R2[RML]-ChatGPT Framework

Alex Randles[1,*] and Declan O'Sullivan[1]

[1] *ADAPT Centre for Digital Content, Trinity College Dublin, Dublin, Ireland*

Abstract

The purpose of this paper is to explore the potential of applying Large Language Models (LLMs) in the processes involved in linked data publication, which require a high level of domain knowledge. In particular, we are interested in the semantic and syntactic correctness of data provided by LLMs, which could be used during the development of declarative uplift mappings. The R2[RML]-ChatGPT Framework is proposed, which integrates ChatGPT to gather useful quality insights on uplift mappings required in the publication of linked data. Two system experiments were conducted, which involved inputting mappings to test the correctness of returned knowledge. The semantic correctness of key ontology terms related to 50 distinct concepts were measured. Furthermore, 150 files of relevant code were automatically generated using the framework and measured for syntactic correctness. Moreover, the framework attempted to resolve invalid syntactics, which were then reassessed.

Keywords

Semantic Web, Mapping Quality, Linked data Generation, ChatGPT.

1. Introduction

Data quality is often defined as *"fitness for use"* [1] and is a multidimensional concept, which is determined by the stakeholders and factors involved in the creation of the data [2]. The quality of the data will influence the usefulness of data for use cases of consumers [1]. Currently, quality assessment within the linked data domain is commonly performed on the published data and is the responsibility of data consumers rather than the producers [3]. *'Uplift'* mapping artefacts typically are responsible for the generation of Resource Description Framework (RDF) data published on the web [4]. Uplift declarative mappings are used to define transformation rules for converting non-RDF data (e.g. excel or relational data) into RDF representation. Common languages include the World Wide Web Consortium (W3C) recommendation named the RDB to RDF Mapping Language (R2RML) [5]. R2RML was designed to express customized transformation rules for converting relational databases to RDF data. Another prominent language is the RDF Mapping Language (RML) [6], which extends the R2RML vocabulary to support source data represented in heterogenous formats, such as CSV, JSON and XML. These two languages are commonly used in the linked data publication process in the semantic web domain [4] and inherently impact the quality of the resulting linked data [1,2]. Creating these mappings is

KGCW'24: 5th International Workshop on Knowledge Graph Construction, May 27, 2024, Crete, GRE

* Corresponding author.

alex.randles@adaptcentre.ie (A. Randles); declan.osullivan@adaptcentre.ie (D. O'Sullivan)

0000-0001-6231-3801 (A. Randles); 0000-0003-1090-3548 (D. O'Sullivan)

a complex, time-consuming task, which is frequently error prone [2]. Often quality issues within these mappings are not detected until the dataset has been published [1,2]. In addition, creating high quality mappings requires a high level of relevant background knowledge [4]. Background knowledge in this context is described as information which informs design decisions involved in declarative mappings, such as knowledge on ontologies, RDF data querying and validation.

The research described in this paper makes a connection between LLMs [7] and semantically represented knowledge graphs, which are generated by declarative uplift mappings. The recent increasing uptake of ChatGPT [8] has demonstrated its ability to provide semantically correct natural language, however, can it provide semantically correct RDF concepts? In addition, can it syntactically correct RDF related code, such as Terse RDF Triple Language (Turtle) [9] and SPARQL Protocol and RDF Query Language (SPARQL) [10]?. Previous research [11] has applied the ChatGPT LLM to enrich data contained in resulting linked data and states that *"One of the aspects which have not yet taken over the spotlight is the combined application of these models with semantic technologies to enable reasoning and inference."*. Thus, additional exploration is warranted to discover the possible benefits of the application of LLMs in the publication of linked data. The intuition is that applying LLMs early in the publication process of linked data could result in improved quality, by informing agents while completing important uplift mapping design decisions. In order to explore the use case, we propose the **R2[RML]-ChatGPT Framework**, which was designed to provide insights from ChatGPT 3.5 turbo[2] on aspects of uplift mappings defined in R2RML and RML.

This paper is structured as follows: Section 2 presents the design and implementation of the framework. Section 3 presents the experiments completed on the framework. Section 4 describes related work. Section 5 outlines future work and concludes the paper.

2. Design and Implementation of R2[RML]-ChatGPT Framework

This section presents the design and implementation of the R2[RML]-ChatGPT Framework[3].

2.1. Design

Figure 1 presents an overview of the components and activities involved in the framework. The initiation of the framework involves an input of an uplift mapping. The concepts in the mapping are used to retrieve information from ChatGPT by inserting them into predefined prompt templates. The processing of user input and output from ChatGPT shown in the diagram is described in the following subsections.

