OLIVE: Ontology Learning with Integrated Vector Embeddings

Yinglun Zhang [a,b], Aryan Singh Dalal [a,b], Caleb Martin [b], Srikar Reddy Gadusu [a,b] and Hande Küçük McGinty [a,b,*]

[a] *Koncordant Lab, Department of Computer Science, Kansas State University,Manhattan USA*
[b] *Department of Computer Science, Kansas State University, KS, USA*
E-mails: aryand@ksu.edu, yinglun@ksu.edu, srikarre@ksu.edu, hande@ksu.edu

Abstract.
The traditional approach to ontology development is characterized by its labor-intensive nature, requiring extensive effort and domain expertise to define intricate structures, relationships, and concepts accurately. This study proposes a paradigm shift in ontology development by harnessing the capabilities of Large Language Models (LLMs). This methodology entails the creation of an interactive interface that empowers users to query LLMs using prompts, facilitating the retrieval of pertinent information with ease. Subsequent analysis of this information allows for identifying key relationships, which are then transformed into graph structures using the Web Ontology Language (OWL). The outcome of this process is OLIVE—an ontology development workflow engineered to streamline manual efforts and minimize the risk of errors.

Keywords: Ontology, knowledge Engineering, Large Language Models, Machine Learning, Artificial Intelligence

1. Introduction

In the era of artificial intelligence and data science, ontology assumes a critical role in data management and retrieval, organizing and standardizing data to facilitate efficient analysis and pattern recognition. Additionally, ontologies support sophisticated information manipulation and advanced data processing, therefore aiding several machine learning and artificial intelligence approaches. The semantic web also derives significant benefits from ontology, leveraging it for semantic interoperability and intelligent search capabilities to deliver tailored results. Looking ahead, as data complexity continues to rise, ontology's importance is poised to increase, empowering machines to interpret and reason about data more effectively as technology evolves.

Although the manual creation of ontologies is indispensable for accurately representing knowledge domains, it comes with inherent drawbacks that must be addressed. One notable concern is the heightened risk of errors during manual development, stemming from the complexity involved in defining concepts, relationships, and properties within a domain. The intricate nature of this process often leads to long development cycles, inaccuracies and inconsistencies that can compromise the ontology's effectiveness in capturing the nuances of the domain. Moreover, the transfer of concepts and methodologies

[*]Corresponding author. E-mail: hande@ksu.edu.

from domain experts to computer scientists—typically responsible for ontology development—may introduce discrepancies and misinterpretations due to differences in expertise and perspective. This transition of information across different domains adds an additional layer of complexity and potential errors to the ontology creation process, underscoring the challenges associated with manual creation in ensuring precision and accuracy. This lack of standardization and automation in ontology creation poses significant challenges, particularly as the volume and complexity of data continue to grow exponentially. Some methods such as KNowledge Acquisition and Representation Methodology (KNARM) by Küçük McGinty et al. (2019) addressed this challenge previously. A challenge which is now multiplied with the addition of new ontology building methods that use Large Language Models (LLMs), which have become increasingly interesting to researchers and industries. Recognizing the above mentioned challenges and opportunities, in this study we describe our work on Ontology Learning with Integrated Vector Embeddings (OLIVE) workflow, which evolves KNARM by adding LLMs in the semi-automated ontology building process. During this project, we built on data and models that were established during previous studies. As a result, we focus on the semi-automated ontology building process and tools and methods for introducing OLIVE in this paper. With the entire KNARM methodology in mind, we realize that the automation of ontology building process using LLMs using OLIVE brings about concerns for the knowledge acquisition and ontology validation steps, which we will discuss as a part of this study.

In the subsequent sections, we will briefly summarize some of the current practices of ontology building and its methodologies, describe how we evolved our KNARM methodology with our OLIVE workflow, highlight the tools and methods built as a part of this study, and discuss potential strategies for enhancement. Drawing insights from various sources, we aim to contribute to the ongoing efforts to leverage the power of LLMs through the effective utilization of ontologies.

2. Literature

In the context of this research, manual ontology development refers to the process of manually formulating or defining relations, classes, and properties within the ontology. Creating an ontology involves several approaches, each suited to different needs and contexts. The manual approach involves domain experts directly engaging in the definition of classes, properties, and relationships that characterize the ontology. Manual ontology development relies heavily on human involvement and domain expertise, providing better opportunities for control and customization. This method enables practitioners to intricately craft the ontology to align with the precise requirements and objectives of a given domain. However, the introduction of increased flexibility and frequent modifications may inadvertently introduce human errors, amplifying the complexity and challenges with interpretation of the ontology. As a result, managing this heightened complexity may require heightened involvement from domain experts to identify and rectify errors, ensuring accuracy and reliability. Although ontology development tools like Protégé by Gennari et al. (2003) can enhance the manual process to some extent, they may not fully address the complexities inherent in human-centric ontology construction.

A multitude of approaches are available for ontology creation, with the choice depending on the ontology's size and dynamic nature. This section provides an overview of different ontology development methodologies, such as AGILE, DILIGENT, Methontology, KNARM, Pay-as-you-go, and MOMO. These methodologies share a common goal of prioritizing the human element in ontology construction. They aim to foster a deeper understanding of domain-specific concepts and promote the reuse of established knowledge frameworks.

In scenarios involving larger or continuously evolving ontologies, agile methodologies, such as the methodology by Peroni (2017), Pinto et al. (2009), Küçük McGinty et al. (2019) divide the ontology development process into smaller, manageable tasks, allowing for iterative and adaptive progress. As mentioned above an agile methodology is the DILIGENT methodology by Pinto et al. (2009) which focuses on the user's interaction with the ontology and the dynamic changes introduced by the user. Unlike traditional methodologies, DILIGENT is purposefully designed to support domain experts in distributed settings, facilitating collaborative ontology engineering and evolution. This end-user-centric approach makes DILIGENT well-suited for human-centric needs, but in the era of artificial intelligence, methods that don't address ontology implementation and reuse by agents may require additional steps. Another widely used approach is METHONTOLOGY Fernández-López and Gómez-Pérez (2002). It is divided into several steps, each addressing a specific aspect of ontology development. These steps include specification, conceptualization, formalization, integration, implementation, and maintenance.

Another ontology-building methodology example is KNARM (Knowledge Acquisition and Representation Methodology) (KNowledge Acquisition and Representation Methodology) by Küçük McGinty et al. (2019), which allows domain experts and knowledge engineers to build concordant, consistent, modular ontologies formalizing domain data and knowledge in a systematic way using modular ontology architecture and systematically deepening modeling for domain knowledge. One of the main focuses of KNARM is building ontologies with the computer applications that are reusing these ontologies, i.e. machine and application reuse of ontologies in addition to human reuse of ontologies, which is an important distinction for building ontologies for LLMs reuse. KNARM has been applied in various domains, including drug discovery in Lin et al. (2017) and food informatics by Küçük McGinty.

Another methodology named "Pay-as-you-go" by Sequeda et al. (2019) is an incremental approach to ontology development. It's driven by a prioritized list of business questions and is particularly useful in Ontology-Based Data Access (OBDA). The methodology involves specification, knowledge acquisition, conceptualization, integration, implementation, evaluation, and documentation. This approach allows for agility and the development of ontologies that can answer business questions in a step-by-step manner. A similar methodology named Modular Ontology Modeling(MoMo) by Shimizu et al. (2023) that conceptualizes an ontology as a composite of interconnected modules, with each module representing a fundamental notion as defined by domain experts. To instantiate ontology modules, Ontology Design Patterns are customized to fit the specific domain and use-case requirements. These modules are subsequently integrated to form a cohesive modular ontology. The inherent flexibility of a well-structured modular ontology enables individual users to readily adapt it to their unique use cases while preserving integration and relationships with other ontology versions.

