Leveraging Large Language Models for Ontology-Based Data Access: A Preliminary Analysis

Sergio Alejandro Gómez[1,2] and Pablo Rubén Fillottrani[1,2]

[1]Laboratorio de I+D en Ingeniería de Software y Sistemas de Información (LISSI), Departamento de Ciencias e Ingeniería de la Computación, Universidad Nacional del Sur, San Andrés 800, (8000) Bahía Blanca, Argentina
Email: {sag,prf}@cs.uns.edu.ar
[2]Comisión de Investigaciones Científicas de la Provincia de Buenos Aires, Calle 526 entre 10 y 11, La Plata, Argentina

Abstract. In Ontology-Based Data Access (OBDA), we study how to represent legacy data sources using ontologies. This enables a modern, distributed, uniform data representation format with the ability to perform intelligent querying and processing. This task requires the development of software to interpret the data and express it as ontologies, which takes considerable time. On the other hand, large language models (LLM) have lately shown themselves to be great solution providers due to their ability to generate solutions from input specified in natural language by an end user. In this paper, we explore the potential of LLM to perform OBDA automatically. Our research hypothesis is that is possible to use an LLM tool like ChatGPT to perform OBDA. For this purpose, we studied ChatGPT responses with different problems associated with OBDA. We discovered that ChatGPT is able to generate ontologies from free text as well as from tables expressed as text or in CSV format. ChatGPT is also able to generate SPARQL queries, and it is also successful in expressing relational tables as ontologies being capable of correcting violations of integrity constraints when appropriately directed.

Keywords. Ontology-Based Data Access, Large Language Models, Ontologies, CSV.

1 Introduction

Ontology-Based Data Access (OBDA) [1] is a method that utilizes formal ontologies to provide a semantic layer over data tables, enhancing the understanding and retrieval of information. An ontology, in this context, is a structured framework that defines the relationships and categories within a domain, allowing for a shared and explicit representation of knowledge. OBDA is crucial for semantic structuring as it enables data to be queried and interpreted based on its meaning rather than its raw format. This approach not only improves data integration and interoperability across different systems but also allows users to

interact with complex data sets through intuitive, highly expressive definition and query languages allowing for complex reasoning on data yet retaining efficiency, thereby making data more accessible and useful for decision-making. In several past publications, we reported on the development of a tool for OBDA from diverse sources such as relational databases, CSV and spreadsheet documents with the possibility of assisted SPARQL query composition (see [2] and references therein). We are now interested in seeing how the task of OBDA can be leveraged by more modern technologies.

Large Language Models (LLMs) [3] are advanced artificial intelligence systems designed to understand and generate human-like text by processing vast amounts of language data. They use deep learning techniques, particularly transformer architectures, to predict and generate text based on the context provided. ChatGPT [4] is a specific implementation of an LLM by OpenAI that can engage in coherent and contextually relevant conversations via a web interface. [1]

In this paper, we explore the potential of LLM to perform OBDA automatically. Our research hypothesis is that is possible to use an LLM tool like ChatGPT to perform OBDA. We think that integrating ChatGPT with OBDA will enhance data interaction and accessibility by enabling users to query data using natural language. As users can easily retrieve and interact with data conversationally, this will lead to more intuitive and efficient data exploration. Therefore, this approach would make data systems user-friendly and accessible to non-experts, removing the need for specialized tools for ontology materialization and the composition of SPARQL [5] queries.

For this purpose, we studied ChatGPT responses with different problems associated with OBDA, primarily materializing ontologies from text and using OWL as the ontology representation language [6]. We discovered that ChatGPT is able to generate ontologies from free text, from tables expressed as text, and in CSV format. ChatGPT is also able to generate SPARQL queries. ChatGPT is also successful in expressing relational tables as ontologies with correction of the violation of integrity constraints when appropriately directed. In the free tier of ChatGPT, as there are restrictions on which tasks can be performed, such as the uploading of files, there are scalability issues in the generation of ontologies from tables.