---

[2] https://platform.openai.com/docs/models/gpt-3-5-turbo

[3] Video demo at https://drive.google.com/file/d/1_f_bssrOL5e6ATD0Ee1NCkuql8GYy2OE

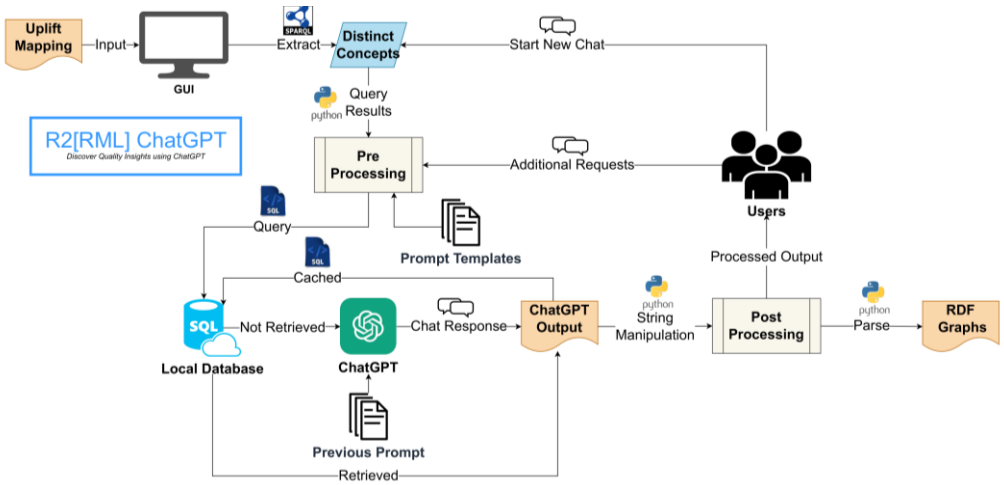

Figure 1: Workflow of the R2[RML]-ChatGPT Framework

Pre-Processing. Figure 2 presents an overview of the activities involved in pre-processing of the framework.

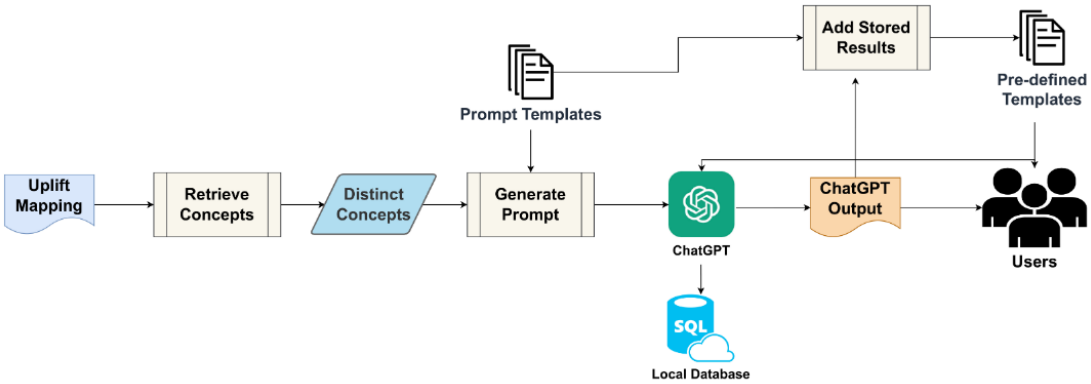

Figure 2: Workflow of Pre-Processing by the framework

Pre-processing involves retrieving necessary information from the input uplift mapping and generating prompts for ChatGPT. First, an uplift mapping is input into the framework using the GUI. Thereafter, a SPARQL [10] query[4] is executed on the mapping to retrieve distinct concepts (classes and properties). Each of the concepts are fed into prompt templates, which are designed to provide initial useful insights related to each concept. Initially, the framework executed a new request to ChatGPT [8] for each concept in order to retrieve the initial response. However, the observed processing time was longer than expected as multiple requests were being executed at once. In particular, those mappings with large numbers of concepts were observed to be performance intensive. Threading[5]

[4] https://github.com/alex-randles/R2RML-ChatGPT-Experiments/tree/main/retrieve_concepts.rq
[5] https://docs.python.org/3/library/threading.html

was experimented with to improve the performance, however, the results varied depending on the current load of the ChatGPT model. Thus, it was decided to include a relational database to store previously retrieved prompts, in order to improve the performance of the framework (acting as a cache). The database is queried to find if the initial generated prompt has been previously requested. Thereafter, the cached response retrieved from the database is output to users. Otherwise, ChatGPT is queried and the response cached for later reuse. It was decided not to use a triple store for storing responses as these are represented in natural language rather than semantic data. Values which could be later used for additional prompts are stored in specific chat threads (<chat_id>). For instance, the value retrieved for the range (rdfs:range) of a property (<property>), which was requested can be extracted from the response using a regular expression ("'rdfs:range' is '(.*)' "). Thereafter, the value is stored in a dictionary default mapping (<chat_id>:{'rdfs:range':<regex_result>}). The dictionary can be queried when additional prompt buttons available on the framework are pressed, such as the "SHACL Shape #1" button, which creates a constraint in the Shapes Constraint Language (SHACL) [12]. SHACL is a W3C recommendation designed to allow the definition of constraints in RDF format using the provided ontology terms.

Post-Processing. **Figure 3** presents an overview of the activities involved in post-processing of the framework.