A newer methodology, SPIRES by Caufield et al. (2024), focuses on ontology learning from text, leverages Large Language Models (LLMs) to extract information without needing new training data. It is engineered to convert unstructured text into a structured instance, guided by three primary inputs: a schema, an entry point class, and the input text. This transformative process unfolds through several systematic steps. Initially, it generates a prompt by crafting textual instructions and attribute templates, demarcating a clear distinction between the template and the input text. Subsequently leveraging a language model, the prompt undergoes completion, yielding a comprehensive response. The completion response is then parsed, segmenting it into a list while concurrently aligning attribute names and extracting values based on attribute range and cardinality. Leaf nodes representing named entities within the instance tree are grounded using existing vocabularies or databases. The grounded outcomes are subjected to normalization and validation against identifier constraints. The instance tree, capable of representation in JSON or YAML syntax, can be further translated into an ontological representation in OWL. This translation, coupled with additional reasoning steps, supports consistency checking and the population of missing axioms. In essence, this rigorous process empowers the extraction of structured insights from unstructured data, thereby facilitating the knowledge base population and enabling zero-shot learning.

Research has rapidly expanded to explore the application of LLMs, with recent papers providing surveys on the use of LLMs in KG engineering along with associated challenges as noticed by Meyer et al. (2023) and Pan et al. (2023). One of the study by Trajanoska et al. (2023) examined the potential for combining LLMs—ChatGPT and REBEL—with semantic technologies to allow reasoning. To extract the relations from unstructured tests and write them into a TBOX, utilize REBEL and ChatGPT. Furthermore, a further experiment is conducted using Chatgpt in which the whole ontology including TBOX and ABOX is created with a single prompt. The temperature settings are set to 0 in order to get predictable outcomes. Unfortunately, neither the study's prompts nor its findings are made available to the general public. Additionally, many prompting strategies are not explored, and only basic tests are carried out as a result of the token imitation of ChatGPT.

A study done by Funk et al. (2023) focuses on using LLM queries to build a conceptual hierarchy for a particular domain, beginning with a seed concept. They do not, however, take into account any other relations; just the subconcept/is-a relation.

A study named LLMs4OL byBabaei Giglou et al. (2023) in order to extract connections between ontology classes or instances, LLMs4OL used LLMs for Ontology learning. That said, this method did not enable the full development of an ontology; rather, it could only extract links between things.

The methodologies discussed above commonly overlook practical application and machine-driven reuse of ontologies. Additionally, validation procedures are often inadequately addressed within these methodologies. OLIVE underscores the critical importance of incorporating robust validation steps to ensure the trustworthy use of ontologies in future endeavors.

3. Method

In this study, we define a knowledge graph that represents information stored in a structured format, typically using a graph-based model such as OWL, RDF (Resource Description Framework), RDF* (an extension of RDF), or Property Graphs. This structure is designed for the integration, unification, and analysis of data, enabling sophisticated querying and inference. KGs are especially valuable in capturing relationships between entities in a manner that is both human-readable and machine-processable. We define an ontology, in the context of knowledge representation, as a formal logical model that focuses on the T-Box (terminological component), defining the types, properties, and interrelationships of the concepts within a domain. Ontologies are foundational to ensuring semantic interoperability and enabling intelligent reasoning over data.

In our paper, while we primarily leverage the graph-based representation of a knowledge graph, we integrate ontological principles to ensure the logical coherence and semantic richness of the data. Thus, OLIVE employs an ontology-driven approach to build and enhance the knowledge graph, ensuring both the structural and logical integrity of the information represented.

As mentioned above, after carefully reviewing the literature, we implemented OLIVE, which evolves KNARM by integrating Large Language Models (LLMs) to KNARM, starting from its *Database Formation* step and pouring into its *Semi-Automated Ontology Building* step in the workflow. Because KNARM is an agile methodology, as seen in Figure 1, even though LLM integration starts in the database backend step, it affects several steps of the methodology, altering each step slightly to ensure correct and usable ontology generation as a result. The effects of the LLM integration is discussed further in the discussion session.

Large Language Models (LLMs) serve as vast repositories of knowledge, capable of generating and retrieving information on diverse topics. However, these models occasionally experience hallucinations during output generation, wherein they produce content that deviates from the intended topic or contains misinformation. The prevalence of hallucination poses significant challenges in validating model outputs and impacts the development of ontology through LLMs. Hallucination of LLM could result in false information regarding the formal definition of axioms, relations among different entities, and the structure of the knowledge graph. Such inaccuracies can propagate misinformation, as LLMs may generate information that is not factually grounded. In certain scenarios, hallucinations may even compromise privacy and raise safety concerns, especially in critical domains like healthcare. Despite concerted efforts to minimize hallucination, eliminating this issue entirely from LLMs remains elusive also seen in the

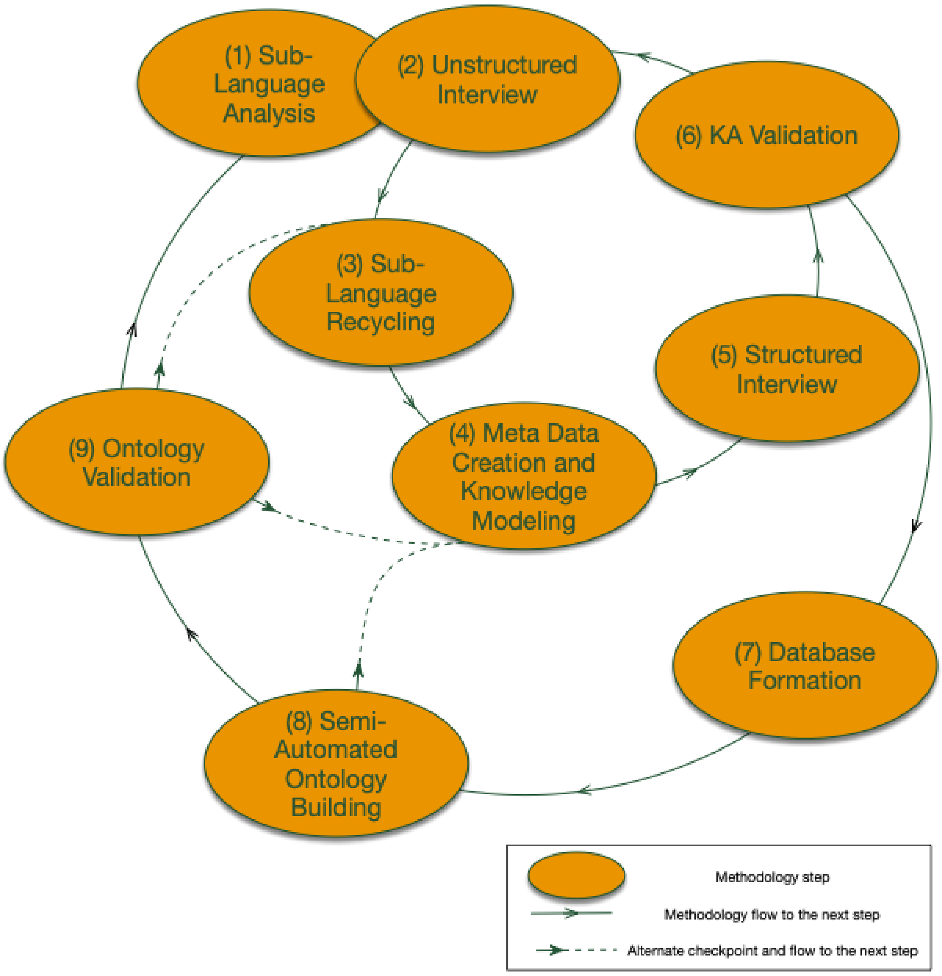

Fig. 1. **Steps of KNARM**

study by Huang et al. (2023). Consequently, users remain vulnerable to both recycled false information from training data and the spontaneous generation of falsehoods by LLMs. In light of these challenges, robust validation of ontology is imperative to ensure the accuracy of model outputs. While LLMs showcase impressive capabilities, their outputs should undergo rigorous scrutiny by experts in addition to the automated scripts prior to utilization in critical decision-making or practical applications.

3.1. OLIVE Workflow

This project aims to develop a methodology for semi-automated knowledge graph building using LLM APIs of ChatGPT and Llama. We implemented our infrastructure using Python, Flask, and HTML/CSS with a database backend implemented in Neo4j. With this infrastructure, we aim to allow domain experts to use LLMs in building and improving existing knowledge graphs. Our infrastructure can compile data from structured and unstructured data sources to build prompts that generate knowledge graphs and visualize it using our simple user interface. The following sections explain the architecture and details of the OLIVE methodology and its components.

