"Semantic Rules from Text: Automating Common-Sense Knowledge Curation with LLM,s"

Muhammad Raza Naqvi a,*, Arkopaul Sarkar a, Emna Lakani a, Farhad Ameri b, Linda Elmhadhbi c and Mohammed Hedi Karray a

a Laboratoire Génie de Production (LGP) Université de Technologie Tarbes Occitanie Pyrénées (UTTOP), 47 Av. d’Azereix, 65000, Tarbes, France
E-mails: snaqvi@enit.fr, arkopaul.sarkar@uttop.fr, emna.lakani@uttop.fr, mohamed-hedi.karray@uttop.fr
b School Of Manufacturing Systems & Networks | Ira A. Fulton Schools of Engineering, Arizona State University, 6075 Innovation Way W, Mesa, AZ 85212 United States
E-mail: farhad.ameri@asu.edu
c INSA Lyon, Université Lumière Lyon 2, Université Claude Bernard Lyon 1, Université Jean Monnet Saint-Etienne, DISP UR4570, Villeurbanne, 69621, France
E-mail: linda.elmhadhbi@insa-lyon.fr

Abstract. Common Sense knowledge (CSK) is critical in improving artificial intelligence (AI) systems by enhancing their understanding, reasoning, and interaction with the human world, especially in planning and decision-making problems. To achieve the practical applicability of CSK, an extensive set of this latter should be available for the problem domain. Large Language Models (LLMs) have shown promise in quickly curating CSK for a particular domain. Modern LLMs engines, such as GPT, can translate CSK expressed in natural language to semantic rules using formal languages, e.g., Semantic Web Rule Language (SWRL) or Datalog rules. However, they fail to use standard vocabularies ISO-21838 while generating domain-specific semantic rules. This paper proposes a hybrid method that leverages LLMs to generate commonsense knowledge, which is then transformed into semantic rules based on standard vocabulary. By using these rules, we address the limitations of LLMs, ensuring the resulting ontologies are semantically rich and comply with established ontology engineering standards. This hybrid approach empowers LLMs to contribute to creating ontologies that are consistent, standardized, and semantically rich by adhering to the principles of formal logics. Finally, this paper proposes a template-based prompt-engineering technique and pre-defined mapping that leverages LLMs to generate commonsense knowledge and transform it into semantic rules based on standard vocabulary. By comparing the quality of the result with other automated approaches like ChatGPT3.5 and GPT4.0, we show that the proposed semantic rules based approach guarantees consistency and adherence to specific upper-level ontologies for expressing CSK as semantic rules.

Keywords: Large Language Models, Common sense knowledge, Knowledge Engineering

1. Introduction

In recent years, Large Language Models (LLMs) have come into the limelight. They not only transformed but also revolutionized many sectors by leaving a huge impact where they have been deployed, especially in educa-
tion and technology [1]. They have seen a huge transformation due to integrating these linguistic-based AI models by utilizing their capability to understand, genre, and interact with human language at an unprecedented scale [2]. LLMs have shown promise in aiding knowledge engineering tasks like building ontologies and knowledge graphs [3]. However, they falter when creating or modeling domain-specific ontologies using standardized vocabulary. Because LLMs excel at identifying textual patterns, they often miss the deeper meaning and formal semantics crucial for knowledge engineering [5][6][7].

In the era of Artificial Intelligence (AI), Common Sense knowledge (CSK) is a crucial part of AI systems that helps in making rational and explainable decisions as humans do [8]. CSK is an inevitable component of today’s AI-driven decision-making applications. When it involves sophisticated tasks and interactions with different domains, it is pivotal for AI systems to be equipped with this type of knowledge. An extensive domain-specific CSK is required for planning and decision-making process for any given domain. This nature of knowledge includes the implicit and frequently used understanding of the world that humans naturally have at their disposal [9]. Researchers in different domains are increasingly highlighting the importance of acquiring and integrating appropriate domain-specific CSK [10][11][12]. Utilizing an enormous amount of domain-specific CSK improves the AI system’s capacity to perform decision-making effectively and in an explainable way. It also permits it to adapt correctly to different scenarios [13][14]. It is then necessary to turn this information into semantic rules that can be used to improve knowledge graphs and ontologies.

This paper proposes a hybrid method that leverages LLMs to generate commonsense knowledge, which is then transformed into semantic rules based on standard vocabulary. By using these rules, we address the limitations of LLMs, ensuring the resulting ontologies are semantically rich and comply with established ontology engineering standards. This hybrid approach empowers LLMs to contribute to creating ontologies that are consistent, standardized, and semantically rich by adhering to the principles of formal logic. We argue that ontology modeling and ontology mapping are more than just a linguistic task because these models may provide information you can convert into formal languages OWL. This is because these models are good at certain linguistic tasks, such as relation extraction and entity recognition, which support ontology engineering. However, in contrast to ontology engineering, it requires many domain experts to view the definitions of terms, the hierarchy of relationships between them, and formal representations based on logical inference. Additionally, one of the keys to making ontologies FAIR and interoperable is to align ontologies with standard vocabulary, such as ISO-21838, that are closely linked to common sense knowledge.

