation, while generating syntactically correct SPARQL queries, consistently requests the labels of AHUs and VAVs rather than their instances. This leads to failures in more explicit scenarios, with only `L70b` successfully returning the correct answer for the most explicit query ($Q_{id} = 17$). With DA-KGQA, we observe a shift in the failure modes, particularly in `4o-mini` and `4o`. These models correctly return all zone air temperature sensors but fail to retrieve the corresponding AHU instances using the equipment hierarchy. However, this omission can be arguably justified, as the question does not explicitly state that AHUs should be included. As we move to $L = 2.1$, hallucinations become a significant challenge. Our approach occasionally introduces incorrect relationships in $Q_{id} = 18$ (e.g., `brick:hasInput`), whereas AutoKGQA instead defaults to using Brick classes such as `brick:Terminal_Unit` and `brick:Terminal`. The intent behind $Q_{id} = 18$ ($L = 2.1$) was to test whether models could recognize that "terminal unit" is a generic reference when the user is uncertain about the exact equipment type (i.e., VAV). Among all models, only `L70b` in our framework correctly inferred this ambiguity, assigning a variable without imposing a class constraint such as `brick:VAV` or `brick:Terminal_Unit`. The most challenging query, $L = 2.2$, required inferring a missing link—a case where the question appears to imply a single-hop relationship, but a domain expert would recognize the need for an intermediate entity (e.g., a VAV). Notably, `L70b` in our framework successfully generated the correct query structure by accurately inferring the missing link.

For the full set of 19 queries, the cost of running `4o` with AutoKGQA was $1.920, while DA-KGQA lowered this to $1.403, representing a 27% cost reduction. Similarly, for `4o-mini`, costs dropped from $0.091 to $0.047, a 48% reduction. More strikingly, our results show that `4o-mini`, when paired with domain-aware prompting, can achieve comparable and even superior performance to `4o` at 41 times lower cost, highlighting the potential of simple yet effective prompt engineering to make KGQA more cost-efficient and scalable. Although `4o-mini` is a commercial model and cannot currently be deployed locally, open-source alternatives such as LLaMA 3.1 8B offer a viable path for running similar models on-device. Based on recent estimates [1], `4o-mini` has approximately 8 billion parameters—placing it at the upper end of what edge computing with high-end hardware accelerators (e.g., Jetson Orin, Apple M-series NPUs, or desktop GPUs with quantized inference) can reasonably support. Future work should evaluate even smaller models, such as LLaMA 3.2 3B, to further explore the performance-cost trade-offs and practical feasibility of deploying DA-KGQA in resource-constrained environments.

Overall, DA-KGQA demonstrates significant performance gains, particularly with large models such as `L70b` performing well even on complex queries that require reasoning over implicit relationships, and smaller models such as `4o-mini` matching and outperforming larger models at a significantly lower cost.

4 LIMITATIONS, CONCLUSIONS AND FUTURE WORK

Our findings show that simple domain-aware prompt engineering significantly improves KGQA, allowing smaller models to match or surpass larger ones. This enhancement supports cost-efficient, privacy-preserving deployment in edge computing. However, challenges remain, particularly in handling ambiguous queries. Inferring intent requires KG traversal and dynamic adjustments based on context. While structured methods help, some cases still require human clarification. Queries with explicit or partially explicit class references perform well, reinforcing KGQA's potential for AI-assisted tools and autonomous agents. These findings highlight the importance of domain knowledge in KGQA and motivate further research into zero-shot foundation models for CPS-IoT applications. More broadly, this work contributes to the long-term vision of autonomous agents capable of navigating large sensory databases, extracting time series data, and applying it effectively in downstream CPS-IoT tasks.

Nonetheless, several limitations remain. Our evaluation was conducted on a single building and limited to the Brick ontology, which constrains the generalizability of our findings. Furthermore, we focused exclusively on AutoKGQA without comparing against other state-of-the-art KGQA systems that may exhibit different performance characteristics under domain-specific prompting. The relatively small size of the evaluation set also limited our ability to conduct detailed error analysis.