The rest of the paper is structured as follows. In Sect. 2, we review related work. In Sect. 3, we explain the methodologies used in the paper to carry out the research. In Sect. 4, we show the results that we obtained by performing several experiments consisting of prompting ChatGPT to perform OBDA related tasks. Finally, in Sect. 5, we discuss our results and perform a comparison with related work along with finally foreseeing future work.

2 Related Work

Here, we review recent works in the intersection of large language models and ontology development. In Sect. 5, we compare these works with our proposal.

[1] See https://chat.openai.com/.

Trajanoska et al. [7] discuss the increasing development of LLMs and their applications, noting that combining LLMs with semantic technologies for reasoning and inference is challenging. The paper compares foundational LLMs like ChatGPT with specialized pretrained models like REBEL for entity and relation extraction. Using sustainability-related text as a case study, the authors conducted experiments to evaluate this approach, creating pipelines for automatic Knowledge Graph creation from raw text. Their findings suggest that advanced LLM models improve accuracy in this process. Additionally, they explore automatic ontology creation using foundational LLM models, leading to more relevant and accurate knowledge graphs.

The traditional process of building Ontologies and Knowledge Graphs (KGs) heavily relies on human domain experts to define entities, relationships, maintain relevance, and ensure data quality. However, LLMs offer automated solutions due to their understanding and generation of human-like natural language. Komminemi et al. [8] explore semi-automatic KG construction using open-source LLMs. Their pipeline involves formulating competency questions (CQs), developing an ontology based on these CQs, constructing KGs, and evaluating them with minimal human involvement. They demonstrate the feasibility by creating a KG on deep learning methodologies from scholarly publications. Evaluating answers via Retrieval-Augmented-Generation and automatically extracted KG concepts, they recommend a human-in-the-loop approach to assess automatically generated KGs, despite LLMs potentially reducing human effort in construction.

Joachimiak et al. [9] present the Artificial Intelligence Ontology (AIO), that is a structured system of AI concepts, methodologies, and their relationships, developed through manual curation aided by LLMs. AIO aims to provide a comprehensive framework for the rapidly evolving field of AI, encompassing technical and ethical aspects. It targets AI researchers, developers, and educators, offering standardized terminology and concepts. AIO's six top-level branches, including Networks, Layers, and Bias, support modular AI methods and deepen understanding of deep learning architectures and ethical considerations. Developed with the Ontology Development Kit (ODK) and dynamically updated through AI-driven curation, AIO remains relevant amid AI advancements. Its integration into AI research publications and BioPortal demonstrates its cross-disciplinary utility. AIO is open source, available on GitHub and BioPortal.

Baldazzi et al. [10] discuss how LLMs use fine-tuning to adapt to various goals through task-specific training data. It emphasizes the importance of aligning task specificity with domain orientation, meaning the specialization of an LLM to address tasks within a specific realm effectively. However, current fine-tuning practices often rely on publicly available or grounded data, overlooking business-level definitions and domain expertise. In contrast, Enterprise Knowledge Graphs (EKGs) can capture and enhance domain knowledge through ontological reasoning. To combine the flexibility of LLMs with the domain orientation of EKGs, the authors propose a novel neurosymbolic architecture. This architecture aims to harness ontological reasoning to construct task- and domain-specific datasets for fine-tuning LLMs.

The LLMs4OL approach uses LLMs for Ontology Learning (OL) [11]. LLMs excel in capturing complex language patterns across knowledge domains. The paradigm explores whether LLMs can apply this capability to automatically extract and structure knowledge from text. Through a comprehensive evaluation with nine LLM model families, covering tasks like term typing and taxonomy discovery across diverse knowledge genres, results show that foundational LLMs alone may lack the necessary skills for complex ontology construction. However, effective fine-tuning could make them valuable assistants, easing the knowledge acquisition bottleneck in ontology development.

The process of imbuing intelligent systems with semantic data typically involves manually designing and populating ontologies with domain-specific knowledge, which can be time-consuming, error-prone, and biased. To address that, Ciatto et al. [12] propose a domain-independent approach that leverages LLMs to automatically populate ontologies with domain-specific knowledge. Their method starts with an initial schema and query templates, querying the LLM multiple times to generate instances for classes and properties. This automates the enrichment of the ontology while ensuring compliance with the initial schema. Experts can then refine, adjust, or complement the generated instances as needed. They formalize and instantiate their method across various LLMs, demonstrating its effectiveness through a case study in the nutritional domain.