This paper proposes a methodology based on CSK-driven semantic-rule-based pre-defined mapping to standard ontology. It argues for the importance of ontologies and the necessity of involving humans in the development process, specifically for capturing and incorporating CSK from LLMs. This approach allows the creation of classes and property relationships based on Standard vocabulary. When LLMs are asked to generate natural language statements based on CSK, these statements are then converted into first-order logic based on CSK patterns driven by standard vocabulary. This semantic rule-based method addresses the limitations of LLMs by ensuring that the ontologies produced are consistent, standardized, and semantically rich, adhering to the principles of formal logic.

The remainder of the paper is organized as follows: Section 2 provides a brief overview of how LLMs work, focusing on textual patterns, prompt engineering techniques, and the limitations of LLMs in knowledge engineering. This section also emphasizes the need for human involvement in the loop for future reference in the domain of knowledge engineering. Additionally, we discuss the role of pre-trained LLMs, existing approaches to utilizing LLMs for ontology modeling, the importance of having a standard vocabulary, and ontology development practices. In Section 3, we present our methodology, which is based on CSK-driven semantic rule and predefined mapping to standard ontology. Section 4 offers a technical overview, comparing the existing OPENAI GPT versions 3.5 and 4.0, along with a discussion of our implementation. The paper concludes in Section 5, where we discuss our work’s limitations and future directions.
2. Literature Review

The emergence of LLMs has kick-started a new era where these models have a huge amount of information contained in knowledge embeddings on different domains as a result of having a vast quantity of information that exists among it. LLMs can be a hugely powerful source for the generation of CSK spanning an extensive range of domains. The realization that these models developed and trained on a huge amount of textual datasets and cover a wide variety of knowledge from humans makes it very valuable for the task of capturing CSK [15]. The transition from produced CSK by LLMs, which lacks structure, semantics, and logical alignment about certain domains, to structured and semantic rules is a big challenge [16][17][18], especially where GPTs (Generative Pre-trained transformers) being utilized for automatic knowledge engineering [19] [20].

LLMs evolutionary landscape is now at the phase where the question is whether every particular industry and business is going to be equipped with these models, having challenges like ethical [21][22], trustfulness [23][24], FAIR data principles [25][26], but crucially, in case of knowledge engineering, especially ontologies, will LLMs automatically generate ontologies? OR will ontologists become obsolete? [27]. CSK is a subset of knowledge that is considered to be universally true [28]; also, it is crucial for the development of ontologies that accurately reflect real-world semantics[29]. When discussing CSK, the Cyc project aimed to develop an ontology containing common knowledge terms, facts, concepts, and rules. The project also focused on creating a system capable of communicating in English and learning from human interactions [30]. The goal has been achieved by LLMs that can interact with the user with natural language and answer user queries using different prompts but without creating an ontology, a goal of CYC.

Understanding the notion between fact and knowledge is pivotal to addressing this challenge of ontology development using LLMs [31]. "A fact is a statement that can be proven to be true or false," whereas knowledge, in the context of ontologies and knowledge engineering, encompasses a broader understanding that includes the interpretation and inference of facts within a certain domain. Knowledge is not about truthfulness but involves the structured organization and representation of information that can be used to infer new insights [32]. Also, ontologies are consistently validated by domain experts and evolving where LLMs validation upon the information they provided is under question [23][24].

2.1. The Evolution and Impact of LLMs in AI

LLMs mark a significant era in technological innovation [33] as this not only reshaped the whole landscape of (AI) [34] but embarked on a new era where a lot of focus in research and development is on LLMs along the evolution of generative AI [35], but we cannot neglect the evolution of natural language processing (NLP) has played the very critical role which enables machines to read, understand and make sense of human language and Machine Learning (ML) systems which allow due to its learning capabilities for the development of models that can predict an outcome based on data [28][36][37][38].