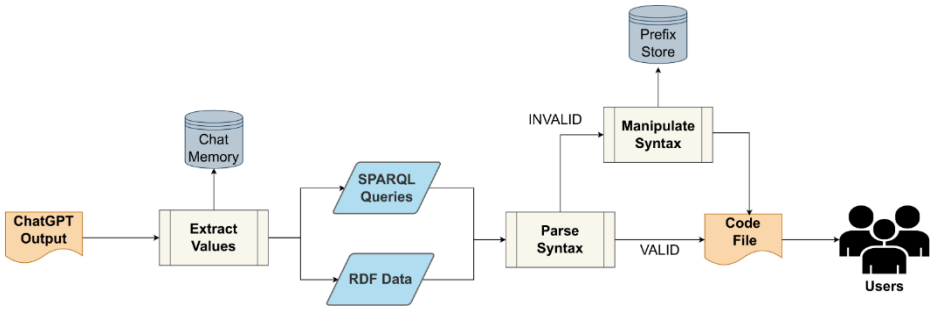

Figure 3: Workflow of Post-Processing by the framework

Post-processing involves retrieving necessary information from responses to feed into other prompts or extraction and validation of sample data. The response from ChatGPT is either processed to extract relevant values, which can be used in additional queries or to extract requested code. A regular expression ("(.*?)") is used to extract concepts from responses, which can be inserted into additional prompt templates. For instance, the range of a property which was requested can be inserted into another prompt in order to create a SHACL [12] constraint to validate the resulting dataset. However, it was observed that some of the data returned was missing required prefix definitions. Regular expressions ('Undefined prefix "(.*)"', 'Prefix "(.*)"') are used to extract information from the output of the syntax parser if the code was unsuccessfully parsed. The extracted information can be

used to improve the quality of the code by inserting prefixes stored in a CSV file[6]. The resulting data can be exported into a file using the framework, with a visual indication of syntactic correctness provided to users.

2.2. Implementation

The framework was implemented using several Python libraries[7]. Flask was used to develop the web framework. SPARQLWrapper is used to execute SPARQL queries. RDFLib is used to parse RDF data. The Open-AI library is used to communicate with ChatGPT. **Figure 4** presents a screenshot of the implementation information that is displayed by the framework related to `prov:generatedAtTime`[8].

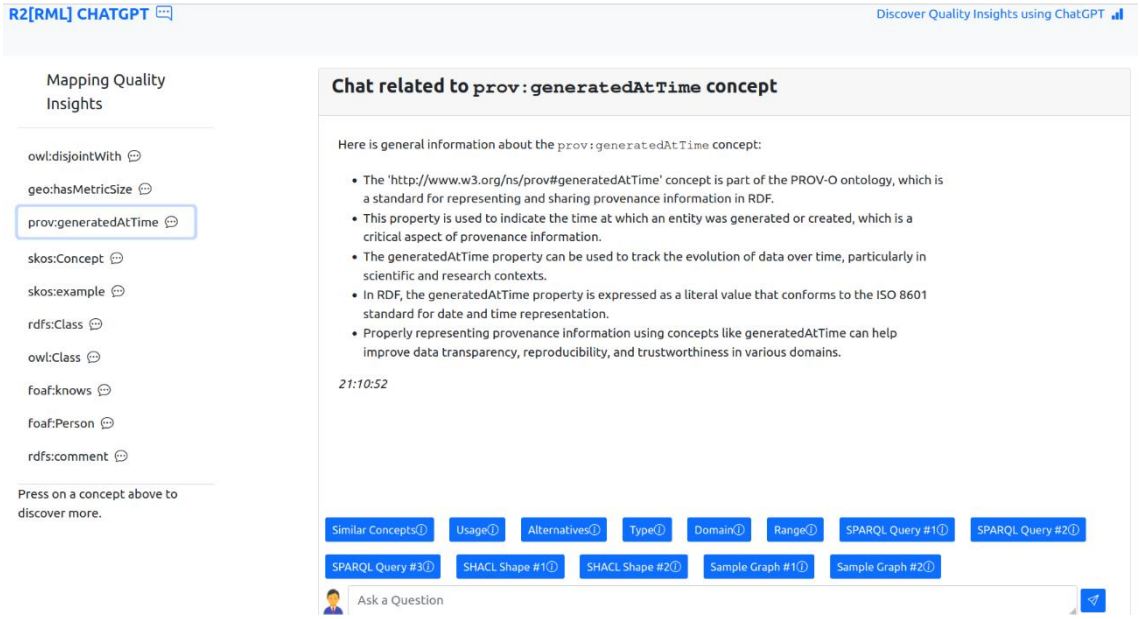

Figure 4: Screenshot of Implementation displaying information related to concepts in a mapping

The framework allows users to interact with chats related to the various concepts using the side bar shown. The initial prompt response shown is created using the following prompt templates: *"Can you provide me with key information related to the '<concept_name> used in RDF/OWL Technology?"*, which inserts concepts retrieved from the input mapping. It is hoped this initial information can provide useful insights on each concept in the uploaded mapping. The blue buttons above the input text area are designed to input further prompts based on the current concept. For instance, the "Range" button, inserts the respective concept into the following prompt: "*Can you tell me the 'rdfs:range' value for the RDF concept named '<concept_name>?*". Thereafter, the term returned for the range can be

[6] https://github.com/alex-randles/R2RML-ChatGPT-Experiments/tree/main/prefixes.csv
[7] Libraries used at https://github.com/alex-randles/R2RML-ChatGPT-Experiments/tree/main/libraries.pdf
[8] http://www.w3.org/ns/prov#generatedAtTime

compared with the mapping to ensure consistent use of the respective ontology. Hover text is provided on the framework for each prompt button to further clarify their intended usage. Certain buttons are designed to output SPARQL ("SPARQL Query #1"), Turtle ("Sample Graph #1") and SHACL ("SHACL Code #1") code.