3 Methodology

In this section, we describe the methods used in this work. As explained in the introduction, we asked ChatGPT to perform several tasks related to the creation of ontologies and, in particular, to solve activities directly related to OBDA. Of the two ways of interacting with ChatGPT, namely from a text prompt or from a (Python) API, in this work we centered on the former. Also, we restricted our experiments to the free tier of ChatGPT. Thus, we separated the tasks in the following categories: expressing ontologies from simple text descriptions; performing activities related to OBDA; detecting errors and inconsistencies; entity linking, and scalability.

In relation to the expression of ontologies from simple text descriptions, we performed the following experiments: expressing a textual description of an ontology as an OWL ontology; expressing a meaningless textual description of an ontology as an OWL ontology; generating an ontology from the definition of term taken from a traditional dictionary; generating an ontology from a portion of a CSV file expressed in tabular form, and generating ontologies in different codifications of OWL like Turtle and XML.

For performing activities related to OBDA, the following experiment was directly related to the OBDA task: expressing tables expressed as text as ontologies. In particular, we were interested in the ability of ChatGPT to detect errors and inconsistencies when translating data for materializing ontologies. The following experiments aimed at: expressing a text table were there was a date which could not exist; correcting the incorrect date upon prompting; the abil-

ity of composing SPARQL queries from a textual description; generating other OWL class for a one-to-many relation; generating an ontology from a table that does not satisfy a unique value constraint with self correction; generating an ontology where a related table does not satisfy a referential integrity constraint detecting non-compliant records, and, most importantly, generating corrections for problems in the previous experiments.

Ontologies whose terminologies stand on their own are self-contained. However, due to the distributed nature of linked data, one of the tenets of the research area is that, whenever possible, ontologies have to refer to other ontologies. The task of determining if the names of the individuals discovered in an ontology can be defined in terms of an external ontology is called *entity linking*. Thus, we were also interested in checking if the tool can generate ontologies that refer to existing external ontologies without telling it what ontology to look for.

We were also concerned with the behavior of the tool in front of increasing demands in the size of the input tables. So we proposed experiments for performing: a scalability test with an embedded CSV table of 65+K records; a scalability test with a CSV table of 65+K records as an attached file, and a scalability test with a CSV table of 200 records.

4 Results

In this section, we show the results of the experiments that we performed using ChatGPT for creating ontologies and carrying out activities related to OBDA. For reasons of space, we will make a partial presentation of the results obtained. However, all the results obtained are published online in supplementary material. [2] Therefore, to reproduce the findings presented here, readers can access the online documentation and try the use cases by themselves in the free tier interface of ChatGPT. The results that we gather are summarized in Table 2.

In the rest of the section, we show some of the results along with a short accompanying explanation. The problems that we tackled are classified by its kind. We then present the results obtained for every kind of problem that we addressed. For every title, there is a footnote indicating the URL that takes the reader to the specific point in the auxiliary online documentation where the data being discussed is collected. Posteriorly, in Sect. 5, we summarize the results obtained.

Expression of ontologies from simple text descriptions.[3] ChatGPT was able to generate an ontology from the textual description of the concepts that compose it and the inclusion relationships between them. The application was able to correctly detect properties for descriptions with real elements, both as fictitious and as taken from real dictionary definitions. The ontologies were cor-

[2] See http://cs.uns.edu.ar/~sag/cacic2024.
[3] http://cs.uns.edu.ar/~sag/cacic2024/#onto-from-text

rectly produced when their format was text (i.e. is-a relations), OWL Turtle and OWL/RDF. These results corresponds to Experiments 1–3 and were successful.

Generating an ontology from a portion of a CSV file expressed in tabular form.[4] Experiments 4–7 correspond to typical OBDA tasks concerning the materialization of ontologies from tabular tables in text form, whether in CSV format as well as raw text tables. We translated simple tables for people with implicit schemas containing fields id, name, height and weight. The experiments were successful in interpreting the tables and generating the OWL ontologies in both Turtle and RDF formats.