It becomes evident that LLMs like GPT showed a huge leap in the evolution of AI. LLMs are designed in a way to understand, interact, and generate at an unprecedented scale; because of their access to huge amounts of text data to learn text patterns, the structure of the language enables these models to perform different ranges of linguistics-based tasks with amazing efficiency [39]. Because of these AI-based models, there is currently an argument about whether these LLMs (like GPT) are as good at everyday tasks as humans are actually and what impacts they might have on society at large [40]. These days, it’s very simple to find examples that show LLMs are either very smart or vague in some other way, thanks to the word "hallucinations" [41]. Any way you look at it, they are very flexible and impressive when it comes to understanding human language and talking and responding in a manner that needs a great deal of background information. [42].
2.2. Challenges of Utilizing LLMs in Ontology Development

Despite the significant advancements brought about by LLMs, their application in ontology development/learning presents unique challenges due to a lack of knowledge modeling and a high degree of reasoning capabilities [43]. In the context of knowledge engineering, representations of a set of concepts within a domain and the relationships between those concepts. They are used to reason about the entities within that domain and to make inferences about the data. The development of ontologies involves the creation of terms and concepts aligning with a standard vocabulary and expressing rich formal semantics through formal axioms. The challenge arises from the inherent design of LLMs, which excel in statistical pattern recognition but struggle with understanding the deep semantic structures and formal logic principles essential for ontology creation [44] [45].

LLMs lack proficiency in developing ontologies due to their limited capacity to comprehend intricate and nuanced relationships and classifications [46]. LLMs can comprehend language and generate text by relying on statistical patterns. However, they lack the capacity to comprehend the fundamental semantic relationships and logical structures necessary for creating accurate and meaningful ontologies [47] [48]. This constraint highlights the significance of discovering innovative approaches to construct ontologies that can adequately address the requirements for formal logic and a substantial level of semantic complexity [49].

2.3. Ontology Development Practices and the Role of Pre-Trained LLMs

In terms of ontology use and why ontologies are being developed, especially reference ontologies where the goal is to provide standard, controlled vocabulary regarding a specific domain, for example, IOF Core Ontology contains notions found to be common across multiple manufacturing domains, and there are top-level ontologies [50] like Basic Formal ontology (BFO), Suggested Upper Merged Ontology (SUMO) and Open CYC as well that have general understanding of the terms and concepts and What are the most general classes of all classifications. An open-source reference ontology provides a human- and machine-readable definition of their vocabulary of terms. Interoperability between datasets that use the ontology’s terms in their data or metadata is a major use case for reference ontologies. This is an important use case for reference ontologies. These LLMs can provide different information based on prompts while interacting with humans through natural language. Many prompt techniques exist as we provide the summary of prompt engineering programming patterns by Schmidt et al. [51] in Tab.1.

Pre-trained Large language models such as OpenAI ChatGPT series and Google’s BERT have shown many capabilities in various natural language processing tasks as they are linguistic models, their primary task is content generation and interacting with natural language translation now they are also the center of attention for creating ontologies due to following capabilities and such as 1) Semantic Understanding; 2) Entity Recognition; 3) Relation Extraction; 4) Concept Generation; that due to having these following capabilities they can be utilized for ontology development Because they have been trained on a large amount of text data which enable them to capture the relationships between words and phrases like synonymy, hyponymy, hypernymy, and others due to their semantic understanding of these relationships [52].

They also showed promise on the named entity relation (NER) task of identifying entities such as locations, organizations, and persons and relation extraction, which involves identifying the relationship between entities that are mentioned in the text [53]. According to Grandi et al., LLMs can generate text based on prompts, which can be used to automatically generate the conceptual design [54].

2.4. Limitations of LLMs in Knowledge Engineering

In this section, we discuss the limitations of pre-trained LLMs in the context of ontology development. These pre-trained LLMs don’t provide controlled vocabularies. We found the following limitations in the literature that make them unsuitable as automatic tools for developing ontologies and make them incapable of creating ontologies automatically in our specific use case problem[54][55][56][57][58][59][60][61][62][63][64][65][66][67]:
### Table 1
Summary of LLM Prompting Techniques [51]

<table>
<thead>
<tr>
<th>Technique Name</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zero-Shot</td>
<td>Requires no examples; the model generates responses based solely on the prompt’s instructions.</td>
</tr>
<tr>
<td>Mem Prompt</td>
<td>Enhances context understanding by incorporating memory elements or references into the prompt.</td>
</tr>
<tr>
<td>Chain of Thoughts</td>
<td>Breaks down complex queries into simpler, intermediate steps for more accurate answers.</td>
</tr>
<tr>
<td>Self Ask</td>
<td>The model generates its questions and answers them to refine understanding or output.</td>
</tr>
<tr>
<td>Prompt Chaining</td>
<td>Sequentially links multiple prompts and responses to tackle complex tasks or reasoning.</td>
</tr>
<tr>
<td>Act</td>
<td>Involves simulating actions or decisions within the prompted scenario for dynamic response generation.</td>
</tr>
<tr>
<td>ReAct</td>
<td>Build on previous actions or responses to evolve the dialogue or decision-making process further.</td>
</tr>
<tr>
<td>Few Shots</td>
<td>Uses a small number of examples to guide the model’s understanding and response generation.</td>
</tr>
<tr>
<td>Inception</td>
<td>Embeds prompts within prompts to create layered or nested instructions for nuanced tasks.</td>
</tr>
<tr>
<td>Memetic Proxy</td>
<td>Utilizes culturally or contextually significant references to anchor or guide responses.</td>
</tr>
<tr>
<td>Self Consistency</td>
<td>Aims for coherent and consistent responses across multiple iterations or related prompts.</td>
</tr>
</tbody>
</table>