Figure 5 presents the functionality used to export and validate RDF related code from the framework.

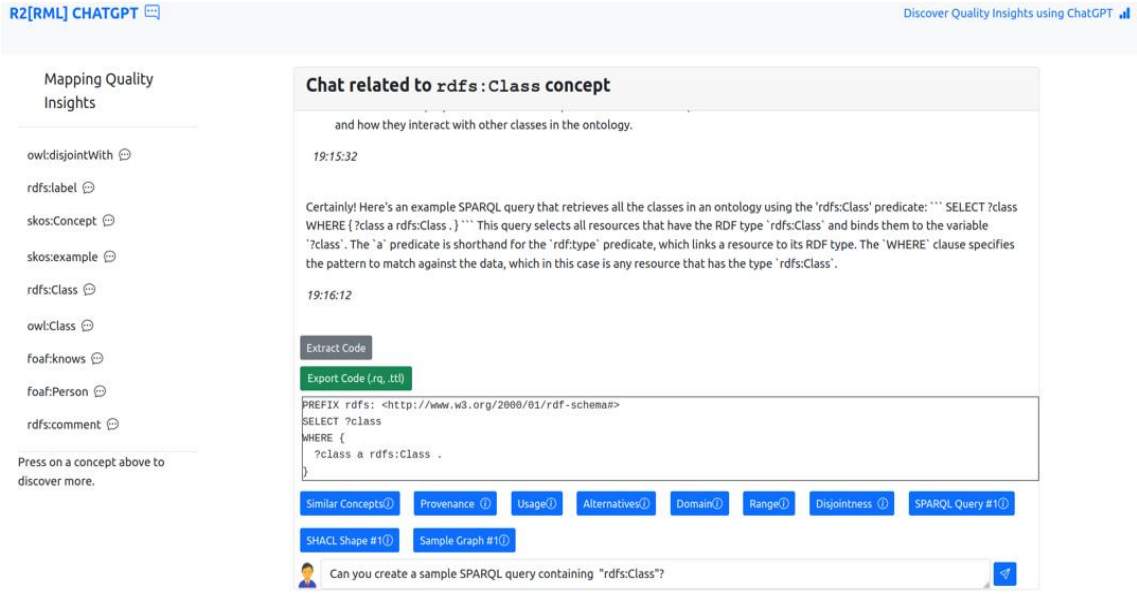

Figure 5: Screenshot of code validation and exportation available on the framework

First, the code can be extracted ("Extract Code") from raw the response received from ChatGPT. The processed code can be export ("Export Code") into a file for reuse later. A green export button (as shown) indicates the code was successfully parsed. A red button indicates to users that the syntax parsing was unsuccessful, and the framework could not repair the code. The sample shown includes a SPARQL query for checking if a resource is defined as a type of `rdfs:Class` [13]. As can be seen, the initial response from ChatGPT was missing prefix definitions, which were added by the framework.

3. Experimentation

A system experimentation[9] was conducted in order to validate two key aspects related to the application of ChatGPT for supporting the mapping quality improvement use case. These aspects related to the semantic and syntactic correctness of RDF concepts and code output by the developed R2[RML]-ChatGPT framework. "Code" in this experiment, refers to Turtle data and SPARQL queries. Two research questions (**RQ**) were posed in order to explore these aspects:

[9] Experiment Results at https://github.com/alex-randles/R2RML-ChatGPT-Experiments

RQ1: *To what extent will ChatGPT produce **semantically** correct data for certain values in a declarative uplift mapping (e.g. type, domain, range and label)?*

RQ2: *To what extent will ChatGPT produce **syntactically** correct RDF data and SPARQL queries?*

RQ1 was tested by retrieving ontology terms for concepts using ChatGPT and then comparing it with the term definition from the respective ontology. Matching terms would indicate that ChatGPT has output a semantically correct concept. For instance, the range (`rdfs:domain`) of the `foaf:based_near` property in the FOAF ontology [14] is defined as `geo:SpatialThing`[10], which would be compared to the domain that is output by ChatGPT when provided with a respective prompt. The comparison takes into account subclasses of the concept output. The `foaf:Person` concept is a valid domain for this case as it is a subclass of `geo:SpatialThing`. **RQ2** was tested by validating the syntax of the RDF and SPARQL code output by ChatGPT. The framework attempted to resolve issues in incorrect syntax using the post-processing described in Section 2.1. Thereafter, the updated syntax was assessed similarly. **Figure 6** presents an overview of the activities involved in the experimentation.

