TESTING CHATGPT-AIDED SPARQL GENERATION FOR SEMANTIC CONSTRUCTION INFORMATION RETRIEVAL

Yuan Zheng & Prof.Olli Seppänen
Aalto University, Finland

Sebastian Seiß & Prof. Jürgen Melzner
Bauhaus-University Weimar, Germany

ABSTRACT: *Recently there has been a strong interest in using semantic technologies to improve information management in the construction domain. Ontologies provide a formalized domain knowledge representation that provides a structured information model to facilitate information management issues such as formalization and integration of construction workflow information and data and enables further applications such as information retrieval and reasoning. SPARQL Protocol And RDF Query Language (SPARQL) queries are the main approaches to conduct the information retrieval from the Resource Description Framework (RDF) format data. However, there is a barrier for end users to develop the SPARQL queries, as it requires proficient skills to code them. This challenge hinders the practical application of ontology-based approaches on construction sites. As a generative language model, ChatGPT has already illustrated its capability to process and generate human-like text, including the capability to generate the SPARQL for domain-specific tasks. However, there are no specific tests evaluating and assessing the SPARQL-generating capability of ChatGPT within the construction domain. Therefore, this paper focuses on exploring the usage of ChatGPT with a case of importing the Digital Construction Ontologies (DiCon) and generating SPARQL queries for specific construction workflow information retrieval. We evaluate the generated queries with metrics including syntactical correctness, plausible query structure, and coverage of correct answers.*

KEYWORDS: *Semantic web, Ontology, ChatGPT, SPARQL, RDF, Information retrieval, Construction*

1. INTRODUCTION

Construction is an information-intensive and dynamic industry, which requires effective information management and exchange. Especially, construction professionals need always to retrieve demanding information about the construction process to be aware of the prompt situation to support their decision-making and action-taking (Akinci, 2015). With the ongoing advancements of digital implementation in the construction domain, a large amount of semantic data can be collected from heterogeneous systems, which requires a systematic solution to formalize, integrate, and manage the data (Kosovac et al., 2000). Therefore, researchers in the construction domain have investigated the application of semantic web technologies to facilitate information formalization and interoperability issues (Zhou et al., 2016). For example, numerous ontologies have been developed in the construction domain, to provide a formalized construction domain knowledge representation and comprehensive semantic vocabulary. These ontologies support the conversion of construction information into Resource Description Framework (RDF) format and the establishment of an integrated semantic graph database.

Such integrated semantic graph database has been proven to the advantageous in the application of information integration, reasoning, and retrieval by academic scholars in the construction domain (Akinyemi et al., 2018). However, in terms of the practical implementation, one drawback of the graph database can be identified. To retrieve the demanding information from the graph database, the SPARQL Protocol and RDF Query Language (SPARQL) queries are needed. However, the construction sector has limited practitioners with sufficient knowledge of ontology and proficient skills to code the SPARQL queries at the moment. This makes it difficult for end-users to interact directly with the graph database and retrieve the particular construction information they need. While there have been various attempts to employ web-based lists or templates to assist users in crafting SPARQL queries without coding skills, generating SPARQL queries quickly and easily remains a challenge for end-users. Given this challenge, it would be beneficial to develop an intuitive and self-explanatory method that allows end-users to create SPARQL queries more easily.

ChatGPT is an AI-empowered language generation model, which uses the Transformer algorithm and large language model (LLM) principle as the basis to generate human-like text based on the given prompts and context

Referee List (DOI: 10.36253/fup_referee_list)
FUP Best Practice in Scholarly Publishing (DOI 10.36253/fup_best_practice)
Yuan Zheng, Olli Seppänen, Sebastian Seiß, Jürgen Melzner, *Testing ChatGPT-Aided SPARQL Generation for Semantic Construction Information Retrieval,* pp. 751-760, © 2023 Author(s), CC BY NC 4.0, DOI 10.36253/979-12-215-0289-3.75

with rapid response time (van Dis et al., 2023). After its public release at the end of 2022, ChatGPT received huge attention from academia, industries, and consumers. ChatGPT also can analyze and generate structured syntax-based contents such as codes, scripts, and ontology syntax (Lin et al., 2023). ChatGPT can also generate the SPARQL query sentences based on the predefined ontology inputs since its database involves numerous examples of SPARQL and ontologies in the OWL representation (Tan et al., 2023). Based on this feature, using ChatGPT could be considered to be an alternative approach for information retrieval for the RDF data, which would be easy and simple to use. ChatGPT can also be used to directly retrieve information from a provided ontology, including instance data, without using SPARQL queries. However, due to the private and sensitive nature of company and construction-related instance data, it is imperative not to share this kind of data with a self-learning LLM model.