- Lack of Explicit Knowledge Representation
- Lack of logical consistency
- Semantic Ambiguity and inconsistent responses
- Domain Specificity
- Data Bias and Incompleteness
- Semantic Ambiguity
- Limited Multi-modal Understanding
- Scalability Issues

Pre-trained LLMs’ response to prompts engineering techniques is based on patterns in data rather than understanding explicit structures like ontologies. For example, while generating definitions or relationships between different terms and concepts, a pre-trained LLMs’ may produce inconsistent information because those concepts are also represented wrongly in their training data[68][69]. Since pre-trained LLMs’ are designed in a way that cannot detect any logical Consistency this can also happen while extracting knowledge for creating ontologies. This is a huge challenge to rely on LLM,s only for extracting knowledge from LLM,s for the ontology’s creation part [70][71]. Pre-trained LLMs models might not capture the depth of the knowledge required to model a particular domain ontology due to not having an in-depth understanding of the concepts and terms [72][73]. LLMs struggle with words with different meanings in different contexts, so it will lead to consistency while developing ontologies. For example, it’s difficult to comprehend the word Bank unless you provide a context of what it is about, either about bank of the river or a financial institution [74][75]. Pre-trained LLMs are trained in textual data and have very little capabilities when it comes to processing other than textual resources [76][77]. For example, they may determine a Visual art piece yea that a particular painting reflects or designs but can’t comprehend the technical information about an art piece an art love can express like an artist, the paint used, etc. Regarding using LLMs. in ontology development, tasks require many resources, especially data, to fine-tune these models. For example, a project that involves ontology modeling using fine-tuned LLMs will involve computational costs for a particular deployment, so
when we talk about using LLMs for ontology development, scalability is also an issue that needs to be considered
[78][79].

2.5. Comparison with Existing Pre-trained GPTs

LLMs’ response to prompts engineering techniques is based on patterns in data rather than understanding explicit
structures like ontologies. For example, while generating definitions or relationships between different terms and
concepts, an LLM may produce inconsistent information because those concepts are also represented wrongly in
their training data [80][81].

A significant aspect of this paper is highlighting the limitations of LLMs in generating ontologies without an
ontological background. LLMs can convert natural language statements into semantic rules (e.g., SPARQL, SWRL,
or Datalog), but they often invent predicates instead of reusing those from a standard vocabulary. This behavior
underscores the necessity for LLMs to generate ontologies based on text patterns they trained on or given by an
end user, which needs to be aligned with established standards, for example, Top-Level Ontologies (TLOs) like
the Basic Formal Ontology *(BFO) ISO/IEC 21838-2:2021 - Information technology and Mid-level Ontology Like
Industry Foundry Ontology 2 (IOF) in our example use case. We emphasize the fact that for some cases like Core
FOAF3 (Friend of a Friend ontology) OR Family ontology4, LLM can easily model your text and align with ontol-
ogy because it is very general in nature of meaning, linguistics terms, and labels, but when we need to model our
data with standards vocabulary like BFO because classes are more abstract, we showed a prompt use-case 5 that we
try with GPT 3.5 even when we want to model very abstract level terms using LLMs it doesn’t follow any standard
vocabulary even you mention follow the particular vocabulary it creates hypothetical relations because of its inner
designed capabilities to the model based on patterns in data rather than rich semantics that ontologies provide also
GPT has a Limitations to words in a single prompt which is a 4096 so we can provide our RDF or TTL to these
Models. When we try GPT 4.0 6 and can attach both ontologies, as GPT 4.0 allows you to attach files in prompt,
it improves the results, but it still fails to completely reuse the given ontologies. It is very important to note that
while we utilize a rule base approach an argument can be that it is better to give rules to LLM,s and ask to create
ontologies hence there is in-depth study have been done by Ning et al.[82] which shows that why LLM,s work better
with stories based prompt to interact with LLMs than rules. Zhang et al.[83] also empathizes the fact that LLM,s
works better story based prompt in terms of knowledge engineering rather than rules.