These limitations suggest multiple directions for future work. Expanding the evaluation to cover multiple buildings and ontologies would provide stronger evidence of generalizability across settings. Incorporating additional KGQA baselines would allow for a more comprehensive performance comparison. Lastly, increasing the scale and diversity of the evaluation set would support a deeper analysis of failure modes, particularly in cases where prompt-based strategies fall short of resolving query ambiguity.

ACKNOWLEDGMENTS

The authors would also like to acknowledge the support from the Assistant Secretary for Energy Efficiency and Renewable Energy, Building Technologies Office, of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231, by the New York State Energy Research & Development Authority (NYSERDA) through the NextGen HVAC Innovation Challenge program, by California Energy Commission through grant EPC-19-013, and the National Alliance for Water Innovation (NAWI), funded by the U.S. Department of Energy, Office of Energy Efficiency and Renewable Energy (EERE), Industrial Efficiency and Decarbonization Office, under Funding Opportunity Announcement DE-FOA-0001905.

REFERENCES

[1] Asma Ben Abacha, Wen-wai Yim, Yujuan Fu, Zhaoyi Sun, Meliha Yetisgen, Fei Xia, and Thomas Lin. 2024. Medec: A benchmark for medical error detection and correction in clinical notes. *arXiv preprint arXiv:2412.19260* (2024).

[2] "ASHRAE". 2022. Introduction to Open223. https://docs.open223.info/intro.html Accessed: 2024-02-27.

[3] Caio Viktor S. Avila, Vânia M.P. Vidal, Wellington Franco, and Marco A. Casanova. 2024. Experiments with text-to-SPARQL based on ChatGPT. In *2024 IEEE 18th International Conference on Semantic Computing (ICSC)*. IEEE, Laguna Hills, CA, USA, 277–284. https://doi.org/10.1109/ICSC59802.2024.00050

[4] Bharathan Balaji, Arka Bhattacharya, Gabriel Fierro, Jingkun Gao, Joshua Gluck, Dezhi Hong, Aslak Johansen, Jason Koh, Joern Ploennigs, Yuvraj Agarwal, et al. 2016. Brick: Towards a unified metadata schema for buildings. In *Proceedings of the 3rd ACM International Conference on Systems for Energy-Efficient Built Environments*. 41–50.

[5] Ozan Baris, Yizhuo Chen, Gaofeng Dong, Liying Han, Tomoyoshi Kimura, Pengrui Quan, Ruijie Wang, Tianchen Wang, Tarek Abdelzaher, Mario Bergés, et al. 2025. Foundation Models for CPS-IoT: Opportunities and Challenges. *arXiv preprint arXiv:2501.16368* (2025).

[6] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. 2021. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258* (2021).

[7] Abir Chakraborty. 2024. Multi-hop Question Answering over Knowledge Graphs using Large Language Models. http://arxiv.org/abs/2404.19234 arXiv:2404.19234 [cs].

[8] Huajun Chen. 2023. Large knowledge model: Perspectives and challenges. *arXiv preprint arXiv:2312.02706* (2023).

[9] Sébastien Ferré. 2014. SQUALL: The expressiveness of SPARQL 1.1 made available as a controlled natural language. *Data & Knowledge Engineering* 94 (2014), 163–188.

[10] Gabe. Fierro and Pieter. Pauwels. 2022. Survey of Metadata Schemas for Data-driven Smart Buildings (Annex 81).

[11] Gabe Fierro, Marco Pritoni, Moustafa Abdelbaky, Daniel Lengyel, John Leyden, Anand Prakash, Pranav Gupta, Paul Raftery, Therese Peffer, Greg Thomson, and David E. Culler. 2019. Mortar: An Open Testbed for Portable Building Analytics. *ACM Transactions on Sensor Networks* 16, 1 (Dec. 2019), 7:1–7:31. https://doi.org/10.1145/3366375

[12] Mikhail Galkin, Xinyu Yuan, Hesham Mostafa, Jian Tang, and Zhaocheng Zhu. 2023. Towards foundation models for knowledge graph reasoning. *arXiv preprint arXiv:2310.04562* (2023).