The KNARM methodology takes domain expert input as a significant part of its process since domain experts are the main source of truth for knowledge graphs. OLIVE workflow evolves KNARM using LLMs beginning with user interaction of inputting data to the system. This action, can be considered

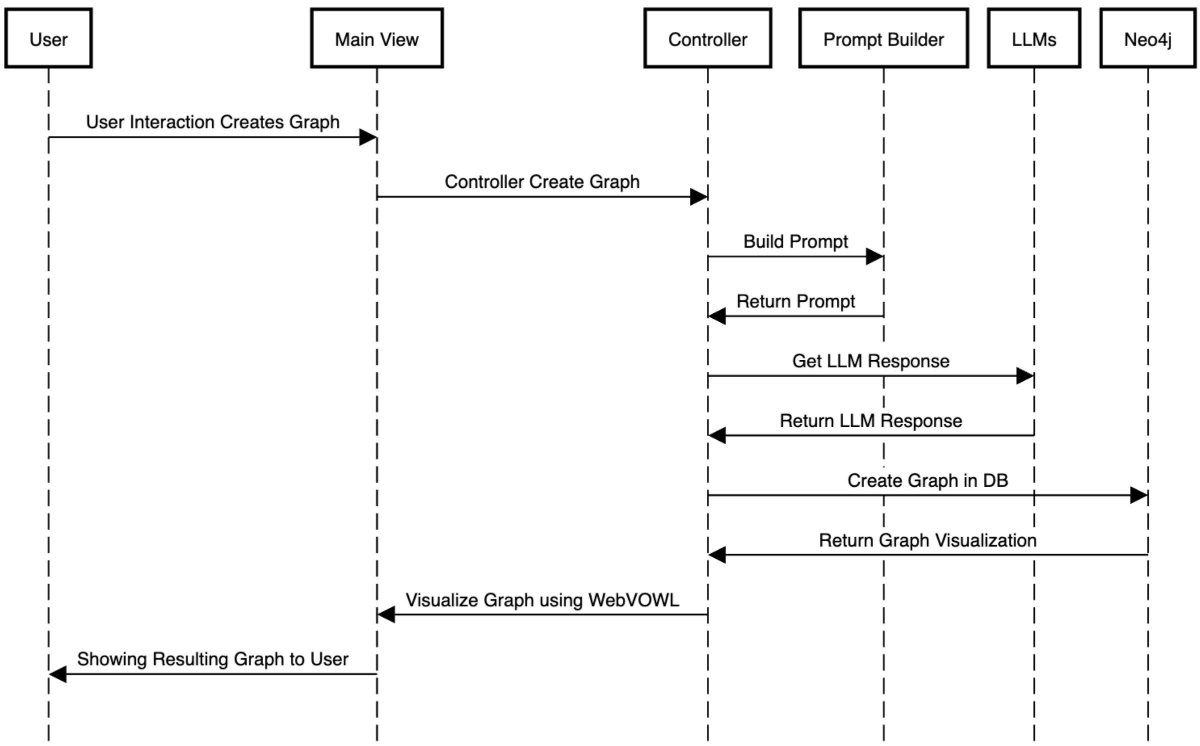

Fig. 2. **OLIVE Workflow** This sequence diagram illustrates the step-by-step interaction process within the OLIVE workflow, from user input through the Main View to the creation and visualization of the ontology graph via the Controller, Prompt Builder, ChatGPT, and Neo4j database.

replacement of the first two steps of KNARM, but because in this study, we used data from previous projects, this evolution in the workflow's user input stage was not changed significantly.

For OLIVE, we used structured data (i.e. ROBOT templates for previously implemented ontologies, which can be considered semi-schema like structures) and unstructured data from abstracts collected using a script that pulls paper abstracts using Semantic Scholar's API. This can be viewed as evolving KNARM by starting from its fourth step, but then following with semi-automated ontology building step (step eight) and using the agile workflow, ensuring the validation of the resulting KG, therefore allowing LLM exposure to all the steps.

The system also allows the users to input text using the simple UI implemented for this project, which may introduce different pieces of data to the LLMs and ultimately to the resulting KG. OLIVE currently does not support the third step of KNARM, which focuses on reusing existing ontologies. However, we can identify ontologies that can be reused as part of the verification and validation steps and use KNARM's agile design to introduce the ontologies that can be reused and recycled as a part of OLIVE.

The general workflow for OLIVE, as seen in Fig 2, begins with user interaction. First a user inputs a request for some kind of graph manipulation such as creating or extending. Then the controller takes the user input, processes it, and provides that necessary information to the prompt builder. The prompt builder will then construct a dedicated and specific prompt to provide to an LLM to generate the graph. Then these responses are gathered and processed to be uploaded to the Neo4j graph database. Finally, OLIVE will output a visualization for the user to be able to see the graph that was generated.

The OLIVE Workflow implementation brings ability to interact with user interface that empowers users to engage directly with the knowledge graph. This reduces the need for computer scientists to intervene in the graph's modification or update processes. As a result, this enhances the domain experts' capacity to help develop the ontology with maximum customization. Moreover, direct communication between domain experts and the knowledge graph via LLM help reduce the possiblities of misrepresenting concepts, relations, and entities, thereby enhancing the ease of ontology updating and review process.

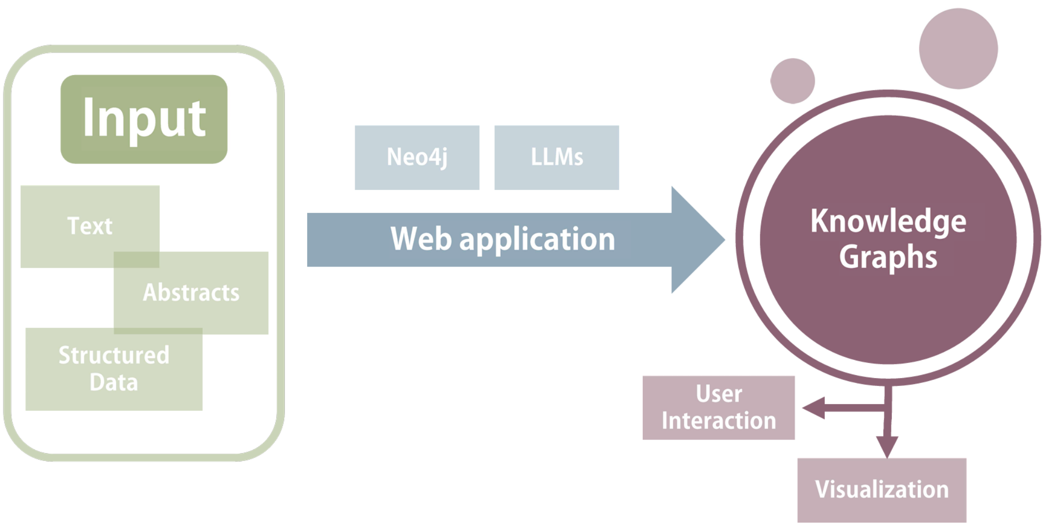

Fig. 3. **The overview of OLIVE**

3.2. Graph Database Interaction

With the refined LLM responses in hand, the Controller delegates the task of graph construction to the Neo4j database. This phase is critical, as it translates the abstract ontology components into a concrete graph representation. The system ensures fidelity in the depiction of conceptual relationships, laying the groundwork for a visualization that is both informative and reflective of the underlying data structure.

While description logic provides an abstract framework, the interaction with the graph database is necessary for practical implementation.

Graph databases like Neo4j are optimized for handling complex relationships and large datasets, allowing for faster data retrieval and manipulation. This efficiency is crucial for applications requiring rapid access to interconnected information. Additionally, representing the ontology in a graph format facilitates intuitive visualization through tools such as WebVOWL, making it easier for users to understand and explore relationships within the data.

Moreover, graph databases are designed to scale horizontally, accommodating growing volumes of data without compromising performance. Neo4j also supports sophisticated querying through languages like Cypher, enabling complex analysis that would be inefficient in a purely description logic framework. By ensuring fidelity in the depiction of conceptual relationships, the system bridges the gap between abstract ontological concepts and practical, usable data representations, enhancing overall utility and effectiveness.

The Neo4j database serves as the primary storage service for graphs inside OLIVE. Since Neo4j is a database designed for graph storage, it was quite simple to integrate into OLIVE. First it creates a node for every node in the ontology, ignoring duplicates. Then it adds in all of the relations between those nodes. It uses an id assigned to each node and edge to keep track of which graph they are a part of.