Detecting errors and inconsistencies.[5] In Experiment 8, we were interested in the ability of ChatGPT to detect errors and inconsistencies. We first tried to express a text table where there was a date which could not exist. In particular, we proposed a table for people with fields id, name, height and birth date having an erroneous date such as 29-feb-2014. The ontology generated was loyal to the data in the table. So ChatGPT did not mention the error in the data. Therefore, in Experiment 9, we were concerned with correcting the incorrect date upon prompting (by telling ChatGPT that 2014 is not a leap year). In this case, ChatGPT detected the problem and proposed a viable correction (viz., changing the offending date for 28-feb-2014). In consequence, as the first result was a failure and the following was a success, we consider these two results a partial success.

Query composition.[6] In Experiment 10, we were interested in the ability of ChatGPT to compose SPARQL queries from textual descriptions. We proposed a simple selection query with respect to the table of the previous experiment which was successfully generated.

Harnessing relationships.[7] In Experiments 11–14, we were concerned with the ability of ChatGPT of understanding how to translate tables involved in one-to-many relations. So, in Experiment 11, we proposed if it can generate a new class from a table representing phones with its owners in the class representing persons of the previous example. ChatGPT successfully generated the related class. In Experiment 12, we tested ChatGPT with the generation of a related ontology possessing a data item not satisfying a unique value constraint. In particular, we provided a table for persons having identifiers and names, and phones with attributes identifiers, phone number and identifiers of its owners. In these cases, ChatGPT successfully generated the asked ontologies, proving that it can manage the OBDA cases with simple one-to-many relations.

Detecting unique value constraint errors.[8] In Experiment 13, we asked ChatGPT to generate an ontology from a table where the key field had repeated

[4] http://cs.uns.edu.ar/~sag/cacic2024/#onto-from-csv
[5] http://cs.uns.edu.ar/~sag/cacic2024/#error-detection
[6] http://cs.uns.edu.ar/~sag/cacic2024/#composing-sparql
[7] http://cs.uns.edu.ar/~sag/cacic2024/#relations
[8] http://cs.uns.edu.ar/~sag/cacic2024/#error-detection

values and to replace the offending values accordingly. ChatGPT suggested corrections so every individual was correctly identified, thus making the experiment a success. In Experiment 14, we asked ChatGPT to redo the tasks in Experiments 12 and 13, so the errors could be corrected. In this case, each Phone instance correctly references a Person instance using the ownedBy property. IDPersonOwner values 8 and 9 were reassigned to IDPersonOwner 1. Thus, Phone IDs 1 and 2 are owned by Person/1. And the structure maintains the integrity of the relationship between Person and Phone entities.

Entity linking.[9] We were also interested in checking if the tool can generate ontologies that refer to existing external ontologies without telling it what ontology to look for. Thus, in Experiment 15, we provided ChatGPT with a table for representing people with columns person's id and name, city of origin and the name of their favorite team (see Table 1). ChatGPT proposed known ontologies for representing elements of the table such as using FOAF [13] for naming the class Person and used DBPedia[10] por referencing both city and team names. The ontology for the table can be visualized in Fig. 1. Therefore, we consider this result a success.

Table 1. Table for people with city and favorite team

IDPerson	Persons Name	City of origin	Favorite team
1	John	New York	New York Knicks
2	Mary	Denver	Denver Nuggets
3	Peter	New York	New York Yankees