However, the accuracy and real capability of ChatGPT to generate practical SPARQL queries for achieving specific information retrieval in the construction domain have not been tested. To assess whether if it is a feasible solution to aid construction information retrieval tasks, in this paper, we aim to test and evaluate the current capability of ChatGPT to generate the SPARQL queries for accomplishing domain-specific construction information retrieval tasks. The Digital Construction Ontologies (DiCon) (Zheng et al., 2021) previously developed by our research group is used as a case study for the test. We first feed the DiCon ontologies to the ChatGPT for the initial ontological parse as the fundamental context of generating the SPARQL queries. We experimented on the generated SPARQL with four different scenarios and used the metrics of syntactical correctness, plausible structure, and the coverage of the correct result to assess the generated SPARQL queries.

The paper is structured as follows. Section 2 provides a review of related works of ontologies and ChatGPT. Section 3 introduced the research methodology and the architecture of the DiCon-ChatGPT system. In Section 4 the tests and results are illustrated. This is followed by the discussion, limitation, and future research in Section 5. Finally, in section 6 the conclusion of the paper is given.

2. BACKGROUND

2.1 Semantic Web and construction information

Data and information are the key resources of the construction industry to guarantee smooth collaboration for the operations. However, data and information are also influenced by the segmented nature of the construction industry and the diverse software solutions in use. The data is generated in isolated systems by the different stakeholders in the different disciplines and is usually formed into various file formats (Kosovac et al., 2000). Such interoperability issue is notorious in the construction domain. The semantic web and associated Linked Data concept is considered as an approach to facilitate the information interoperability problem, which provides technical standards for a comprehensive heterogeneous information integration and machine-readable representation of the data for further implementation (Beetz et al., 2021).

Ontologies serve as the foundation of the semantic web, which provides a shared and machine-readable conceptualization of domain knowledge and could be further used as a data structure to formalize and integrate data and information. Recently, ontologies have become increasingly relevant in the construction domain, to address challenges like data integration and knowledge management (Pauwels et al., 2017). Various iconic domain ontologies have been created in the construction field, including generic ones like e-COGNOS (El-Diraby et al., 2011) and IC-PRO-Onto (El-Gohary et al., 2010). Our research team also developed the DiCon, which defines construction workflow-related entities with the Semantic Web Ontology Language (OWL) representation and achieves integrating data from diverse systems. DiCon also aligned with other ontologies such as IFCOWL (Pauwels, 2016), the Building Topology Ontology (BOT) by Rasmussen et al. (2020), and SOSA/SSN ontology (Janowicz et al., 2019) to link the construction information with building information modeling, topological, and sensor data.

The foundational language of the semantic web is the Resource Description Framework (RDF). The data structure of RDF is organized into a triple format of a subject, predicate, and object, or subject, property, and value (Manola et al., 2004). The RDF triples constitute an RDF graph and store it in RDF graph stores for further utilization such as information retrieval (Hitzler et al., 2008; Allemang et al., 2020). SPARQL Protocol and RDF Query Language (SPARQL) is a semantic graph query language specifically designed to query RDF data (W3C, 2013). SPARQL allows users to query RDF data by specifying patterns to match against the triples in the RDF graph. Utilizing an RDF-based ontology together with SPARQL can significantly enhance the efficiency of information extraction.

2.2 ChatGPT and related works

ChatGPT is a state-of-art large language model (LLM) developed by OpenAI. The G to refers to generative, which means the ability to generate human-like responses and demonstrate a level of language understanding that has been groundbreaking in the field of natural language processing. P refers to pre-trained because the model is pre-trained on a massive dataset containing a diverse range of text from the internet. The T refers to Transformer, the architecture of a deep-learning model designed for natural language processing tasks (van Dis et al., 2023). It learns to predict the next word in a sentence, which enables it to understand grammar, context, and semantics in the text. Currently, ChatGPT has four versions, including GPT-1, GPT2, GPT3.5, and the latest GPT4.

As an AI language model, ChatGPT's responses are based on patterns learned from a diverse range of data during training, which includes general information on ontologies, SPARQL, and RDF. Therefore, ChatGPT also can collaborate with Semantic Web technologies. Several scholars have explored the combination of ChatGPT with semantic web technologies in different directions. Lin et al. (2023) involved context-based ontology modeling with ChatGPT to represent database semantics in natural language representation for supporting database management in data integration. Tan et al. (2023) assessed the capability of ChatGPT to conduct knowledge-based question-answering with generated SPARQL queries. The experimental results showed that ChatGPT is a promising tool for question-answering under continuous updating and iterating of the model. Meyer et al. (2023) conducted a set of experiments using ChatGPT with the knowledge of graph engineering. The result showed ChatGPT has a remarkable capability to support knowledge graph engineering in constructing knowledge graphs, translating natural language queries into precise and organized SPARQL queries, tailored to the provided knowledge graphs, and diagrams illustrating expansive schemas of knowledge graphs.