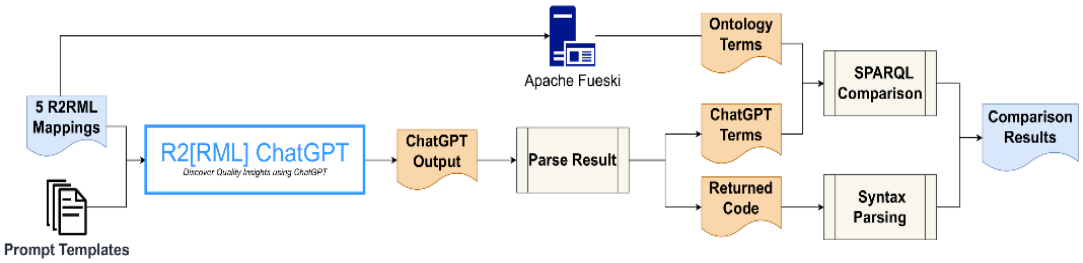

Figure 6: Overview of Activities involved in the Experimentation

Initially, the 50 distinct common RDF concepts (25 classes and 25 properties) in 5 R2RML [5] mappings[11] were input into the framework. 10 concepts (5 classes and 5 properties) were retrieved from the following five well-known ontologies: RDF [15], RDFS [13], FOAF [14], SKOS [16] and PROV-O [17]. It was decided to use these ontologies as they are designed to represent diverse information such as provenance (PROV-O), social networks (FOAF) and taxonomies (SKOS). Only concepts with the respective related properties were chosen to ensure that a value for comparison existed. For instance, only properties with an associated range were tested for range correctness. Ontology terms and RDF related code related to these concepts were retrieved from ChatGPT using prompt templates, where each respective concept name was inserted. Thereafter, relevant values were extracted from the response and stored for comparison. The comparison was completed by inserting respective values into an ASK SPARQL [10] query. Thereafter, the query was executed on the respective ontology which was stored in the local Apache Jena

[10] http://www.w3.org/2003/01/geo/wgs84_pos#SpatialThing
[11] https://github.com/alex-randles/R2RML-ChatGPT-Experiments/tree/main/mappings

Fueski Triple Store[12]. The queries when executed resulted in boolean (True or False) values, which represented the result of comparison. In addition, 150 files containing SPARQL, SHACL and Sample instances were generated using the framework, by asking ChatGPT to produce sample data related to each ontology term in the sample mappings. Relevant information for testing **RQ1** was the concept name returned for each requested ontology term. Relevant information for testing **RQ2** was associated RDF graphs and SPARQL queries.

3.1. Experiment 1: Semantic Correctness of RDF terms

Testing of **semantic correctness** (**RQ1)** involved retrieving the type (`rdf:type`), range (`rdfs:range`), domain (`rdfs:domain`) and human readable label (`rdfs:label`) associated with the 50 concepts. The results from ChatGPT were compared to the corresponding term defined in the ontology using an ASK SPARQL query template. For instance, the type of `prov:atLocation` was tested. ChatGPT was provided with the following prompt: *"Can you provide the 'rdf:type' value for the 'prov:atLocation' concept defined in an ontology used in RDF/OWL Technology?"*. The type returned was extracted from the response and inserted into a SPARQL query template[13], which queries the namespace ontology. The results of the comparison were manually validated to ensure consistent results. **Table 1** presents the results of the comparison of the output for each of the ontology terms. The ASK query returning True ("True Count") indicate the ontology term returned by ChatGPT was the same as the ontology. The query returning False ("False Count") indicates the concept output does not match the ontology. In addition, no respective value ("No Response") could be returned from ChatGPT. The label and type term related to all 50 concepts. However, the domain and range only related to properties as classes do not have these restrictions.

Table 1: Results of Semantic Correctness Experiment

Ontology Term	True Count	False Count	No Response	Results
`rdf:type`	38	12	0	Link
`rdfs:domain`	22	3	0	Link
`rdfs:range`	21	4	0	Link
`rdfs:label`	37	13	0	Link

Figure 7 presents an overview of the results shown. Each pie chart shown relates to the correctness of each ontology term tested.

[12] https://jena.apache.org/documentation/fuseki2/
[13] https://github.com/alex-randles/R2RML-ChatGPT-Experiments/tree/main/ask_query_template.rq

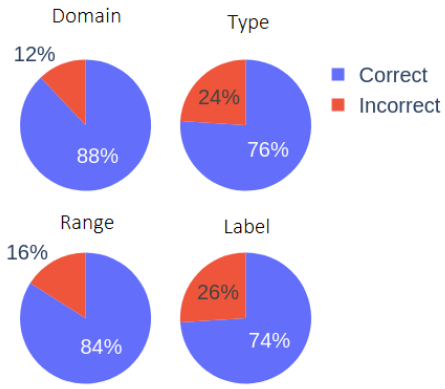

Figure 7: Results of correctness of each ontology term tested

The results indicate that ChatGPT produces correct terms for 118 (78.6%) of the tested cases, with the correctness influenced by the ontology and requested term. It scored best for retrieving the correct domain (`rdfs:domain`) of properties with 88% semantically correct terms retrieved. Slightly worse scores were identified for the range (`rdfs:range`) of properties with 84% correct. While ChatGPT was capable of providing semantically correct types (`rdf:type`) of concepts with a slightly worse degree of accuracy (76%). Most correct cases related to the generalized RDF class (`rdfs:Class`) or property (`rdf:Property`), which all RDF concepts are types of [13]. In addition, a high proportion of domain and ranges returned related to the generalized RDF resource (`rdfs:Resource`) which all RDF resources are instances [13]. Most incorrect cases related to ChatGPT returning a type of the name of the concept itself, such as `prov:Agent`, which returned a type of `prov:Agent`. Labels (`rdfs:label`) scored worst (74%), which could be a result of the natural language representation of them. As LLMs [7] are trained using natural language it could be harder for them to distinguish these values from other text when compared to RDF concepts. Interestingly, it was observed that ChatGPT made inferences about certain labels. For instance, the label returned from the `rdf:HTML` class was `"HTML/XML Syntax for RDF"`, however, the value defined in the ontology is `"HTML"`. It could have made inferences about the usage for RDF due to the context of the request. Similarly, the label returned for the `rdf:rest` property was `"rest of list"`, whereas the correct value is `"rest"`. As ChatGPT knows that the property is related to lists, it could infer the label based on the background information. In addition, a limitation for labels is that ChatGPT may not understand the common naming convention (property name in camel case). **Table 2** presents an overview of the results categorized by respective ontology.