There are several advantages of Neo4j that Olive is able to leverage. One of the main features that is quite useful is its flexibility. Since OLIVE allows the user to iterate and change their graphs easily, it is essential that Neo4j enables this behavior with an easy to use query language to modify existing data. Along with that Neo4j also provides many of the same advantages that a typical database does, such as high performance and data reliability. Ontologies are stored as graph in NEO4J by using Cypher Query Language, it is based on Structured Query Language. Using Cypher query, patterns are used to identify specific graph structures within the data. Once a matching structure is identified or generated, Neo4j can utilize it for subsequent processing.

3.3. User Interaction

The OLIVE framework initiates with a user-centric approach, where the graph creation process is triggered through an intuitive Main View interface as shown in Fig 6. By clicking the "Create Graph" button, users can either input initial concepts and relations or allow the system to autonomously generate these starting points based on predefined queries, ref fig 5

This dual-mode entry caters to both novice and expert users, accommodating a diverse array of ontology construction scenarios and ensuring the system's adaptability to various knowledge domains (ref fig 6). By default, the ChatGPT API is used to process the prompt, but if users prefer to use Llama instead of ChatGPT, that option is also provided.

For Merge Graph function, users can merge two different graphs into one. The two graph names to be merged should be different. Otherwise, an error will be returned. Additionally, users must provide a new graph name that does not exist in the database. After fulfilling these conditions, users can click the Merge Graphs button. A visualize button will appear after merging the graphs, allowing users to visualize the new graph.

Load Graph section is used to view existing graphs in the database. There will be a dropdown menu to select which graph the user wants to view. After selecting the graph, users can click the Load Graph button to load it. Once the graph is successfully loaded, a visualize button will appear. Clicking this will allow users to view the graph in the WebVOWL UI refer fig. 7.

3.4. *Controller Activation*

Upon user engagement, the Controller activates the "Controller Create Graph" function, serving as the central orchestrator. It dynamically interprets user inputs, determining the precise sequence of operations required. This adaptive logic enhances the system's responsiveness and flexibility, tailoring the ontology construction process to the user's specifications and objectives. The controller is also responsible for a lot of the other operational steps within the program. Such as providing the prompt to the LLM model

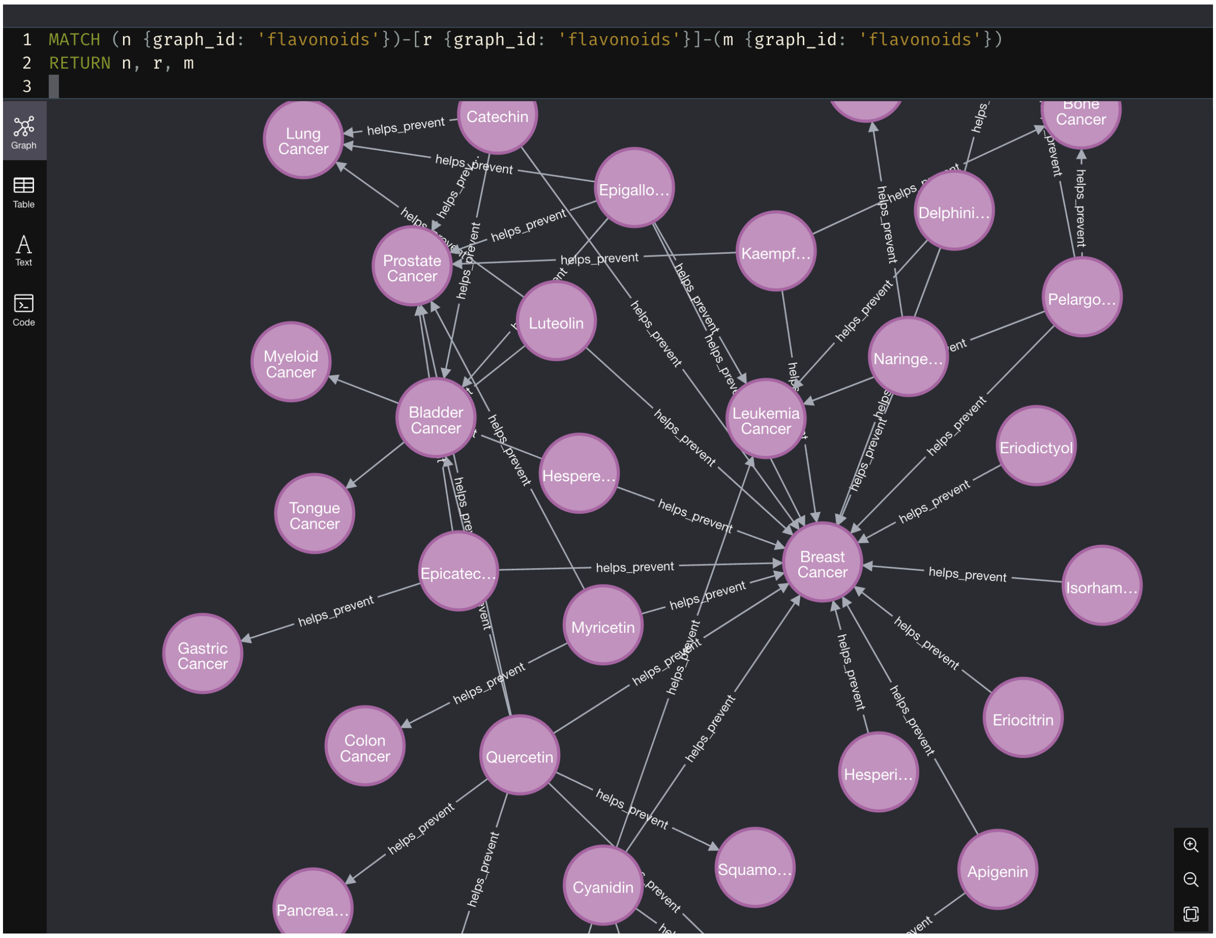

Fig. 4. **Neo4j graph database for storing OLIVE results**

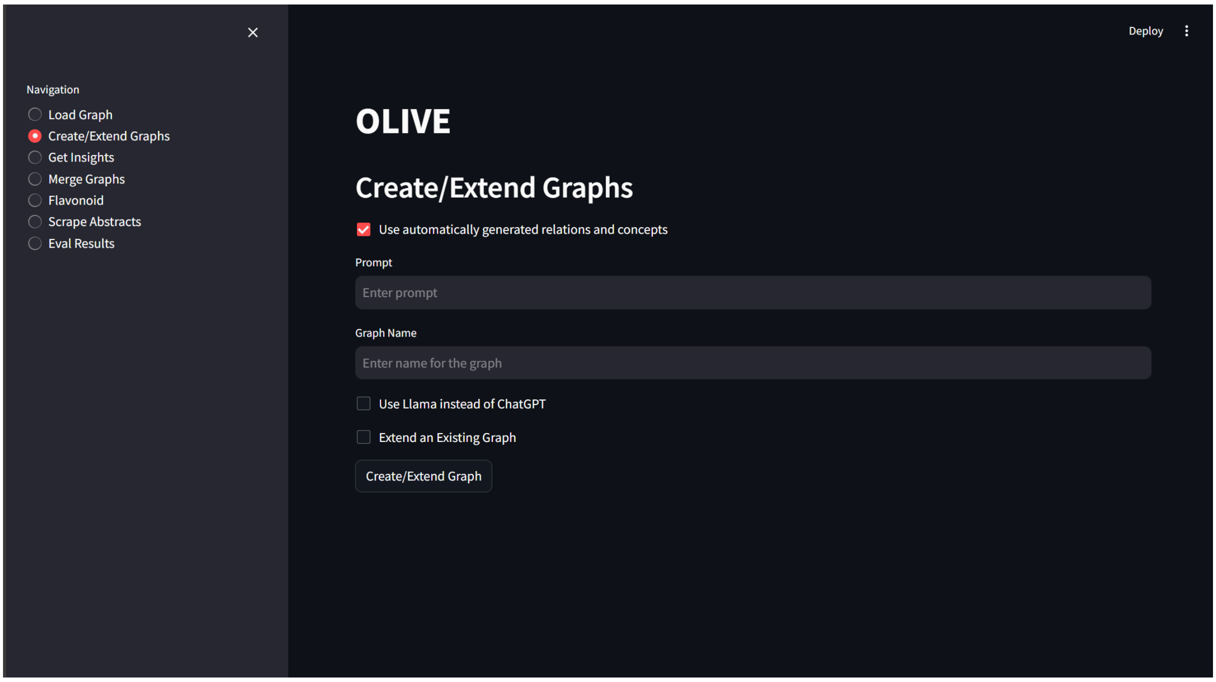

Fig. 5. **Automatic generation allows user to generate ontology on the basis of prompt**

class once it gets generated and also getting the response from the LLM and providing it to the Neo4j model class. Furthermore, is serves as the primary method of validating and processing user input.