3. The Proposed Methodology

The proposed method addresses the significant challenge of translating CSk into the semantic rule, which is
retrieved from LLMs through a natural language statement and then converted into ontology classes, sub-classes,
instances, or relations between them while aligned with standard vocabulary. The primary goal is to bridge the gap
between the flexible nature of natural language and the structured, rigid requirements of ontologies necessary for
effective reasoning and data integration. To achieve this goal, a rule-based mechanism identifies relevant concepts
within CSK statements and systematically integrates them into ontology classes, sub-classes, instances, or relations
between them. This process allows for the systematic transformation of CSK into structured, semantic rules that
align with ontologies leveraging the owlready2 library for ontology manipulation.

1https://basic-formal-ontology.org
2https://oagi.org/pages/industrial-ontologies
3http://xmlns.com/foaf/spec/
4https://core.ac.uk/download/pdf/42955298.pdf
5https://chat.openai.com/share/44a3d1a6-66b2-407d-b6db-509f158a30f9
6https://chat.openai.com/share/86565869-7a00-47e8-9067-d6359f61c32c
To have a comprehensive presentation of our methodology, we have divided our work into three parts, as we utilize GPT. In the first part, we demonstrate how GPT can assist us in extracting common-sense knowledge. LLMs, especially those trained on large-scale corpus datasets, incorporate commonsense knowledge. As illustrated in Fig. 1, our approach is divided into three segments; first segment discusses the utilization of GPT models, specifically focusing on their training and internal mechanisms. It demonstrates that they are trained on large-scale corpora of text, which includes commonsense knowledge. Second segment is rule base approach to formalize CSK Statement into semantic rules adhering the principles of Knowledge engineering, and third implementation of the system on front-end application that user can interact and create their own semantic rules using CSK queries. Second part as explained step by step in Table 2, our use-case scenario focuses on creating semantic rules that can later utilize for knowledge engineering. We employ the OpenAI API, utilizing prompt engineering methods to achieve specific pattern-based responses. Among various prompt engineering techniques, we use the chain of thought method. This approach is chosen because it enables us to elicit a specific type of response from ChatGPT. After obtaining the commonsense knowledge (CSK) patterned response, we convert it into singular natural language statements. These statements are then transformed into rules. Once we have CSK-based natural language rules, we use OWL2Ready for ontology manipulation. This manipulation allows us to align our natural language rules semantically with standard vocabularies such as IOF and BFO.

<table>
<thead>
<tr>
<th>#</th>
<th>Step</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Chain of Thought (CoT) Prompt Engineering Method</td>
<td>A technique that structures user Commonsense queries in a way that guides the GPT model to better understand and process the question and response with pattern based on give structure.</td>
</tr>
<tr>
<td>2.2</td>
<td>Generative Pre-trained Transformers (GPT)</td>
<td>AI models that use extensive training to generate responses that mimic human writing or speaking as shown in upper block of Fig. 1.</td>
</tr>
<tr>
<td>2.3</td>
<td>Pattern-Based Response</td>
<td>The generated response includes patterns that reflect commonsense knowledge, providing insights into human reasoning.</td>
</tr>
<tr>
<td>2.4</td>
<td>CSK Patterns</td>
<td>Identifiable elements within responses that showcase fundamental human understanding and knowledge.</td>
</tr>
<tr>
<td>2.5</td>
<td>CSK Patterns based NL Statements</td>
<td>Transformation of these patterns into clear, coherent statements in natural language.</td>
</tr>
<tr>
<td>2.6</td>
<td>CSK Patterns based NL Rules</td>
<td>Refining the natural language statements into structured rules, leveraging the identified commonsense knowledge.</td>
</tr>
<tr>
<td>2.7</td>
<td>Standard Vocabulary</td>
<td>Utilizing a predefined set of terms and definitions to ensure the rules are clearly understood and universally applicable.</td>
</tr>
<tr>
<td>2.8</td>
<td>CSK Patterns based Semantic Rules</td>
<td>The final set of rules, semantically enriched with the standard vocabulary to be meaningful and contextually appropriate.</td>
</tr>
</tbody>
</table>

Table 2
Simplified Workflow of purposed method

Fig. 1. High-Level Architecture Process Flow of Implementation

In the third part, we implemented our system on front end application which can publicly accessed by user while creating a user interface to create semantic rules using our model.

### 3.1. Formalizing Semantic Rule Specialization using Common Sense Knowledge (CSK)

In the following subsection, we will deep dive into the detail of Formalizing Semantic Rule Specialization using Common Sense Knowledge (CSK)

**Definition 1: Rule Template**

In logic, a Rule Template (RT) is like a formula where you fill in the blanks. The formula isn’t complete on its own because some parts are left blank. You can fill in these blanks with specific information later on, which will turn the template into a rule that can be used in certain situations. A Rule Template, *RT*, is defined as a logical expression containing placeholders that represent general classes or relationships. These placeholders are meant to be replaced with specific classes or instances to create a concrete Rule. The *x* and *y* are place holders fro specific processes.