Table 2: Results of semantically correct concepts for each ontology tested

Tested Ontology	True Count	False Count	No Response
RDFS	28	2	0
SKOS	19	11	0
FOAF	24	6	0
PROV-O	22	8	0

RDF	25	5	0

The results indicate that the ontology where the term was defined influenced the semantic correctness of values returned. In addition, the high standard deviation (3.4) indicates that the scores for each ontology were spread around the mean. As can be seen, RDF [15], RDFS [13] and FOAF [14] scored best, while SKOS [16] and PROV-O [17] scored worse. Thus, these results indicate that the ontology where the tested concept originates influences the semantic correctness of ChatGPT. However, all ontologies scored between 63% (SKOS) and 93% (RDFS) correctness for each of the 30 tests completed on them. The worse scores could be as a result of the amount and quality of documentation published by the ontology, which was used to the train the LLM [7]. **Table 3** presents a sample of results from this experiment. The tested ("Concept") is shown, along with the term returned from ChatGPT ("ChatGPT Output") and the semantically correct corresponding term ("Ontology Term") from the respective ontology.

Table 3: Samples of concepts tested in the experiment and respective values retrieved from ChatGPT and namespace ontology

Tested Term	Concept	ChatGPT Output	Ontology Term
rdf:type	rdf:Bag	rdf:Bag	rdfs:Class
	rdf:value	rdf:Property	rdf:Property
	rdf:Statement	rdf:Class	rdfs:Class
	foaf:Document	foaf:Document	owl:class
rdfs:domain	foaf:based_near	foaf:Agent	geo:SpatialThing
	rdfs:range	rdfs:Property	rdf:Property
	prov:hadMember	prov:Collection	prov:Collection
	foaf:knows	foaf:Person	foaf:Person
rdfs:range	skos:topConceptOf	skos:Concept	skos:ConceptScheme
	prov:used	prov:entity	prov:Entity
	foaf:based_near	geo:SpatialThing	geo:SpatialThing
	prov:generatedAtTime	xsd:dateTime	xsd:dateTime
rdfs:label	rdf:HTML	"HTML/XML Syntax for RDF"	"HTML"
	foaf:based_near	"based near"	"based near"
	skos:Collection	"A collection of concepts"	"A collection of concepts"
	rdf:rest	"rest of list"	"rest"

The overall results show that ChatGPT can provide semantically correct concepts with 118 (78.6%) correct for this use case. Overall, it can be concluded from **RQ1** that the framework could be beneficial for agents involved in quality assessment of the publication process of linked data. The information related to ontology reuse could be used by agents to inform crucial design decisions which will impact overall quality.

3.2. Experiment 2: Syntactic Correctness of RDF-related code

Testing of **RQ2** involved retrieving a sample instance Turtle [9] graph, sample SHACL [12] shape and sample SPARQL [10] queries related to the 50 ontology concepts. In total, 150 files (3 for each concept) of code were generated. **Table 4** presents the results of inputting

these 150 files into the RDFLib parser[14] in order to validate **syntactic correctness**. Each tested category consisted of a total of 50 files. Files which were tested and contained initially correct code syntax required no further actions. Incorrect code was post-processed by the framework as described in Section 2.1, resulting in the final syntax of the code.

Table 4: Results of Syntax Correctness Experiment

Category	Initial Correct	Final Correct	Results
SPARQL Query	43	47	Link
SHACL Constraints	46	50	Link
Sample Instances	43	49	Link

Figure 8 presents an overview of the results shown. The percentage of each category correct before (left) and after (right) post-processing.

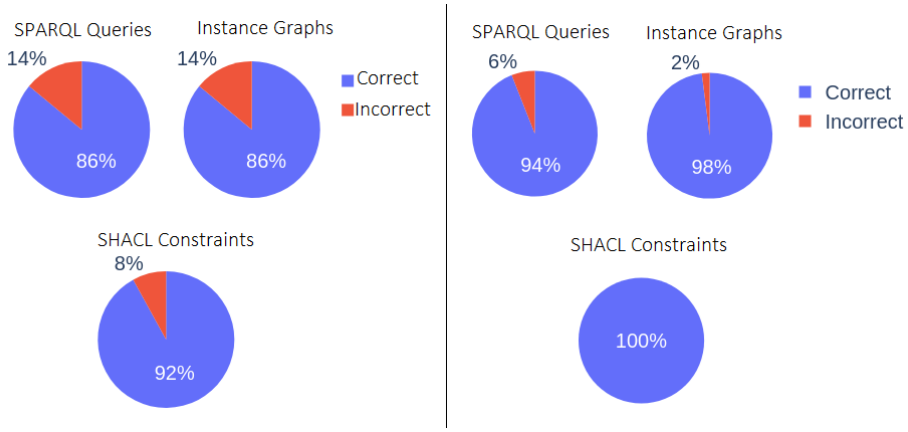

Figure 8: Results of occurrences of syntactic correctness for each category tested before (left) and after (right) post-processing by the framework