In summary, the existing tests indicate that ChatGPT can generate SPARQL queries for information retrieval tasks. Therefore, ChatGPT could be a potential tool for the automated generation of SPARQL queries for construction information retrieval and management. However, the previous related works focus on assessing the capability of ChatGPT in general Linked Data and knowledge graph domains. The capability of ChatGPT to generate domain-specific SPARQL queries for construction information retrieval has not been explicitly tested or validated. Therefore, in this paper, we aim to test and evaluate the current capability of ChatGPT to generate SPARQL queries for retrieval-specific construction workflow information.

3. METHODOLOGY

3.1 Research design

To achieve the identified research objective, the research is designed as shown in Fig.1. We set up an experimental case study of feeding the DiCon ontologies to ChatGPT to test its current capability of generating the SPARQL for retrieval construction information. The results of the experiment are the generated SPARQL queries. To evaluate these queries, we followed Meyer et al. (2023) who selected syntactical correctness, plausible structure, and the coverage of the correct result as three metrics to assess the generated SPARQL queries. First, syntactical correctness aims to check whether the query is following the correct syntax of SPARQL. A query is considered syntactically correct if it can be executed by an SPARQL engine without encountering errors. Second, the plausible structure is used to assess if the query has missing prefixes, wrong use of the classes and properties, or ChatGPT creates the classes and properties out of the given ontologies. The plausible structure is evaluated by manual assessment. Third, the coverage of the correct results is investigated by comparing the query result of the generated SPARQL with the ground truth data.

To evaluate the capability of the ChaptGPT in different scenarios four tests are designed. In the first test, we asked the ChatGPT to create SPARQL queries for direct construction information retrieval. The second test was to ask ChatGPT to generate the SPARQL queries based on different natural language expressions of prompts. In terms of the third test, we asked ChatGPT to generate CONSTRUCT-type SPARQL queries based on updating information based on the given ontology. Finally, we guide ChatGPT to refine the SPARQL queries which are not performing well, to assess the performance of the refinement. Finally, the results of the four tests are assessed

based on the previously defined metrics.

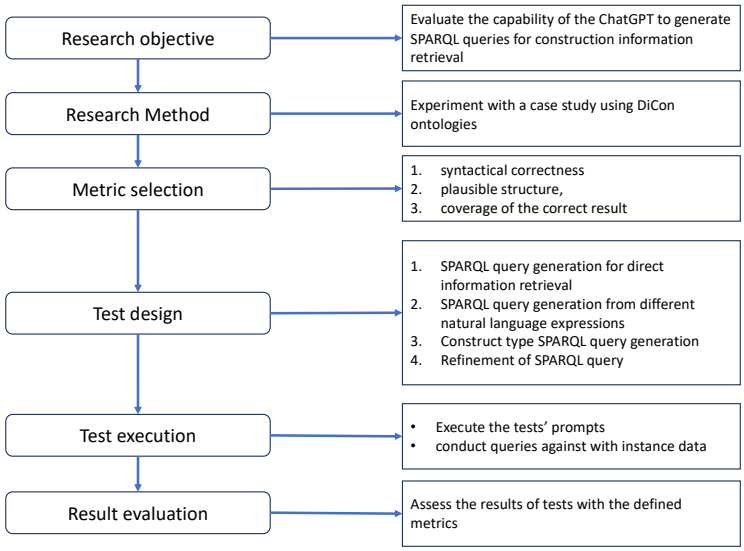

Fig. 1: Research design

3.2 The technical architecture of the experiment

To conduct the above pre-defined tests, we set up a technical architecture of the experiment following the ChatGPT prompt engineering guideline (White et al., 2023) shown in Fig.2. First, due to the large size and high complexity of the original DiCon ontologies, in tests we created a subset of the DiCon ontology contains all essential classes and properties that mapped with the example data graph. Such a process could keep the essence of the ontology but reduce the cost of ChatGPT tokens. ChatGPT 4.0 is the latest model that can directly read textual files (OpenAI, 2023). Thus, we select this version and feed the ontology subset file in Turtle format to ChatGPT 4.0 as the prime prompt. Based on the context feature of the ChatGPT, further SPARQL generation experiments can use the prompt-input ontology subset as the basis of generating the SPARQL queries. Then, the predefined four tests are conducted with the different prompt inputs. The generated SPARQL queries are also used against the instance data stored in GraphDB, a commercial graph store platform by Ontotext (Ontotext, 2023), to retrieve the target information as an evaluation of the syntactical correctness and coverage of the correct result.