**Example:**

\[ \text{process}(x) \rightarrow \exists y \text{process}(y) \land \text{precedes}(x,y) \]  

**Definition 2: Common Sense Knowledge (CSK)**

A CSK is an NL statement that provides specific Knowledge about a particular case, like classes or instances regarding Painting extracted from LLM using the Chain of thought prompt engineering method. The CSK serves as the source from which specific information is extracted to replace placeholders in the Rule Template.

**Example:** “The result of painting process is a painted object.”

**Example:** “After painting process, you do drying process.”

**Example:** “The drying process involves a dryer machine.”

**Definition 3: Concrete Rule**

A Concrete Rule, *CR*, is derived from a Rule Template by replacing its placeholders with specific classes or instances extracted from a CSK.

**Function: SpecializeRule**

Formally defined, the function SpecializeRule is:

\[
\text{SpecializeRule} : RT \times CSK \rightarrow CR
\]

**Process:**

1. **Input:** A Rule Template, *RT*, and Common sense Knowledge, *CSK*.
2. **Extraction:** Identify and extract specific classes or instances from the *CSK*.
3. **Substitution:** Systematically replace the placeholders in *RT* with the extracted classes/instances.
4. **Output:** Return a Concrete Rule, *CR*.

**Rule 1**

Given the Rule Template, *RT* based on Standard vocabulary classes and property relations from BFO and IOF:

\[
\text{IOF:MaterialProduct}(x) \rightarrow \exists y \text{BFO:process}(y) \land \text{IOF:isOutputof}(x,y)
\]

CSK: “The result of painting is a painted object.”

Applying **SpecializeRule**:

\[
painted\text{object}(x) \rightarrow \exists y \text{painting}(y) \land \text{IOF:isOutputof}(x,y)
\]
**Rule 2**

Given the Rule Template, $RT$:

$$BFO:\text{process}(x) \rightarrow \exists y BFO:\text{process}(y) \land BFO:\text{precedes}(x, y)$$  \[5\]

CSK: “After painting, you should dry.”

Applying SpecializeRule:

$$\text{drying}(x) \rightarrow \exists y \text{painting}(y) \land BFO:\text{precedes}(x, y)$$  \[6\]

**Rule 3**

Given the Rule Template, $RT$:

$$BFO:\text{process}(x) \rightarrow \exists y IOF:\text{machine}(y) \land BFO:\text{participates in at some time}(y, x)$$  \[7\]

CSK: “The drying process involves a dryer machine.”

Applying SpecializeRule:

$$\text{drying}(x) \rightarrow \exists y \text{dryer}(y) \land BFO:\text{participates in at some time}(y, x)$$  \[8\]

**Explanation:** The function *SpecializeRule* refines a broad rule template according to CSK to yield a concrete rule suited for a particular context. In this instance, the rule template intends to associate a process with the machine involved in it. Utilizing the CSK that “The drying process involves a dryer machine,” the general rule template is adapted to denote that the drying process involves a dryer machine at some point in time.

### 3.2. Logical Axioms of Commonsense Knowledge-driven Semantic Rules Patterns

#### 3.2.1. Process Rule Axiom:

To capture the relationship between something being produced and the requirement of a process, we’ll first have to define or choose the appropriate relations from the ontology (BFO and IOF) that can connect a product (or output) with a process. The relations that are relevant to describe this kind of relationship are:

- **has output:** $y$ has output $x$ if $x$ is an instance of BFO: Continuant and $y$ is an instance of BFO: Process, such that the presence of $x$ at the end of $y$ is a necessary condition for the completion of $y$.
- **is output of:** $x$ is output of $y$ if $x$ is an instance of BFO: Continuant and $y$ is an instance of BFO: Process, such that the presence of $x$ at the end of $y$ is a necessary condition for the completion of $y$.

Using these relations, let’s formulate the axiom:

$$\text{product}(x) \rightarrow \exists y \text{ process}(y) \land \text{is output of}(x, y)$$  \[9\]

**Explanation in context of Domain and Range:**

- $\text{product}(x)$: Specifies that $x$ is a type of product or something that has been produced.
- $\rightarrow$: Represents the implication. It states that if something is a product, then what follows must also be true.
- $\exists y [\text{process}(y) \land \text{is output of}(y)]$: For any product $x$, there exists a process $y$ such that $x$ is the output of the process $y$.