The results show that ChatGPT can produce RDF related code with a high degree of accuracy. Initially, all categories scored better than 42 (84%) syntactically correct. The post-processing of the incorrect files resulted in an improvement to a mean of 48 (96%) for all categories. The results show SHACL constraints scored slightly better than the other categories, which could be due to less prefixes being needed in most cases. SPARQL queries and Sample instance graphs scored similar. Majority (14 out of 18) of the syntax problems were due to missing prefixes in the initial response from ChatGPT. Only 1 out of the 11 initially incorrect Turtle graphs could not be repaired using the post processing, which indicates that ChatGPT has a good understanding of the overall structure of these graphs. SPARQL queries accounted for the most (3) files where syntax could not be repaired. These results provide an indication that ChatGPT has the least understanding of them.

[14] Parser used to validate syntax at https://rdflib.readthedocs.io/en/stable/plugin_parsers.html

4. Related Work

In recent years, research into frameworks to support the generation of high-quality mappings have been conducted. The Mapping Quality Improvement (MQI) Framework [18] to improve and maintain the quality of R2[RML] [5,6] uplift mappings has been proposed and evaluated. The framework consists of two core components. The mapping quality assessment and refinement component is designed to detect quality issues and suggest semi-automatic refinements to resolve them. The change detection component detects changes in respective source data and provides suggestions on how to maintain alignment between them. An approach [1] exists which was designed to assess and refine the quality of R2[RML] mappings using rule-based reasoning, which involves executing various test cases on them. Some of the related approaches [2,3] extend existing linked data quality assessment frameworks. These approaches are designed to target mappings represented in RDF format, such as R2RML [5] and RML [6]. EvaMap is an approach [19] which was designed with the requirements in mind and uses information contained in respective ontologies to assess the quality of YARRRML[15] mappings. The assessment involves quality metrics in 7 dimensions, which are used to calculate a weighted mean score. However, these approaches are limited to information contained in ontologies used by respective mappings, which are queried in order to assess quality. Thus, mapping engineers who use these approaches are required to search other forms of data on the web to resolve issues out of scope of the used ontologies, such as alternative ontologies to reuse.

5. Future Work and Conclusion

Future work includes expanding the test cases applied during the experimentation outlined in this paper. It is hoped expanded test cases will provide further indications of the accuracy of the relevant knowledge supplied. In addition, conducting a usability experiment on the framework will help to identify limitations for respective end users. A standardized questionnaire, such as the Post-Study System Usability Questionnaire (PSSUQ) [20], which was designed by IBM could be used. The PSSUQ measures user satisfaction from a software system, which involves rating various key aspects using a Likert scale. Furthermore, the semantic correctness of generated SPARQL queries and SHACL shapes could be tested, by executing them on respective input and comparing the output with expected results. Moreover, knowledge from the generation of SHACL [21] shapes using Ontology Design Patterns [22] could be integrated into the framework to generate ontology specific constraints. Finally, the framework could be extended to support other mappings represented in RDF format, such as the Database to RDF Mapping Language (D2RQ) [23].

The R2[RML]-ChatGPT framework proposed in this paper provides possible direction for future applications of LLMs in the publication process of linked data. It is hoped the approach can be used to provide accurate quality insights on various aspects of associated artefacts to alleviate the high requirement of background knowledge from domain experts. In addition, it is hoped that the availability of diverse prompt templates will result in a more

[15] https://rml.io/yarrrml/

straightforward knowledge discovery process using the framework. The results of the experiments demonstrated that ChatGPT is capable of providing syntactically and semantically correct data. 118 (78.6%) of the 150 tested ontology terms were semantically correct and a mean of 48 (96%) of the 50 code files for the tested categories (after post-processing) were syntactically correct, which indicates a high level of accuracy. These results indicate that the information could be beneficial to mapping engineers when making crucial design decisions within the linked data publication process. It is hoped the easily accessible information covering various knowledge domains could be used to support domain experts when retrieving required knowledge during the publication process. In addition, it is hoped the automation of syntactically correct RDF related code will reduce workload for involved agents.

Acknowledgements

This research was conducted with the financial support of the ADAPT SFI Research Centre (Grant No. 13/RC/2106_P2) at Trinity College Dublin.

References

[1] A. Dimou, D. Kontokostas, M. Freudenberg, R. Verborgh, J. Lehmann, E. Mannens, S. Hellmann, R. Van de Walle, Assessing and refining mappings to RDF to improve dataset quality, in: 14th International Semantic Web Conference (ISWC 2015), 2015: pp. 133–149. https://doi.org/10.1007/978-3-319-25010-6_8.

[2] A.C. Junior, J. Debattista, D. O'Sullivan, Assessing the Quality of R2RML Mappings, in: Joint Proceedings of the International Workshop On Semantics For Transport and on Approaches for Making Data Interoperable Co-Located with 15th Semantics Conference, 2019: pp. 12–24.