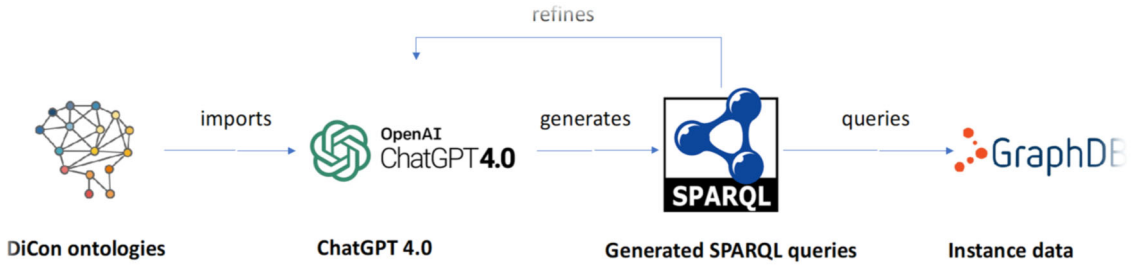

Fig. 2: Technical architecture of the experiment

4. TEST FOR DICON-CHATGPT TO GENERATE SPARQL QUERIES

In this section, a detailed description and result of the aforementioned tests are demonstrated. We also obtained the practical data from previous projects as the ground-truth data to test the SPARQL queries answering.

4.1 Test 1: SPARQL query generation for direct information retrieval

In our first test, we wanted to determine the general capability of ChatGPT to generate SPARQL queries based on the given DiCon ontology subset, to accomplish direct construction information retrieval task. Thus, we asked ChatGPT with Prompts 1.1 and 1.2 to retrieve essential construction information. Both Prompts only utilize terminologies from the DiCon (Zheng et al., 2021).

Prompt 1.1: *Based on the given ontology, create a SPARQL query to find the activity and its related agent and location.*

Prompt 1.2: *Based on the given ontology, create a SPARQL query to find the activity information about its start time and end time.*

Prompt 1.1 aims to retrieve information about construction activity, including its assigned location and agent. This prompt is simple in that in the DiCon class a *dicp:Activity* has direct properties of *dicp:hasLocation* and *dica:hasAgent* towards *dice:Location* and *dica:Agent*. Prompt 1.2 aims to extract the information of the start and end time of the activity. This prompt has an indirect relationship between activities and their start or end times. Because in DiCon, we define the property dicp:occupiesTimeInterval to represent the temporal information of an activity. The range of the *dicp:occupiesTimeInterval* is a *dice:TimeInterval*, which has beginning and end to *dice:TimeInstant* to indicate as the start and end time.

Each of the prompts was asked five times, and the generated results are partially shown in the Appendix due to the length of the paper. In terms of Prompt 1.1, for the five times of the generation, there are four times the ChatGPT uses the correct terminologies in the provided ontology. One time, it was not sure if *dicp: location* was the correct property to use, thus it defined a location property but with the wrong prefix dice: in the query. For Prompt 1.2, ChatGPT generates only one own property to describe the start and end time of an activity. To check the coverage and syntactical correctness of the query, we queried the generated SPARQL to GraphDB with the example data graph. For both of the prompts, all the queries can be successfully executed by the SPARQL engines of GraphDB, which confirms all the queries are syntactically correct. In terms of coverage, all the queries with the correct structure can query the correct answer from the example graph. The metrics results for Prompts 1.1 and 1.2 are listed in Table 1.

Table 1: Metric results of generated SPARQL queries in Test 1.

Prompt	Metric	Result
1.1	Syntactical correctness	*5/5*
	Coverage of the correct result	*4/5*
	Plausible query structure	*4/5*
1.2	Syntactical correctness	*5/5*
	Coverage of the correct result	*4/5*
	Plausible query structure	*4/5*

4.2 Test 2: SPARQL query generation from different natural language expressions

The second test intends to evaluate the capability of the ChatGPT to generate SPARQL queries based on different natural languages but with the same information retrieval target as Prompt 1.1. This test also analogs the practical nature and scenario that the different end users may have different natural language expressions for the prompts.

Prompt 2.1: *Based on the given ontology, create a SPARQL query that lists the agent and location of an activity*

Prompt 2.2: *Based on the given ontology, create a SPARQL query that lists the worker, workplace of an activity*

In terms of Prompt 2.1, we provided a different expression of Prompt 1.1 but still used the terminologies from the DiCon. In terms of Prompt 2.2, we use similar terminologies that have been mixed usually in the construction domain from describing the classes, to check if the ChatGPT can generate SPARQL with the terminologies out of the ontology. Each of the prompts was asked for five times. For Prompt 2.1, four times it generated the same queries as Prompt 1.1, and one time it used the wrong prefix of the dicp: *location* property. All the queries can be executed in GraphDB, the query with the erroneous prefix returned no location data, while the other four returned correct answers. For Prompt 2.2, only one generated SPARQL was correct to link the term 'worker' to the class agent and the term 'workplace' to the location. ChatGPT created the classes of worker and workplace one time, and the other three times it managed to link the term 'worker' to the class *dica: Agent* but failed to link the term 'workplace' to the class *dice:Location*. All of the generated queries are syntactically correct, but only one query provided us with the correct answer. The metrics results are shown in Table 2.