This axiom captures the idea that for any given product or something produced, there’s a process that leads to its creation or production. It essentially formalizes the statement, “If something needs to be produced, some type of process is needed.”
Process Precedence Rule Axiom

Given that there’s a process denoted as \( x \), then there must exist another process \( y \) that occurs before \( x \). Symbolically, this relationship can be represented as:

\[
\text{process}(x) \rightarrow \exists y \left[ \text{process}(y) \land \text{precedes}(x, y) \right]
\]  

(10)

Explanation:

1. **For any given process** \( x \): This is about any process we are considering, which we are naming \( x \).
2. **There exists another process** \( y \) **that must occur before** \( x \): This means that before the process \( x \) can occur, there must be some other process, which we are naming \( y \), that happens first.
3. **This establishes a precedence relationship**: Here, “precedence” means the order in which things should occur. In this context, it means that process \( y \) should occur before process \( x \).
4. **process(x)**: This denotes \( x \) as a process.
5. **→ Again, this represents an implication. In logical terms, an implication (often represented as →) is a way of saying “if this, then that.” In our axiom, it’s saying, “if \( x \) is a process, then there must be some other process \( y \) that precedes it.”
6. \( \exists y \left[ \text{process}(y) \land \text{precedes}(x, y) \right] \): This is a formal way of saying there exists a process \( y \) such that \( x \) precedes \( y \).
   In other words, \( y \) must happen before \( x \).

Machine Requiremnt Axiom:

\[
\text{Manufacturing process}(x) \rightarrow \exists y \left[ \text{machine}(y) \land \text{participates in at some time}(y, x) \right]
\]  

(11)

Explanation: This axiom states that for any given process \( x \), there exists a machine \( y \) such that \( y \) participates in the execution of the process \( x \) at some time. This formulation is in line with the domain and range constraints provided. If ‘machine’ is an instance or subclass of ‘independent continuant’ (excluding ‘spatial region’), then it can participate in a process as per the ontology constraints.

**Relation**: participates in at some time

**Domain Constraints**:

- **specifically dependent continuant**: Entities that exist by depending on some other entities. For instance, a color (like the redness of an apple) specifically depends on the apple’s existence.
- **generically dependent continuant**: Entities that exist by depending on some entities, but not any particular entity specifically. A classic example might be a type of software that depends on being installed on some computer to exist but not on any specific computer.
- **independent continuant**: Entities that exist independently and are not specifically or generically dependent on other entities. However, they cannot be ‘spatial regions’. An example could be a physical object, like a machine.

**Range Constraints**: The entities that can act as the object (or endpoint) of the relation are: Given the domain constraint: If the term ‘machine’ falls under ‘independent continuant’ (and isn’t a ‘spatial region’), then our axiom from before is consistent with these constraints:

- **process**: Temporal entities or events that unfold or occur over time. Examples might include manufacturing, baking, or melting.
- **Domain**: ‘specifically dependent continuant’ or ‘generically dependent continuant’ or (‘independent continuant’ and (not (‘spatial region’)))

In summary, our axiom states that for any process, there’s a machine (that is a suitable type of continuant) that actively engages or takes part in that process at some point in time. This is in line with both the domain and range constraints provided.

Given that the range is “process,” the relation participates in at some time is suitable to link a machine (as long as it falls within the domain constraints) to a process.
Given the domain constraint: Domain: ‘specifically dependent continuant’ or ‘generically dependent continuant’ or (‘independent continuant’ and (not (’spatial region’)))

If the term ‘machine’ falls under ‘independent continuant’ (and isn’t a ’spatial region’), then our previous axiom is consistent with these constraints.

3.3. Technical Overview and Implementation

The core function update_ontology_with_class_and_instance is designed to either create a new class and instance within the ontology based on the CSK or retrieve them if they already exist. This function\(^8\) ensures that each new ontological element adheres to a predefined naming convention and structure, facilitating consistency and retrievability within the ontology. Parameters that have been used as ontology. The target ontology to be updated.\(\text{class name}\) The class name is derived from the CSK.\(\text{parent class iri}\) The IRI of the parent class ensures hierarchical organization within the ontology.\(\text{Process}\) The function defines a prefix (CSK) for all new elements to maintain name-space integrity.

It checks if the class already exists within the ontology; if not, it creates a new class under the specified parent class. An instance of the class is then created or retrieved, establishing a concrete representation of the CSK. The specialization of rule templates into concrete rules (CR) is a critical step in translating CSK into ontological statements. The process begins with identifying a suitable rule template based on keywords within the CSK statement.

The SpecializeRule function then extracts relevant entities and their relationships from the CSK, mapping them onto the selected template to generate a CR that is both semantically rich and aligned with the ontology’s structure.