[13] Fang He, Cheng Xu, Yanhui Xu, Dezhi Hong, and Dan Wang. 2020. EnergonQL: A Building Independent Acquisitional Query Language for Portable Building Analytics. In *Proceedings of the 7th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*. ACM, Virtual Event Japan, 266–269. https://doi.org/10.1145/3408308.3427979

[14] Chengkai Huang, Tong Yu, Kaige Xie, Shuai Zhang, Lina Yao, and Julian McAuley. 2024. Foundation models for recommender systems: A survey and new perspectives. *arXiv preprint arXiv:2402.11143* (2024).

[15] Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and S Yu Philip. 2021. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE transactions on neural networks and learning systems* 33, 2 (2021), 494–514.

[16] Kento Kawaharazuka, Tatsuya Matsushima, Andrew Gambardella, Jiaxian Guo, Chris Paxton, and Andy Zeng. 2024. Real-world robot applications of foundation models: A review. *Advanced Robotics* 38, 18 (2024), 1232–1254.

[17] Liubov Kovriguina, Roman Teucher, Daniil Radyush, Dmitry Mouromtsev, N Keshan, S Neumaier, AL Gentile, and S Vahdati. 2023. SPARQLGEN: One-Shot Prompt-based Approach for SPARQL Query Generation.. In *SEMANTiCS (Posters & Demos)*.

[18] Adila Alfa Krisnadhi, Mohammad Yani, and Indra Budi. 2024. Entity and Relation Linking for Knowledge Graph Question Answering Using Gradual Searching. *Jurnal Nasional Teknik Elektro dan Teknologi Informasi* 13, 2 (2024), 139–146.

[19] Meng-Chieh Lee, Qi Zhu, Costas Mavromatis, Zhen Han, Soji Adeshina, Vassilis N Ioannidis, Huzefa Rangwala, and Christos Faloutsos. 2024. HybGRAG: Hybrid Retrieval-Augmented Generation on Textual and Relational Knowledge Bases. *arXiv preprint arXiv:2412.16311* (2024).

[20] Jens Lehmann, Preetam Gattogi, Dhananjay Bhandiwad, Sébastien Ferré, and Sahar Vahdati. 2023. Language models as controlled natural language semantic parsers for knowledge graph question answering. In *ECAI 2023*. IOS Press, 1348–1356.

[21] DaiFeng Li and Fan Xu. 2024. Synergizing knowledge graphs with large language models: a comprehensive review and future prospects. *arXiv preprint arXiv:2407.18470* (2024).

[22] Yuxuan Liang, Haomin Wen, Yuqi Nie, Yushan Jiang, Ming Jin, Dongjin Song, Shirui Pan, and Qingsong Wen. 2024. Foundation models for time series analysis: A tutorial and survey. In *Proceedings of the 30th ACM SIGKDD conference on knowledge discovery and data mining*. 6555–6565.

[23] Vanessa Lopez, Lam Hoang, Marcos Martinez-Galindo, Raúl Fernández-Díaz, Marco Luca Sbodio, Rodrigo Ordonez-Hurtado, Mykhaylo Zayats, Natasha Mulligan, and Joao Bettencourt-Silva. 2025. Enhancing foundation models for scientific discovery via multimodal knowledge graph representations. *Journal of Web Semantics* 84 (2025), 100845.

[24] Ozan Baris Mulayim, Lazlo Paul, Marco Pritoni, Anand Krishnan Prakash, Malavikha Sudarshan, and Gabe Fierro. 2024. Large Language Models for the Creation and Use of Semantic Ontologies in Buildings: Requirements and Challenges. In *Proceedings of the 11th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*. 312–317.