3.5. *Prompt Construction*

Interfacing seamlessly with the Prompt Builder, the Controller leverages this subsystem to craft domain-specific prompts that are designed to elicit detailed ontology components from the LLMs. The Prompt Builder utilizes a sophisticated set of heuristics and syntactic rules, ensuring that each prompt is not only contextually relevant but also optimized for maximum efficacy in the data extraction phase of ontology construction. Specifically, the system generates prompts based on the availability of predefined concepts and relations. For example, it tailors prompts to include only provided concepts or relations, or both, depending on the input. This approach improves the quality and relevance of the prompts, guiding the LLMs to generate accurate ontology components efficiently.

Rather than claiming true optimization, which is often computationally infeasible, the system focuses on iterative refinement and empirical testing of the prompts. By adjusting the prompt structure based on the input (whether it includes concepts, relations, both, or neither), the Prompt Builder ensures that each prompt is appropriately tailored to the task at hand. This process enhances the effectiveness of data extraction and ontology construction, providing a practical and efficient means to leverage LLM capabilities.

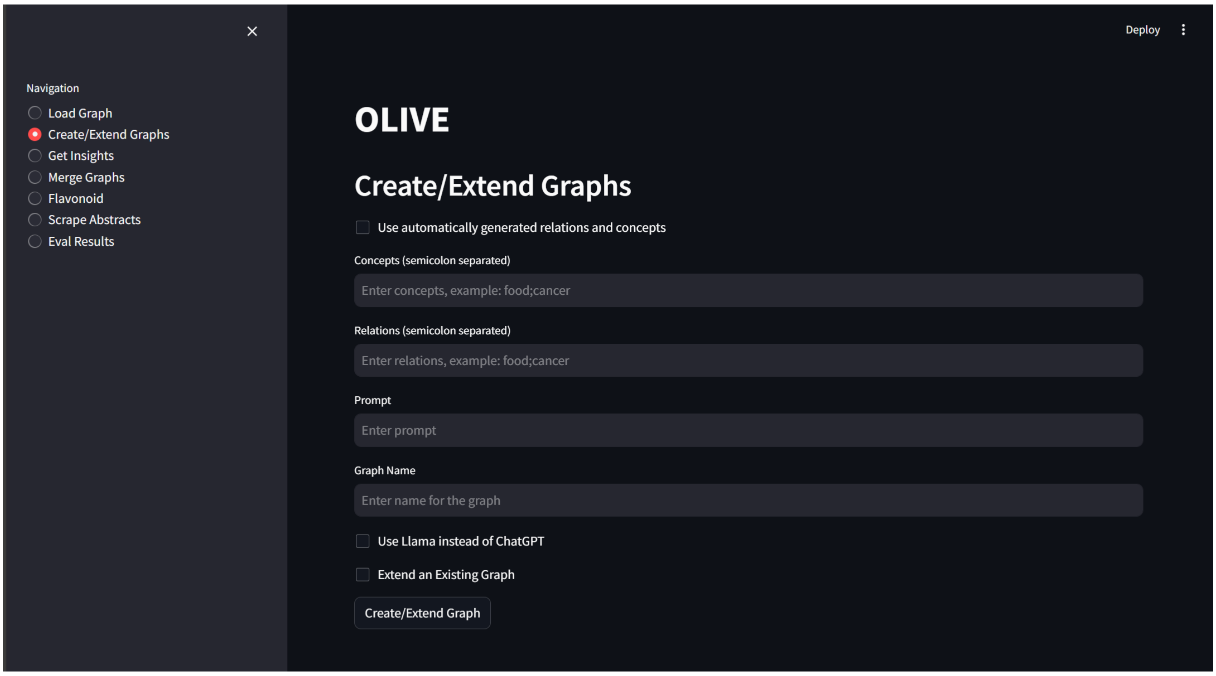

Fig. 6. **Users can create ontology by giving prompt,concepts and relations**

3.6. LLM Response Retrieval

The prompts, once constructed, are dispatched to the LLMs for processing and interpretation. We primarily use ChatGPT by OpenAI (2020) and LLama by Touvron et al. (2023), leveraging their distinct advantages. ChatGPT is widely recognized for its advanced language understanding and conversational capabilities, making it a strong choice for generating detailed and contextually accurate ontology components. LLama, on the other hand, is a free and open-source model, providing a cost-effective alternative with easy access to an API key.

In our system, users have the flexibility to select either LLM through a simple checkbox in the UI, allowing them to toggle between ChatGPT and LLama based on their specific needs and preferences. This choice ensures that users can leverage the strengths of both models. The responses, containing various potential ontology components, are then subjected to the validation steps of OLIVE. We employ an iterative approach to refine the protocol, enhancing the precision of subsequent prompts and adeptly managing any ambiguities or gaps in the initial LLM outputs.

3.7. Visualization

OLIVE includes an interactive visualization component that enables users to engage with the ontology visually. To achieve this, OLIVE leverages WebVOWL, a web application by Lohmann et al. (2015), which allows users to interactively visualize ontologies. WebVOWL provides a user-friendly platform for ontology visualization, enhancing the accessibility and comprehension of complex ontology structures. We chose to use WebVOWL to facilitate the validation process for domain experts, as it does not require any prior knowledge or experience with prominent ontology-building applications such as Protégé.