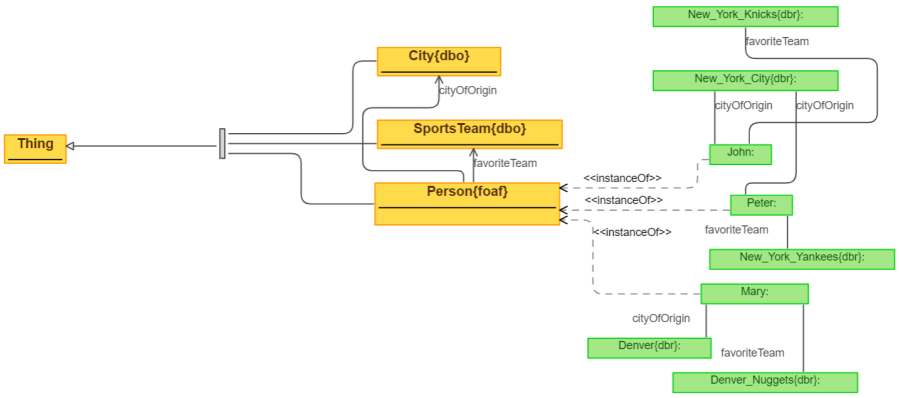

Fig. 1. Ontology for people with city and favorite team from Table 1

[9] http://cs.uns.edu.ar/~sag/cacic2024/#entity-linking
[10] https://dbpedia.org

Scalability.[11] We were concerned with the behavior of the tool in front of increasing demands in the size of the input tables. In Experiment 16, we proposed ChatGPT to express a CSV table with only two columns ID of measure and temperature reading but having about 65,000 records. ChatGPT answered that we should submit something shorter. We consider this result a failure. So, in Experiment 17, we tried with similar tables but of sizes 100 and 200 records, resp. In both cases, ChatGPT successfully generated correct ontologies. So, these results were a success. In Experiment 18, we tried again with 65,000 records but this time we specified the table as an attached file. Like in Experiment 16, ChatGPT refused to do the translation. We consider this experiment a failure.

Table 2. Summary of the results of the experiments

No. of experiment	Description	Result
1	Express a simple meaningful text as an ontology	Success
2	Express a simple meaningless text as an ontology	Success
3	Generate an ontology from a dictionary definition	Success
4	Generate an ontology from a CSV extract	Success
5	Generate ontologies in RDF/Turtle syntax	Success
6	Generate ontologies in RDF/XML syntax	Success
7	Express a text table as a RDF ontology	Success
8	Express a text table as a RDF ontology where there are an incorrect date and detecting it	Fail
9	Correct the date upon description of the problem	Success
10	Generate a SPARQL query for a simple selection	Success
11	Generate another OWL class for a one-to-many relation	Success
12	Generate an ontology from a table that does not satisfy a unique value constraint with self correction	Success
13	Generate an ontology where a related table does not satisfy a referential integrity constraint detecting non-compliant records	Success
14	Generate a correction for problems in experiments #12 and #13	Success
15	Generate ontologies from tables using entity linking	Success
16	Scalability test with a CSV table of 65+K records	Fail
17	Scalability test with a CSV table of 200 records	Success
18	Scalability test with a CSV table of 65+K records as an attached file	Fail

5 Discussion and Conclusions

Now we make a comparison of our proposal in this work with those mentioned in the related work (see Sect. 2). Trajanoska et al. [7] use two models for generating ontologies from text, namely ChatGPT and REBEL, testing their approach for the sustainability domain. We only use ChatGPT but in our case although we also tried preliminary tests on generating ontologies from textual definitions we were mainly concerned with OBDA-related tasks such as table translations and query generation. Komminemi et al. [8] semiautomatically construct KGs using LLM with a human-in-the-loop approach. Likewise, our approach is related

[11] http://cs.uns.edu.ar/~sag/cacic2024/#scalability

because OWL ontologies are a particular kind of KG and our approach requires the usage of LLM, which is implicit in ChatGPT; our approach also requires a human user to supervise the output of the process despite ChatGPT alleviating the task. Joachimiak et al. [9] present an ontology about AI called AIO which is developed using LLM. Our approach is only concerned with the generation of ontologies from tables and related data technologies. Baldazzi et al. [10] use ontological reasoning to fine tune a LLM model, we in contrast use a LLM model to develop ontologies from legacy datasources. The approach of Ciatto et al. [12] aims at populating an ontology schema by extracting facts and relations from textual sources using LLMs; our approach aims at using a LLM model for generating both the schema and performing the population of the schema but instead only from tabular sources.