Table 2: Metric results of generated SPARQL queries in Test 2.

Prompt	Metric	Result

2.1	Syntactical correctness	*5/5*
	Coverage of the correct result	*4/5*
	Plausible query structure	*4/5*
2.2	Syntactical correctness	*5/5*
	Coverage of the correct result	*1/5*
	Plausible query structure	*1/5*

4.3 Test 3: Infer construction information with SPARQL

SPARQL is not just limited to querying data by using the SELECT statement. It also provides other functionalities such as INSERT, UPDATE, and CONSTRUCT statements to update or create new graph contents (W3C, 2013). Therefore, this test aims to evaluate the capability of ChatGPT to generate a SPARQL query using the CONSTRUCT statement to infer additional construction information by creating a new RDF graph. We provide the Prompt 3 to ChatGPT:

Prompt 3: *Based on the given ontology, create a SPARQL query to construct a new graph, in which if the activity has an agent, construct new triples that the agent "is an agent in" the activity.*

This prompt is based on the predefined ontology that *dica:hasAgent* property has the inversed property of *dica:isAgentIn*, which was not included in the example data graph. This test aims to assess whether ChatGPT can generate CONSTRUCT SPARQL queries that utilize the inversed properties in the ontology to construct new graphs. Similar to the previous tests, Prompt 3 was also asked five times, and we analyzed the syntactic correctness and coverage of the generated queries. The evaluation results for the prompt are listed in Table 3. The metrics results show all the generated queries with plausible query structures, are syntactically correct, and cover the correct result.

Table 3: Metric results of generated SPARQL queries in Test 3.

Prompt	Metric	Result
3	Syntactical correctness	*5/5*
	Coverage of the correct result	*5/5*
	Plausible query structure	*5/5*

4.4 Test 4: Refinement of SPARQL query

In terms of Prompt 2.2, which did not perform well in the test 2. Therefore, in this test, we try to use the contextual feature of the ChatGPT to refine the result by providing more explicit prompts based on the ontology structure. We provide the prompt P4 to the ChatGPT:

Prompt 4: *Refine the previous SPARQL query based on the given ontology, that a "workplace" should link to the class dice:Location.*

Similar to the previous tests, the refinement was also conducted five times, and we analyzed the plausible query structure, syntactic correctness, and coverage of the generated queries. After the refinement, the structure of the query has been significantly improved. All of the queries use the correct terminologies from the provided ontology without misused prefixes, are conductible, and can generate correct answers. The analyzed results for the Prompt 4 are listed in Table 4.

Table 4: Metric results of generated SPARQL queries in Test 4.

Prompt	Metric	Result
4	Syntactical correctness	*5/5*
	Coverage of the correct result	*5/5*
	Plausible query structure	*5/5*

5. DISCUSSION

Corresponding to the above four tests and their results, we assess the current capability of the ChatGPT with three predefined metrics. The overall results of different prompts with three metrics as shown in Fig.3. In the following, we will provide a detailed summary and discussion of the experiment results. First, in terms of syntactical correctness, it is obvious from the experimental results that all the ChatGPT-generated queries in the experiment avoid syntax errors. Such absence of syntactical errors across all generated queries proves ChatGPT comprehends and applies the syntactic rules inherent to SPARQL. This accomplishment is not merely an incidental outcome, but a demonstrable indication of the underlying capabilities that ChatGPT harnesses in generating high-quality, error-free SPARQL queries.