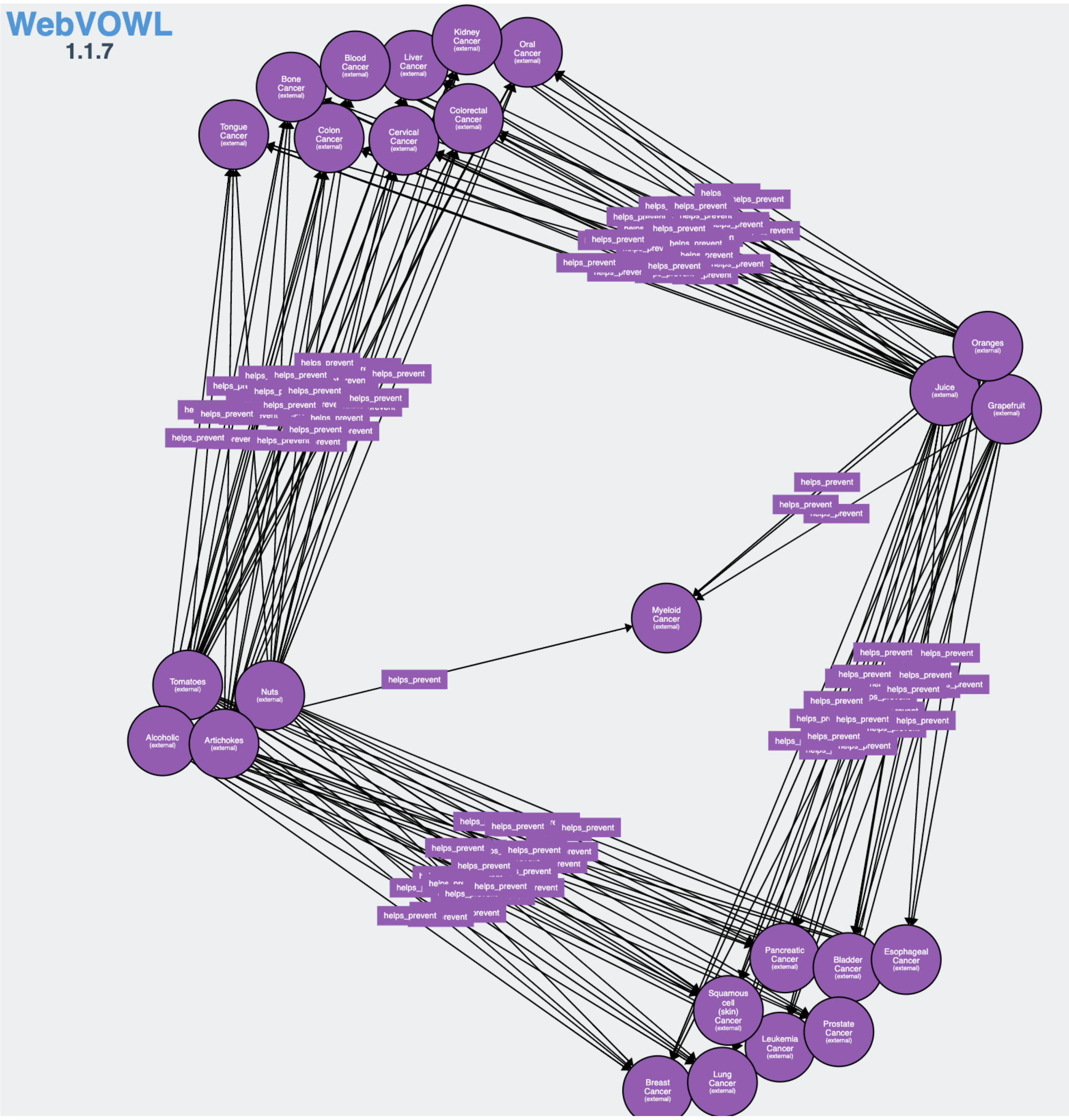

Fig. 7. **Visulatization of ontology using webVOWL**

3.8. Iterative Enhancement

OLIVE is an agile workflow, allowing users to continuously refine the ontology through a series of interactions at every step. This dynamic process is able to allow for the extension of the graph via additional prompts and LLM responses, thereby incrementally improving the ontology's accuracy and completeness. Refinements may include the integration of new concepts, the clarification of existing relationships, or the correction of any discrepancies, ensuring that the ontology remains a true and current representation of the domain knowledge.

As we discussed, developing and communicating with ontologies and knowledge graphs have a unique set of challenges due to their complex nature, for instance, the technical knowledge required to interact with them and the issues of ambiguity and interpretation. However, these complexities are offset by the significant advantages that ontologies offer over traditional relational databases. Ontologies operate at a higher level of abstraction, defining the domain in a more flexible and semantically rich manner. This higher level of abstraction makes ontologies particularly well-suited for automation using LLMs, as they can more effectively capture and reason about the conceptual relationships within a domain. While relational databases focus on table design and data storage, ontologies enable more sophisticated analysis and integration of knowledge, ultimately providing a more powerful tool for understanding and leveraging complex data. As these structures merge and expand, the task of managing and navigating among them becomes increasingly demanding. However, methodologies such as KNARM and OLIVE equipped with graphical interfaces, automated or semi-automated construction processes, and simplified querying tools, are making it more user-friendly and intuitive to interact with these complex structures.

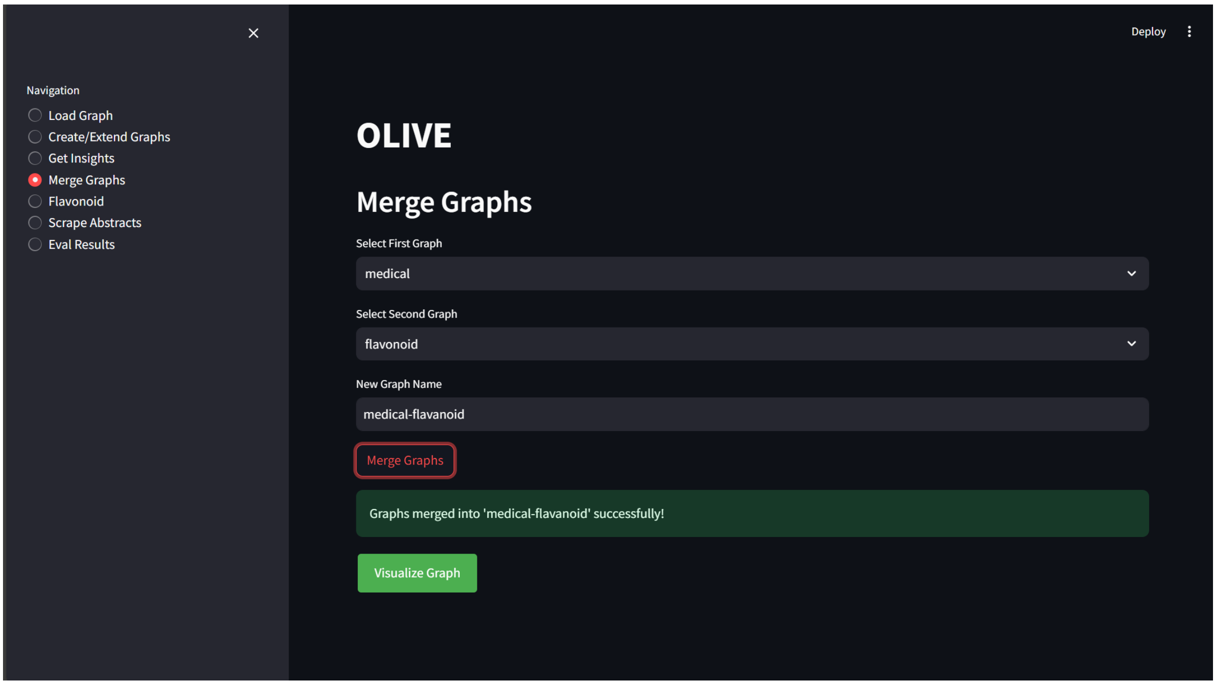

Fig. 8. **Merge graphs**

These developments are bridging the gap between users and these structures, facilitating more effective communication and understanding.

OLIVE addresses these challenges by offering a user-friendly interface that simplifies the interaction with ontology graphs. This approach not only makes the ontology more accessible to a broader audience but also encourages iterative enhancement, allowing users to continuously refine the ontology based on their interactions and insights. Through this iterative process, OLIVE aims to bridge the gap between the complexity of ontology graphs and the needs of end-users, making the knowledge encoded within these graphs more accessible and understandable.

4. Evaluation Metrics and Validation

Validation remains a critical aspect of our work, and we are actively pursuing various validation strategies. For example, one of the approaches involves identifying new entity additions and cross-referencing them with our existing set of vocabularies. By using ontologies developed by following the KNARM methodology, we are able to cross-check these new additions. Once the initial version of the ontology is completed, validation procedure is commenced.

At each stage, all inclusions and removals of entities and relationships undergo scrutiny. Any exclusions are flagged for further investigation, signaling a potential error. The role of domain expert is critical in distinguishing the new addition of entities, either as valid addition or result of hallucination.

In addition, we also used reasoner in Protégé by Gennari et al. (2003) (ref fig 9) and could not find any inconsistencies in the ontology made by using the OLIVE workflow. Another widely used technique is using Retrieval-Augmented Generation (RAG). In this study we did not use this approach in our architecture since this work does not focus on reducing hallucinations introduced by using LLMs, but rather focuses on creating a tool and a workflow for integrating LLMs to the workflow. Furthermore, while RAG does reduces the chances of hallucinations but does not eradicate it all along. Chances of introducing the hallucination by the LLM are still present even after using RAG models.

However, despite advancements and our semi-automated approach relying on our technical skills, we also believe that the input of domain experts remains indispensable. To facilitate their involvement, we have implemented visual representations, as detailed in the visualization section. We are also exploring enhancements to this feature in our forthcoming work.

As part of our evaluation approach we used the reasoners found in Protégé to check for logical inconsistencies. In our testing, no logical inconsistencies were found. Furthermore, we utilized the OOPS! (OntOlogy Pitfall Scanner!) tool by Poveda-Villalón et al. (2014) for additional validation. However, we still believe that current approaches and tools fall short in evaluating the ontologies and knowledge graphs built or altered using the LLMs with the OLIVE workflow. However, generation of evaluation methods and tools is a different research question and is not in the scope of this paper.

5. Conclusions and Discussion

As discussed above, ontology construction has been a manual and time-consuming process, relying heavily on domain experts to delineate categories, properties, and relationships between concepts or entities across various domains. This approach, while effective in certain contexts, is often limited by the availability and expertise of the domain experts and can be challenging to scale across large or rapidly evolving domains. Moreover, the manual nature of this process can lead to inconsistencies and errors, affecting the quality and reliability of the constructed ontology.