The determine_rule_template function selects an appropriate rule template (RT) based on the presence of specific keywords in the CSK statement. The SpecializeRule function parses the CSK, identifying and extracting entities and their relationships. These entities are used to fill placeholders in the RT, creating a CR that formalizes the CSK within the ontology.

Example: Modeling Painting as CSK To illustrate the method, consider the CSK NL statement: "The result of painting is a painted object." This statement is processed as follows:

- **Rule Template Selection**: Based on the keyword "result," the appropriate RT that models the outcomes of processes is chosen.
- **Entity Identification and Extraction**: The process ("painting") and the outcome ("painted object") are identified from the CSK.
- **Ontology Class and Instance Creation**: The update_ontology_with_class_and_instance function is invoked to create or retrieve the "painting" class, as well as the "painted object" class under BFO:Process class within the ontology.
- **Concrete Rule Generation**: The extracted entities are mapped onto the selected RT, generating a CR that formally represents the CSK within the ontology.

The following figures show how the front-end user can interact with the method as shown in Fig. 2 this can also be accessed publicly using this link http://chaikmat-anr.uttop.fr/csk/.

In Figure 3, 4 is the user interface where user can type his own commonsense knowledge NL rule according RT to automatically generate semantic rule as shown in Fig. 5.

4. Application, limitations, and future work

Our methodology employs standard vocabularies (BFO/IOF), making it directly applicable to any other data modeled by the same standards. BFO is a domain-neutral top-level ontology, and IOF Core is a mid-level reference ontology, enabling their application across a wide range of domains. This universality means that the rules can be applied to data from numerous domains and applications, highlighting the methodology’s extensive flexibility.

\(^8\)https://github.com/MRNaqvi/Common-Sense-Knowledge-Driven-SemanticRule-Base-Ontology-Mapping
Furthermore, the methodology is not ontology-specific. While the generic rules are designed for BFO/IOF, adopting a different set of ontologies requires only modifications to these rules with vocabularies from the chosen ontologies. This flexibility demonstrates the methodology’s adaptability to various ontological frameworks.

However, the methodology is not without its limitations. The rule base template method necessitates manual efforts to formulate the generic rules, and the template approach constrains the types of commonsense knowledge (CSK) that can be curated. Additionally, the process of creating rules from natural language (NL) statements, especially those extracted from large language models (LLMs), presents challenges. There is no straightforward way to validate inputs from users or LLMs. For example, incorrect classification of inputs such as cities being classified as processes illustrates the need for enhanced validation mechanisms. To address this, we have implemented a pre-defined chain of thought prompting technique to elicit patterned responses from the GPT model by preserving the state of open AI API after insuring the Chain of thought prompt engineering method to stick with same pattern response.
To further enhance the reliability and robustness of our framework, we are developing a validation framework for generating semantic rules from CSK. Incorporating a knowledge graph into this framework allows for rule verification against a comprehensive database of interconnected entities and relationships. A key feature of this validation framework is semantic consistency checks, which ensure that references to entities and processes are contextually appropriate, thereby eliminating invalid responses.

Looking ahead, the potential applications of this methodology are vast and varied. Future work could explore its applicability in different fields, assess the precision and utility of the created ontologies in real-world scenarios, and investigate how this approach could be integrated with other AI and machine learning techniques to refine the ontology creation process. Additionally, understanding the limitations of this methodology, especially in handling ambiguous or domain-specific CSK, could provide valuable insights and lead to further methodological advancements.

5. Conclusion

This paper introduced an innovative approach for converting common sense knowledge (CSK) obtained from large language models (LLMs) and users queries into structured semantic rules that correspond with standard ontologies. We have shown a systematic method to connect natural language’s flexibility with the structured require-
ments of ontologies for effective reasoning and data integration using a rule-based mechanism and the owlready2 library.

Our work can potentially revolutionize the development of ontology by making it more efficient and scalable. Historically, creating ontology has been a time-consuming task that demands substantial domain knowledge and manual work. Our method automates the conversion of CSK into ontology components, reducing manual work, enhancing scalability, and potentially improving the quality and usefulness of the resulting ontologies. This method enhances knowledge engineering and creates opportunities for incorporating AI-generated content into semantic web technologies.

Our work focuses on the important task of aligning dynamically generated knowledge with established ontological standards at the intersection of natural language processing and ontology development. Our methodology ensures that the created ontologies can work together smoothly, making them useful for reasoning and integrating data in different knowledge areas.

Acknowledgements

This work is performed within the CHAIKMAT project funded by the French National Research Agency (ANR) under grant agreement "ANR-21-CE10-0004-01"

References


[30] Schmidt, Douglas C., Jesse Spencer-Smith, Quchen Fu, and Jules White. "Towards a Catalog of Prompt Patterns to Enhance the Discipline of Prompt Engineering."