[25] Shirui Pan, Linhao Luo, Yufei Wang, Chen Chen, Jiapu Wang, and Xindong Wu. 2024. Unifying large language models and knowledge graphs: A roadmap. *IEEE Transactions on Knowledge and Data Engineering* (2024).

[26] Shirui Pan, Linhao Luo, Yufei Wang, Chen Chen, Jiapu Wang, and Xindong Wu. 2024. Unifying Large Language Models and Knowledge Graphs: A Roadmap. *IEEE Transactions on Knowledge and Data Engineering* 36, 7 (July 2024), 3580–3599. https://doi.org/10.1109/TKDE.2024.3352100 Conference Name: IEEE Transactions on Knowledge and Data Engineering.

[27] Marco Pritoni, Drew Paine, Gabriel Fierro, Cory Mosiman, Michael Poplawski, Avijit Saha, Joel Bender, and Jessica Granderson. 2021. Metadata schemas and ontologies for building energy applications: A critical review and use case analysis. *Energies* 14, 7 (2021), 2024.

[28] Md Rashad Al Hasan Rony, Debanjan Chaudhuri, Ricardo Usbeck, and Jens Lehmann. 2022. Tree-KGQA: an unsupervised approach for question answering over knowledge graphs. *IEEE Access* 10 (2022), 50467–50478.

[29] Md Rashad Al Hasan Rony, Uttam Kumar, Roman Teucher, Liubov Kovriguina, and Jens Lehmann. 2022. SGPT: A Generative Approach for SPARQL Query Generation From Natural Language Questions. *IEEE Access* 10 (2022), 70712–70723. https://doi.org/10.1109/ACCESS.2022.3188714

[30] Wenbo Shang and Xin Huang. 2024. A Survey of Large Language Models on Generative Graph Analytics: Query, Learning, and Applications. *arXiv preprint arXiv:2404.14809* (2024).

[31] Kai Wang and Siqiang Luo. 2024. Towards Graph Foundation Models: The Perspective of Zero-shot Reasoning on Knowledge Graphs. *arXiv preprint arXiv:2410.12609* (2024).

[32] Kai Wang, Yuwei Xu, Zhiyong Wu, and Siqiang Luo. 2024. LLM as Prompter: Low-resource Inductive Reasoning on Arbitrary Knowledge Graphs. *arXiv preprint arXiv:2402.11804* (2024).

[33] Haoyi Xiong, Jiang Bian, Yuchen Li, Xuhong Li, Mengnan Du, Shuaiqiang Wang, Dawei Yin, and Sumi Helal. 2024. When search engine services meet large language models: visions and challenges. *IEEE Transactions on Services Computing* (2024).

[34] Haobo Xiong, Shuting Wang, Mingrong Tang, Liping Wang, and Xuemin Lin. 2021. Knowledge graph question answering with semantic oriented fusion model. *Knowledge-Based Systems* 221 (2021).

[35] Shuangtao Yang, Mao Teng, Xiaozheng Dong, and Fu Bo. 2023. LLM-Based SPARQL Generation with Selected Schema from Large Scale Knowledge Base. In *Knowledge Graph and Semantic Computing: Knowledge Graph Empowers Artificial General Intelligence*, Haofen Wang, Xianpei Han, Ming Liu, Gong Cheng, Yongbin Liu, and Ningyu Zhang (Eds.). Springer Nature, Singapore, 304–316. https://doi.org/10.1007/978-981-99-7224-1_24

[36] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2024. Tree of thoughts: Deliberate problem solving with large language models. *Advances in Neural Information Processing Systems* 36 (2024).

[37] Yichi Zhang, Zhuo Chen, Lingbing Guo, Yajing Xu, Wen Zhang, and Huajun Chen. 2024. Making large language models perform better in knowledge graph completion. In *Proceedings of the 32nd ACM International Conference on Multimedia*. 233–242.