The utilization of methodologies like KNARM by Küçük McGinty et al. (2019) aims at addressing significant bottlenecks in ontology engineering, as it harnesses both structured and unstructured data from domain experts or existing ontologies. Through the combination of this diverse data using tools such as Robot, KNARM facilitates the creation of robust ontologies that encapsulate domain knowledge comprehensively. However, it's essential to recognize that this semi-automatic approach to knowledge graph formation comes with its own set of challenges. The intricate process of data integration and ontology construction demands considerable time and effort, often requiring extensive manual intervention to ensure accuracy and consistency. Consequently, there is a growing recognition of the need to explore more automated approaches to ontology development aimed at streamlining the process and reducing resource overheads.

As technology continues to evolve, the OLIVE methodology enhances KNARM and manages the

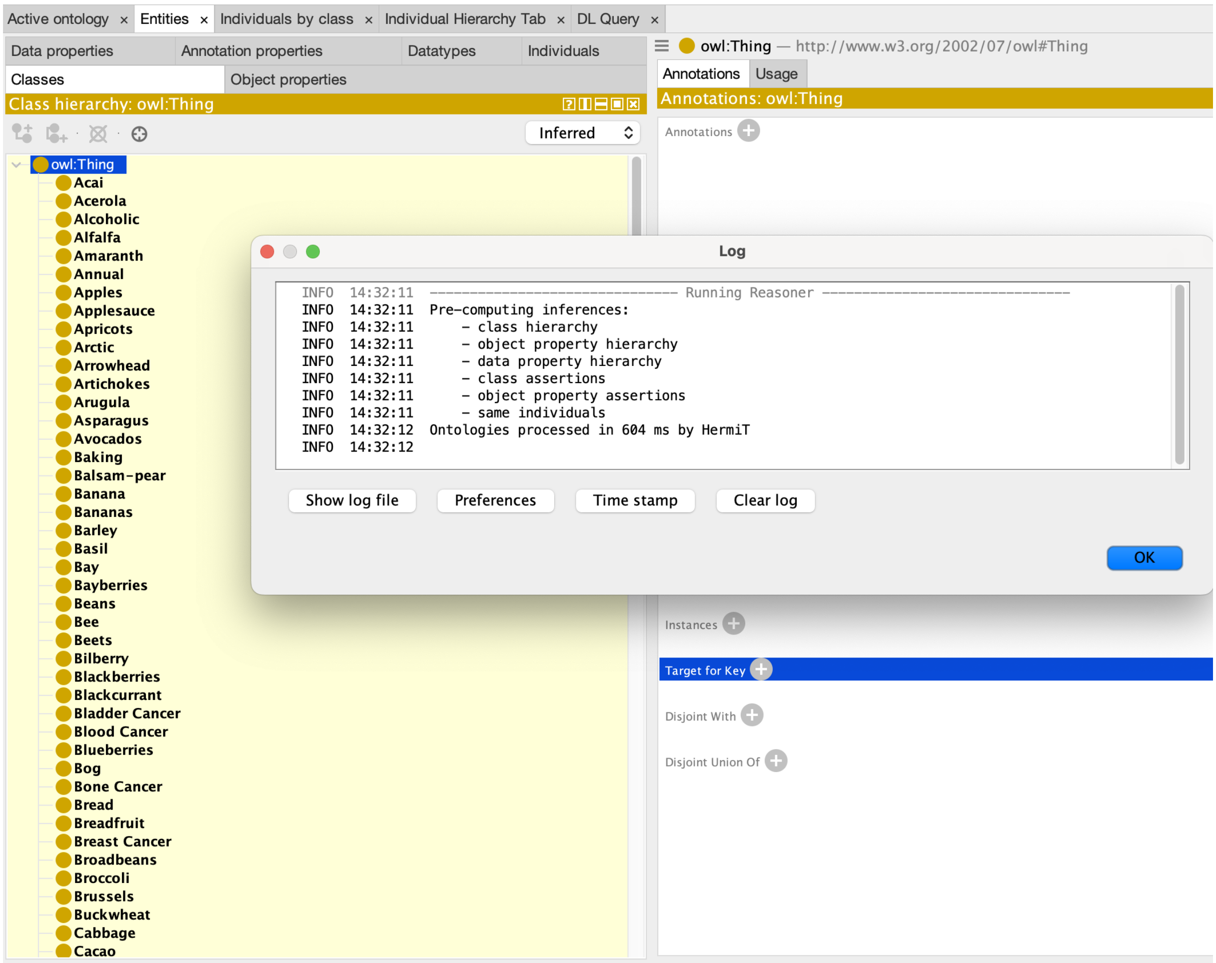

Fig. 9. **Reasoner results**

growing demand for automation. Despite challenges such as validation and data size management posed by increased automation through Large Language Models (LLMs), the development of OLIVE methodology may bring numerous advantages, simplifying processes and making knowledge graph work more accessible to all. By systematically leveraging Large Language Models (LLMs) for ontology construction, OLIVE can help address the challenges associated with traditional methods, offering a more efficient, scalable, and accessible approach to ontology development. As depicted in Fig 2, the workflow of OLIVE allows user interaction for inputting data and integrates the computational models explained above to streamline the process of ontology construction. This approach involves leveraging prompts to interact with the LLMs, initiating the ontology extraction process via a user-friendly interface. Upon receiving responses from the LLMs, the information is analyzed and translated into coherent relations, thereby facilitating the development of the ontology.

Our tests using different datasets created notable successes that exceeded our expectations in ontology building by harnessing the capabilities of Large Language Models (LLMs) through a custom method. We used a few use cases, a few were based on everyday subjects such as the Harry Potter book series and characters in the story as well as story lines. A more through use case was based on our dataset that uses USDA's food and nutrition information focusing on flavanoid content of foods and their connections to different cancer types. The details of this work is currently in review as another paper Küçük McGinty.

However, the challenges of validation remain a big concern for all the semi-automated or automatic tools of ontology development. With KNARM and OLIVE containing dedicated steps for validation of the resulting knowledge graphs, we are continuously building tools and systems dedicated to overcoming these obstacles and maximizing the potential of automation in ontology engineering. By continuing to push the boundaries of innovation and adaptation, OLIVE methodology holds promise for revolutionizing knowledge graph creation and management in the digital age.

6. Future Work

As we look to advance in the realm of ontology development and validation using Large Language Models (LLMs), several critical areas of focus emerge. Building on the human-centric traditional approaches that require substantial domain expertise, our proposed OLIVE workflow aims to leverage LLMs to simplify and enhance the ontology development process. However, as we evolved KNARM and transformed it into our new OLIVE workflow, we observed several challenges. The current constraints on token size within LLMs can restrict the depth and breadth of information processed in a single query, potentially limiting the comprehensiveness of ontology development. Although LLMs are designed to be highly general, this can sometimes be a drawback, as they may lack the specialized knowledge necessary for tasks within specific domains of ontology development. We would like to emphasize that currently the fine-tuning of LLMs used with KGs depends on specific use cases and application needs.

6.1. Token Size Limitation

The primary constraint faced by LLMs is their inability to process sequences longer than 60,000 tokens. This limitation restricts the development of comprehensive ontologies, particularly those that require extensive data processing. At the time of this paper, some newer versions of the larger language model were released that allowed the processing of 1 million tokens. However, with limited resources and time, the output produced using such models could significantly improve the ontology formation. Future research should explore potential solutions to overcome this token size limitation. Techniques such as chunk processing, summarization, and leveraging newer models with higher token limits could be investigated. Additionally, ongoing research into managing longer sequences of data within the constraints of current LLMs is crucial.

6.2. *Broadening LLM Application*

Our application of LLMs in this work is currently confined to a dataset we created previously. As mentioned above, the data was incoming as structured data, which can be viewed as a schema-like structure and unstructured data based on abstracts pulled using Semantic Scholar. The dataset focuses on Flavanoid contents of select food products. This limited and previously created dataset may be limiting the testing of OLIVE workflow's versatility. In an attempt to overcome this, we used other testing data used in Zaitoun et al. (2023). In our future research, we aim to focus on adapting LLMs for various domains beyond our existing datasets. This could involve fine-tuning LLMs on domain-specific datasets or developing domain-agnostic models that can be applied to a wider range of ontology development tasks.