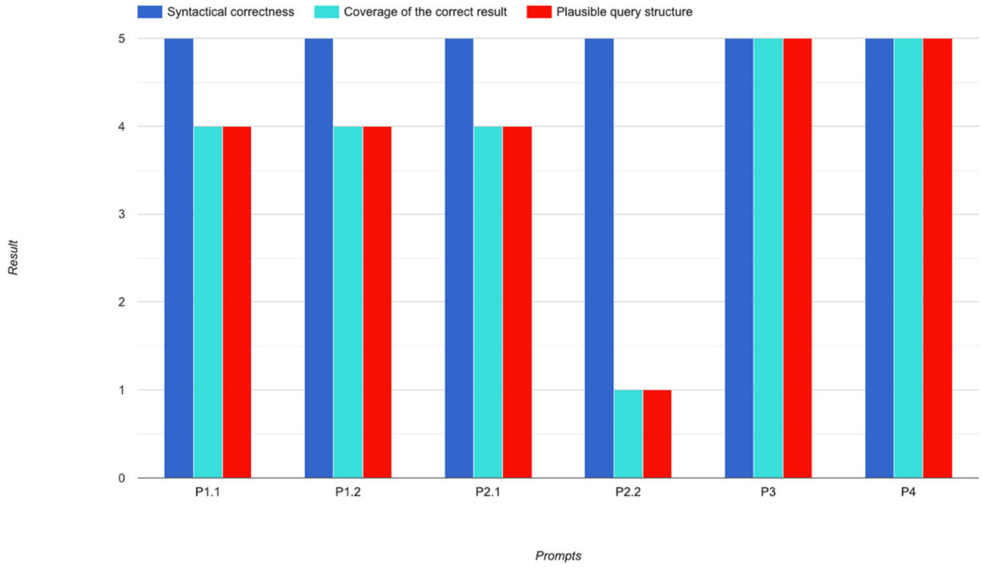

Fig. 3: Overall metrics result of the generated SPARQL for different prompts

Second, when examining the plausible query structure, the detailed analysis revealed that some of the generated results contain structure mistakes. For example, both Prompts 1.1 and 1.2 have one generated query that did not use the correct property in the given ontology but defined a new property as an assumption to complete the query. Although the used terminology is compliant with DiCon, the prefix used was incorrect. For Prompt 2.2, when faced with terminologies outside the ontology, performance had reduced significantly. In this case, it generates only one correct query using the classes and properties defined in the given ontology. The term 'worker' was successfully mapped to the class *dica:Agent*, but the system failed to map 'workplace' to the class *dice:Location* and its associated properties. In test 4, by providing a more explicit mapping instruction prompt to ChatGPT as a refinement, the performance improved. This result reveals that the current ChatGPT still requires explicit prompts using the terminologies defined in the ontology to ensure the generation of plausible queries.

Third, the test results indicate a significant and noteworthy correlation between the structural accuracy of the queries and the subsequent performance of the AI-generated queries in producing accurate query results. In essence, this observation highlights the pivotal role that query structure plays in determining the efficacy of AI-generated queries. Meanwhile, if the generated result is unsatisfactory, ChatGPT can refine the generation by providing new prompts with the correct information to fix the errors.

In summary, the current version of ChatGPT has demonstrated impressive capabilities. It successfully translated natural language questions into syntactically correct SPARQL queries for the DiCon ontology. A detailed analysis revealed some mistakes in the generated results, which can be refined with extra explicit prompts, to elaborate more precise instructions.

6. LIMITATION

This research is just the first research of our research teams to explore the usage of Natural Language Processing (NLP) tools to combine and support semantic construction informatics. Admittedly, this research has the following limitations. First, this research is limited by testing scale. The performance of ChatGPT with entire DiCon ontologies has not been tested and limited numbers of tests have been made. In the future, the test volume and

velocity should be enlarged to improve the accuracy of the research. In the future, the full DiCon ontology suite will be also tested with ChatGPT and more comprehensive prompts. Additionally, as DiCon is aligned with other ontologies, such as Building Topology Ontology (BOT), SOSA/SSN, IFCOWL, etc., these ontologies will also be included in the upcoming more complex tests. Second, this research is only tested using ChatGPT. Currently, ChatGPT is the most used NLP tool and is easy to deploy and test. Although ChatGPT is under continuous updating, as a commercial solution, its closed source and black-box nature makes it difficult to optimize and train. Therefore, besides the ChatGPT, in future research, other LLM models and NLP tools should be also tested and explored, for example, the Falcon LLM (Technology Innovation Institute, 2023). More construction domain-specified training based on the open-sourced LLM will also be conducted. Third, the prompts themselves would affect the generated query results. Throughout this research, the prompts provided to ChatGPT have been intentionally kept simple and straightforward to gauge ChatGPT's performance in generating SPARQL queries. However, recognizing the multifaceted nature of semantic querying and the potential intricacies within construction information, future studies will necessitate a broader spectrum of test cases with more specific construction information retrieval tasks. By incorporating more intricate prompts, further research will better evaluate ChatGPT's capacity to handle complex query generation tasks in the construction domain. Another research track is also studying the prompt development manner based on the readability score.

7. CONCLUSION

This paper tested the capability of ChatGPT to generate SPARQL queries with a case study based on the DiCon ontologies for construction workflow information retrieval. We designed and conducted a set of tests to assess the current capability of ChatGPT to generate the SPARQL query for solving the construction domain-specific task of information retrieval.

Overall, as observed from the result, ChatGPT has demonstrated its impressive capability of generating SPARQL queries for the given ontology to retrieve target information on the construction domain level. The experiments show that ChatGPT can avoid syntax errors, and read and utilize the given ontologies as the basis of SPARQL generation. However, the execution of the test process and evolution of the generation result also reveal several current limitations of ChatGPT in generating SPARQL queries. For example, ChatGPT cannot fully understand the given ontology, and the quality of the generated SPARQL queries highly relies on the explicitness of the given prompts.