[3] P. Heyvaert, B. De Meester, A. Dimou, R. Verborgh, Rule-driven inconsistency resolution for knowledge graph generation rules, Semant Web 10 (2019) 1071–1086. https://doi.org/10.3233/SW-190358.

[4] A. Crotti, C. Debruyne, R. Brennan, D. O'Sullivan, An evaluation of uplift mapping languages, International Journal of Web Information Systems 13 (2017) 405–424. https://doi.org/10.1108/IJWIS-04-2017-0036.

[5] S. Das, S. Sundara, R. Cyganiak, R2RML: RDB to RDF Mapping Language, World Wide Web Consortium (W3C) Recommendation (2012). http://www.w3.org/TR/r2rml/ (accessed April 1, 2023).

[6] A. Dimou, M. Vander Sande, P. Colpaert, R. Verborgh, E. Mannens, R. Van de Walle, RML: A Generic Language for Integrated RDF Mappings of Heterogeneous Data, in: Proceedings of the Workshop on Linked Data on the Web Co-Located Withthe 23rd International World Wide Web Conference (WWW 2014), 2014.

[7] T. Brants, A.C. Popat, P. Xu, F.J. Och, J. Dean, Large Language Models in Machine Translation, in: Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning

(EMNLP-CoNLL), Association for Computational Linguistics, Prague, Czech Republic, 2007: pp. 858–867.

[8] T. Wu, S. He, J. Liu, S. Sun, K. Liu, Q.-L. Han, Y. Tang, A brief overview of ChatGPT: The history, status quo and potential future development, IEEE/CAA Journal of Automatica Sinica 10 (2023) 1122–1136.

[9] World Wide Web Consortium (W3C) Recommendation, RDF 1.1 Turtle, (2014). https://www.w3.org/TR/turtle/ (accessed May 15, 2023).

[10] S. Harris, A. Seaborne, E. Prud'hommeaux, SPARQL 1.1 Query Language, World Wide Web Consortium (W3C) Recommendation 21 (2013) 778. https://www.w3.org/TR/sparql11-query/ (accessed April 1, 2023).

[11] M. Trajanoska, R. Stojanov, D. Trajanov, Enhancing Knowledge Graph Construction Using Large Language Models. arXiv preprint arXiv:2305.04676. (2023). https://doi.org/10.48550/arXiv.2305.04676.

[12] H. Knublauch, D. Kontokostas, Shapes Constraint Language (SHACL), World Wide Web Consortium (W3C) Recommendation (2017). https://www.w3.org/TR/shacl/ (accessed April 1, 2023).

[13] D. Brickley, R. V Guha, B. McBride, RDF Schema 1.1, World Wide Web Consortium (W3C) Recommendation (2014). https://www.w3.org/TR/rdf-schema/ (accessed April 1, 2023).

[14] J. Golbeck, M. Rothstein, Linking Social Networks on the Web with FOAF: A Semantic Web Case Study., in: 23rd AAAI Conference on Artificial Intelligence (AAAI), 2008: pp. 1138–1143.

[15] R. Cyganiak, D. Wood, M. Lanthaler, G. Klyne, J.J. Carroll, B. McBride, RDF 1.1 Concepts and Abstract Syntax, World Wide Web Consortium (W3C) Recommendation 25 (2014). https://www.w3.org/TR/rdf11-concepts/ (accessed April 1, 2023).

[16] W3C Working Draft, SKOS Core Vocabulary Specification, (2005). https://www.w3.org/TR/swbp-skos-core-spec/ (accessed May 15, 2023).

[17] T. Lebo, S. Sahoo, D. McGuinness, K. Belhajjame, J. Cheney, D. Corsar, D. Garijo, S. Soiland-Reyes, S. Zednik, J. Zhao, The PROV Ontology (PROV-O), World Wide Web Consortium (W3C) Recommendation (2013). https://www.w3.org/TR/prov-o/ (accessed April 1, 2023).

[18] A. Randles, D. O'Sullivan, Preserving the Alignment of LD with Source Data, in: Proceedings of the 4th International Workshop on Knowledge Graph Construction (KGCW) Co-Located with the 20th Extended Semantic Web Conference, 2023.

[19] B. Moreau, P. Serrano-Alvarado, Assessing the Quality of RDF Mappings with EvaMap, in: 17th Extended Semantic Web Conference (ESWC2020), 2020: pp. 164–167. https://doi.org/10.1007/978-3-030-62327-2_28.

[20] J.R. Lewis, Psychometric Evaluation of the PSSUQ Using Data from Five Years of Usability Studies, Int J Hum Comput Interact 14 (2002) 463–488. https://doi.org/10.1080/10447318.2002.9669130.

[21] H. Knublauch, D. Kontokostas, Shapes Constraint Language (SHACL), World Wide Web Consortium (W3C) Recommendation (2017). https://www.w3.org/TR/shacl/ (accessed April 1, 2023).

[22] H.J. Pandit, D. O'Sullivan, D. Lewis, An Argument for Generating SHACL Shapes from ODPs, in: Advances in Pattern-Based Ontology Engineering, IOS Press, 2021: pp. 134–141. https://doi.org/10.3233/SSW210011.
[23] C. Bizer, A. Seaborne, D2RQ – Treating Non-RDF Databases as Virtual RDF Graphs, in: Proceedings of the 3rd International Semantic Web Conference (ISWC2004), 2004: pp. 125–127.