6.3. *Automating Validation*

Traditionally, the validation of ontologies has relied heavily on manual processes, predominantly involving the expertise of domain specialists. However, this conventional approach is notorious for its time-consuming and labor-intensive nature. To address this challenge, there is a growing recognition of the need to modernize this approach by integrating semi-automatic or fully automatic tools and methods into the validation process.

One such method involves the use of scripts to compare newly added axioms or entities with existing ones in the knowledge graph corpus. If discrepancies are found, indicating false information, these can be flagged warranting further review. Moreover, this approach serves to highlight any potential hallucinations generated by Large Language Models (LLMs). Additionally, the development of systems capable of generating natural language questions for ontology validation could further alleviate the dependence on domain experts, streamlining the validation process and enhancing its efficiency.

Fig. 10. **Reasoner results using OntOlogy Pitfall Scanner(OOPS)**

To enhance the robustness of the validation process, we have incorporated a reasoner within Protégé to check for logical inconsistencies. In our testing, no logical inconsistencies were found. Furthermore, we utilized the OOPS! (OntOlogy Pitfall Scanner!) tool by Poveda-Villalón et al. (2014) for additional validation. Our ontology generated using OLIVE was tested with OOPS!, and we identified several pitfalls as showing in fig10, including missing disjointness and inverse relationships not explicitly declared. We believe these issues arise because our prompts did not explicitly instruct the LLM to build disjointness or inverse relationships.

While considerable progress has been achieved in ontology validation, there exists a compelling need for further refinement and improvement of these methods. Current researches are underway, focusing on maintaining the accuracy of the validation process. In alignment, we are dedicated to the ongoing development of OLIVE (Ontology Learning with Integrated Vector Embeddings), constantly working on adding feature on validating ontologies. Despite the challenges inherent in achieving fully automated and efficient ontology validation, the potential benefits it offers to the community justify the continued pursuit of this goal.

Our future work will focus on refining our workflow and methodologies by revisiting agile methodologies to better integrate the domain experts' knowledge while promoting computational reuse of ontologies. We think that this approach may enhance the adaptability and sustainability of ontological frameworks. We also plan to develop enhanced interfaces, building on the interactive interface introduced in OLIVE. Our future iterations will aim to support more nuanced queries and deeper analytical capabilities, allowing users to explore and define complex relationships with greater precision. Furthermore, we will explore extending the application of LLM-based ontology tools across more varied fields, enhancing their versatility and utility in broader research and practical contexts.

Acknowledgement. McGinty, acknowledge partial support by the National Science Foundation under award 2333532, *Proto-OKN Theme 3: An Education Gateway for the Proto-OKN* and partial support by an Institutional Development Award (IDeA) from the National Institutes of Health under grant number P20 GM103418.

References

Babaei Giglou, H., D'Souza, J. & Auer, S. (2023). LLMs4OL: Large language models for ontology learning. In *International Semantic Web Conference* (pp. 408–427). Springer.

Caufield, J.H., Hegde, H., Emonet, V., Harris, N.L., Joachimiak, M.P., Matentzoglu, N., Kim, H., Moxon, S., Reese, J.T., Haendel, M.A., et al. (2024). Structured prompt interrogation and recursive extraction of semantics (SPIRES): A method for populating knowledge bases using zero-shot learning. *Bioinformatics*, btae104.

Fernández-López, M. & Gómez-Pérez, A. (2002). Overview and analysis of methodologies for building ontologies. *The knowledge engineering review*, *17*(2), 129–156.

Funk, M., Hosemann, S., Jung, J.C. & Lutz, C. (2023). Towards Ontology Construction with Language Models. *arXiv preprint arXiv:2309.09898*.

Gennari, J.H., Musen, M.A., Fergerson, R.W., Grosso, W.E., Crubézy, M., Eriksson, H., Noy, N.F. & Tu, S.W. (2003). The evolution of Protégé: an environment for knowledge-based systems development. *International Journal of Human-computer studies*, *58*(1), 89–123.

Huang, L., Yu, W., Ma, W., Zhong, W., Feng, Z., Wang, H., Chen, Q., Peng, W., Feng, X., Qin, B., et al. (2023). A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *arXiv preprint arXiv:2311.05232*.

Küçük McGinty Knowledge-graph integration of food products on the USDA's Food Data Central (FDC) using KARMA (Knowledge Acquisition and Representation Methodology (KNARM) and Its Application) for Connecting it to Global Food Systems' Component. *In preparation*. In preparation.

Küçük McGinty, H., Visser, U. & Schürer, S. (2019). How to Develop a Drug Target Ontology: KNowledge Acquisition and Representation Methodology (KNARM). *Bioinformatics and Drug Discovery*, 49–69.

Lin, Y., Mehta, S., Küçük-McGinty, H., Turner, J.P., Vidovic, D., Forlin, M., Koleti, A., Nguyen, D.-T., Jensen, L.J., Guha, R., et al. (2017). Drug target ontology to classify and integrate drug discovery data. *Journal of biomedical semantics*, *8*, 1–16.

Lohmann, S., Link, V., Marbach, E. & Negru, S. (2015). WebVOWL: Web-based visualization of ontologies. In *Knowledge Engineering and Knowledge Management: EKAW 2014 Satellite Events, VISUAL, EKM1, and ARCOE-Logic, Linköping, Sweden, November 24-28, 2014. Revised Selected Papers. 19* (pp. 154–158). Springer.

Meyer, L.-P., Stadler, C., Frey, J., Radtke, N., Junghanns, K., Meissner, R., Dziwis, G., Bulert, K. & Martin, M. (2023). Llm-assisted knowledge graph engineering: Experiments with chatgpt. In *Working conference on Artificial Intelligence Development for a Resilient and Sustainable Tomorrow* (pp. 103–115). Springer Fachmedien Wiesbaden Wiesbaden.

OpenAI (2020). ChatGPT. https://openai.com/research/chatgpt.

Pan, J.Z., Razniewski, S., Kalo, J.-C., Singhania, S., Chen, J., Dietze, S., Jabeen, H., Omeliyanenko, J., Zhang, W., Lissandrini, M., et al. (2023). Large language models and knowledge graphs: Opportunities and challenges. *arXiv preprint arXiv:2308.06374*.

Peroni, S. (2017). A simplified agile methodology for ontology development. In *OWL: Experiences and Directions–Reasoner Evaluation: 13th International Workshop, OWLED 2016, and 5th International Workshop, ORE 2016, Bologna, Italy, November 20, 2016, Revised Selected Papers 13* (pp. 55–69). Springer.

Pinto, H.S., Tempich, C. & Staab, S. (2009). Ontology engineering and evolution in a distributed world using DILIGENT. *Handbook on ontologies*, 153–176.

Poveda-Villalón, M., Gómez-Pérez, A. & Suárez-Figueroa, M.C. (2014). OOPS! (OntOlogy Pitfall Scanner!): An On-line Tool for Ontology Evaluation. *International Journal on Semantic Web and Information Systems (IJSWIS)*, *10*(2), 7–34.

Sequeda, J.F., Briggs, W.J., Miranker, D.P. & Heideman, W.P. (2019). A pay-as-you-go methodology to design and build enterprise knowledge graphs from relational databases. In *The Semantic Web–ISWC 2019: 18th International Semantic Web Conference, Auckland, New Zealand, October 26–30, 2019, Proceedings, Part II 18* (pp. 526–545). Springer.

Shimizu, C., Hammar, K. & Hitzler, P. (2023). Modular ontology modeling. *Semantic Web*, *14*(3), 459–489.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E. & Lample, G. (2023). LLaMA: Open and Efficient Foundation Language Models. *arXiv preprint arXiv:2302.13971*. https://arxiv.org/abs/2302.13971.

Trajanoska, M., Stojanov, R. & Trajanov, D. (2023). Enhancing knowledge graph construction using large language models. *arXiv preprint arXiv:2305.04676*.

Zaitoun, A., Sagi, T. & Hose, K. (2023). Automated Ontology Evaluation: Evaluating Coverage and Correctness using a Domain Corpus. In *Companion Proceedings of the ACM Web Conference 2023* (pp. 1127–1137).