In future research, we would also continue the exploration of using LLM solutions, including ChatGPT and other tools, to combine with the Semantic Web implementation in the construction domain. One future research topic is to illustrate the construction information, from SPARQL queries generated by Chat-GPT on Extended Reality devices. Simultaneously, we would also test the feasibility of ChatGPT to create RDF converters, which will convert tabular data into RDF format mapped with the DiCon ontology.

ACKNOWLEDGMENTS

Partial of the researchers received financial support from the ACTOR project. ACTOR is supported by Business Finland's Low Carbon Built Environment Program and receives funding from the EU's Recovery and Resilience Facility (from 2022-). The authors are grateful for the study's financial and data support.

APPENDIX: Prompts and Example ChatGPT SPARQL results

Prompts	Example result

Prompt 1.1: *Based on the given ontology, create a SPARQL query to find the activity and its related agent and location.*	PREFIX dicp: <https://w3id.org/digitalconstruction/0.5/Processes#>
	PREFIX dica: <https://w3id.org/digitalconstruction/0.5/Agents#>
	PREFIX dice: <https://w3id.org/digitalconstruction/0.5/Entities#>
	SELECT ?activity ?agent ?location
	WHERE {
	?activity rdf:type dicp:Activity .
	?activity dica:hasAgent ?agent .
	?activity dice:hasLocation ?location .
	}

Prompt 1.2: *Based on the given ontology, create a SPARQL query to find the activity information about its start time and end time.*	PREFIX dicp: <https://w3id.org/digitalconstruction/0.5/Processes#> PREFIX dice: <https://w3id.org/digitalconstruction/0.5/Entities#> SELECT ?activity ?startTimeInstant ?endTimeInstant WHERE {

	?activity rdf:type dicp:Activity . ?activity dicp:occupiesTimeInterval ?timeInterval . ?timeInterval dice:hasStart ?startTimeInstant . ?timeInterval dice:hasEnd ?endTimeInstant . }

Prompt 2.1: *Based on the given ontology, create a SPARQL query that lists the agent and location of an activity*	PREFIX dicp: <https://w3id.org/digitalconstruction/0.5/Processes#>
	PREFIX dica: <https://w3id.org/digitalconstruction/0.5/Agents#>
	PREFIX dice: <https://w3id.org/digitalconstruction/0.5/Entities#>
	SELECT ?activity ?agent ?location
	WHERE {
	?activity rdf:type dicp:Activity .
	?activity dica:hasAgent ?agent .
	?activity dice:hasLocation ?location .
	}

Prompt 2.2: *Based on the given ontology, create a SPARQL query that lists the worker, workplace of an activity*	PREFIX dicp: <https://w3id.org/digitalconstruction/0.5/Processes#>
	PREFIX dica: <https://w3id.org/digitalconstruction/0.5/Agents#>
	PREFIX dice: <https://w3id.org/digitalconstruction/0.5/Entities#>
	SELECT ?activity ?worker ?workplace
	WHERE {
	?activity rdf:type dicp:Activity .
	?activity dica:hasAgent ?worker .
	?activity dice:hasLocation ?workplace .
	?worker rdf:type dica:Worker . # Assuming "Worker" is a subclass or type of Agent
	}

Prompt 3: *Based on the given ontology, create a SPARQL query to construct a new graph, in which if the activity has an agent, construct new triples that the agent "is an agent in" the activity.*	PREFIX dicp: <https://w3id.org/digitalconstruction/0.5/Processes#>
	PREFIX dica: <https://w3id.org/digitalconstruction/0.5/Agents#>
	CONSTRUCT {
	?agent dica:isAgentIn ?activity .
	}
	WHERE {
	?activity dicp:hasAgent ?agent .
	}

Prompt 4: *Refine the previous SPARQL query based on the given ontology, that a "workplace" should link to the class dice:Location.*	PREFIX dicp: <https://w3id.org/digitalconstruction/0.5/Processes#>
	PREFIX dica: <https://w3id.org/digitalconstruction/0.5/Agents#>
	PREFIX dice: <https://w3id.org/digitalconstruction/0.5/Entities#>
	SELECT ?activity ?agent ?location
	WHERE {
	?activity rdf:type dicp:Activity .
	?activity dica:hasAgent ?agent .
	?activity dice:hasLocation ?location .
	}

REFERENCES

Akinci, B. (2015). Situational awareness in construction and facility management. Frontiers of Engineering Management, 1(3), 283-289. https://doi.org/10.15302/j-fem-2014037.

Akinyemi, A., Sun, M., & Gray, A. J. (2018). An ontology-based data integration framework for construction information management. Proceedings of The Institution of Civil Engineers-Management, Procurement and Law, 171(3), 111-125. https://doi.org/10.1680/jmapl.17.00052.