We now sum up the results of our experiments using ChatGPT for creating ontologies and performing OBDA-related activities. Due to space constraints, only partial results were shown here, with full details available online in the supplementary material. Readers can reproduce these results by accessing the online documentation and trying the use cases in the free tier interface of ChatGPT. We presented the results obtained for each type of problem tackled, with each title linked to the specific point in the auxiliary online documentation. In the first experiments, ChatGPT successfully generated ontologies from textual descriptions, correctly identifying properties and relationships in both real and fictional contexts. When generating ontologies from CSV files, ChatGPT effectively translated tabular data into OWL formats. However, error detection was only partially successful, as initial error detection failed but subsequent correction attempts were successful. ChatGPT also composed SPARQL queries accurately and managed one-to-many relationships in ontology generation. It successfully detected and corrected unique value constraint errors and demonstrated the ability to link entities to external ontologies. Scalability tests showed limitations with large datasets, as ChatGPT failed to process CSV tables with 65,000 records but succeeded with smaller datasets. These findings were summarized in Table 2.

In the approach presented in this work, we mainly focused on the natural language interface provided by ChatGPT to perform OBDA. In short, we simply used an LLM as a black box and tried to see what problems it could solve relative to OBDA. There are several avenues of research that can be explored in the future, which include, for example, testing the Python programmatic interface of the ChatGPT API for having better control of the process, testing other LLM models installed locally on our computer, and also comparing the performance of such models and ChatGPT with established OBDA tools.

Acknowledgments. This work was supported by Secretaría General de Ciencia y Técnica, Universidad Nacional del Sur, Argentina, and by Comisión de Investigaciones Científicas de la Provincia de Buenos Aires (CIC-PBA).

References

1. Xiao, G., Calvanese, D., Kontchakov, R., Lembo, D., Poggi, A., Rosati, R., Zakharyaschev, M.: Ontology-Based Data Access – A Survey. In: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI-18). (2018) 5511–5519
2. Gómez, S.A., Fillottrani, P.R.: A Query-By-Example Approach to Compose SPARQL Queries in the GF Framework for Ontology-Based Data Access. In Pesado, P., ed.: 28th Argentine Congress, CACIC 2022 – Revised Selected Papers. Springer (2023) 211–226
3. Brown, T.e.a.: Language models are few-shot learners. In Larochelle, H., et al., M.R., eds.: Advances in Neural Information Processing Systems. Volume 33., Curran Associates, Inc. (2020) 1877–1901
4. OpenAI, et al., J.A.: Gpt-4 technical report (2024)
5. Diaz, G., Arenas, M., Benedikt, M.: SPARQLByE: querying RDF data by example. Proceedings of the VLDB Endowment **9** (09 2016) 1533–1536
6. Bao, J., Kendall, E.F., McGuinness, D.L., Patel-Schneider, P.F.: OWL 2 Web Ontology Language Quick Reference Guide (Second Edition) W3C Recommendation 11 December 2012 (2012)
7. Trajanoska, M., Stojanov, R., , Trajanov, D.: Enhancing knowledge graph construction using large language models (2023)
8. Krishna, K.V., König-Ries, B., Sheeba, S.: From human experts to machines: An llm supported approach to ontology and knowledge graph construction. arXiv preprint arXiv:2403.08345 (2024)
9. Joachimiak, M.P., Miller, M.A., Caufield, J.H., Ly, R., Harris, N.L., Tritt, A., Mungall, C.J., Bouchard, K.E.: The artificial intelligence ontology: Llm-assisted construction of ai concept hierarchies (2024)
10. Baldazzi, T., Bellomarini, L., Ceri, S., Colombo, A., andEmanuel Sallinger, A.G.: Fine-tuning large enterprise language models via ontological reasoning. (2023)
11. Giglou, B., Hamed, D'Souza, Jennifer, Auer, Sören: Llms4ol: Large language models for ontology learning. In: International Semantic Web Conference, Springer (2023) 408–427
12. Ciatto, G., Agiollo, A., Magnini, M., Omicini, A.: Large language models as oracles for instantiating ontologies with domain-specific knowledge. arXiv preprint arXiv:2404.04108 (2024)
13. Brickley, D., Miller, L.: FOAF vocabulary specification 0.99 – namespace document 14 january 2014 – paddington edition (2014)