Allemang, D., Hendler, J., & Gandon, F. (2020). Semantic Web for the Working Ontologist. New York, NY, USA: ACM. https://doi.org/10.1016/C2010-0-68657-3.

Beetz, J., Pauwels, P., McGlinn, K., Tormä, S. (2021). Linked Data im Bauwesen. In: Borrmann, A., König, M., Koch, C., Beetz, J. (eds) Building Information Modeling. VDI-Buch. Wiesbaden: Springer Vieweg. https://doi.org/10.1007/978-3-658-33361-4_11.

El-Diraby, T. E., & Osman, H. (2011). A domain ontology for construction concepts in urban infrastructure products. Automation in Construction, 20(8), 1120–1132. https://doi.org/10.1016/j.autcon.2011.04.014.

El-Gohary, N. M., & El-Diraby, T. E. (2010). Domain ontology for processes in infrastructure and construction. Journal of Construction Engineering and Management, 136(7), 730–744. https://doi.org/ 10.1061/(asce)co.1943-7862.0000178.

Hitzler, P., Krötzsch, M., Rudolph, S., & Sure-Vetter, Y. (2007). Semantic Web. Grundlagen. Berlin, Heidelberg: Springer Berlin Heidelberg (SpringerLink Bücher). https://doi.org/10.1007/978-3-540-33994-6.

Janowicz, K., Haller, A., Cox, S. J., Le Phuoc, D., & Lefrançois, M. (2019). SOSA: A lightweight ontology for sensors, observations, samples, and actuators. Journal of Web Semantics, 56, 1-10. https://doi.org/10.1016/j.websem.2018.06.003.

Kosovac, B., Froese, T., & Vanier, D. (2000). Integrating heterogeneous data representations in model-based AEC/FM systems. Proceedings of CIT, 2, 556-567. Retrieved July 23, 2023, from https://itc.scix.net/paper/w78-2000-556

Lin, W., Babyn, P., & Zhang, W. (2023). Context-based Ontology Modelling for Database: Enabling ChatGPT for Semantic Database Management. Retrieved July 23, 2023, from arXiv preprint arXiv:2303.07351.

Manola, F., Miller, E., & McBride, B. (2004). RDF primer. W3C recommendation, 10(1-107), 6.

Meyer, L. P., Stadler, C., Frey, J., Radtke, N., Junghanns, K., Meissner, R., ... & Martin, M. (2023). LLM-assisted Knowledge Graph Engineering: Experiments with ChatGPT. Retrieved July 23, 2023, from arXiv preprint arXiv:2307.06917.

Ontotext. (2023). GraphDB. Retrieved July 23, 2023, from https://graphdb.ontotext.com/.

OpenAI. (2023). GPT-4 is OpenAI's most advanced system, producing safer and more useful responses. Retrieved July 23, 2023, from https://openai.com/gpt-4.

Pauwels, P., & Terkaj, W. (2016). EXPRESS to OWL for the construction industry: Towards a recommendable and usable ifcOWL ontology. Automation in Construction, 63,100-133. https://doi.org/10.1016/j.autcon.2015.12.003.

Rasmussen, M. H., & Lefrançois, M. (2018). Ontology for property management. Retrieved July 23, 2023, from https://w3c-lbd-cg.github.io/opm/.

Tan, Y., Min, D., Li, Y., Li, W., Hu, N., Chen, Y., & Qi, G. (2023). Evaluation of ChatGPT as a question answering system for answering complex questions. Retrieved July 23, 2023, from arXiv preprint arXiv:2303.07992.arXiv:2307.11449.

Technology Innovation Institute. (2023). Introducing Falcon LLM. Retrieved July 23, 2023, from https://falconllm.tii.ae/.

Van Dis, E. A., Bollen, J., Zuidema, W., van Rooij, R., & Bockting, C. L. (2023). ChatGPT: five priorities for research. Nature, 614(7947), 224-226. https://doi.org/10.1038/d41586-023-00288-7.

White, J., Fu, Q., Hays, S., Sandborn, M., Olea, C., Gilbert, H., Elnashar, A., Spencer-Smith, J., & Schmidt, D. C. (2023). A prompt pattern catalog to enhance prompt engineering with chatgpt. Retrieved July 23, 2023, from arXiv preprint arXiv:2302.11382.

Harris, S., & Seaborne, A. (2013). SPARQL 1.1 Query Language: W3C Recommendation 21 March 2013. W3C Recommendations. Retrieved July 23, 2023, from https://www.w3.org/TR/sparql11-query/.

Zheng, Y., Törmä, S., & Seppänen, O. (2021). A shared ontology suite for digital construction workflow. Automation in Construction, 132, 1–24. https://doi.org/10.1016/j.autcon.2021.103930.