Actively Learning: an LLM extension for the Exact Learner framework

Riccardo Squarcialupi
riccard.squarcialupi@studio.unibo.it

15 giugno 2024

Contents

Abstract

Active learning is a method where a learner attempts to learn some kind of knowledge by posing questions to a teacher. In computational learning theory, classically, the questions made by the learner are called *membership queries* and are answered with 'yes' or 'no' (or equivalently, with 'true' or 'false'). Here we consider that the teacher is a language model and we study the case where the knowledge is expressed as an ontology. To evaluate the approach, we present results showing the performance of GPT and other language models when answering whether concept inclusions on existing $\mathcal{EL}$ ontologies are 'true' or 'false'.

1 Introduction

Large language models (LLMs) have now amassed so much information and improved their question-answering abilities that we are willing to interact and learn from them. These prompts range from general knowledge queries, such as basic definitions and historical events, to more domain-specific questions, like scientific facts related to health and medicine. Recently, initial efforts have been made to automate knowledge extraction from LLMs [17], where the knowledge is extracted in the form of an *ontology* [8]. In the field of knowledge representation and reasoning, ontologies are the standard method for representing the relevant conceptual knowledge of a domain. The most popular formalism for specifying ontologies is given by *description logic* [5]. Ontologies have been extensively used to represent hierarchies and support data integration, information retrieval, and automated reasoning in life sciences [15]. Below, we provide an expression corresponding to a *concept inclusion* in description logic and its translation into natural language.

$\mathsf{Algae} \sqsubseteq \mathsf{Plant}$ Algae can be considered a subcategory of Plant.

What can we learn from LLMs? And, given that they can provide false information, is there an automated way to discover whether responses are incorrect or at least inconsistent? To explore these questions, this work investigates an active learning approach to learn *description logic ontologies* from LLMs. Our approach is based on Angluin's exact learning framework [3] from computational learning theory, where a learner attempts to learn target knowledge from a teacher by posing questions. This method, known as *active learning*, considers the LLM as the teacher and studies the case in which the knowledge to be learned is expressed as an ontology in the classical $\mathcal{EL}$ description logic [4]. $\mathcal{EL}$ ontologies allow for expressions involving conjunctions and existential quantification. The following example illustrates a concept inclusion in $\mathcal{EL}$ and its translation into controlled natural language.

$$\mathsf{Conifer} \sqsubseteq \mathsf{Gymnosperm} \sqcap \exists \mathsf{bears.Cone}$$

Conifers can be considered a subcategory of Gymnosperms that bear cones.

Exact learning of lightweight description logic ontologies has been studied for theoretical purposes [13, 12, 16]. One of the main challenges in employing such algorithms in practice is finding a teacher that can answer the questions posed by the algorithms. The only known implementation for exact learning description logic ontologies involves a synthetic teacher [7], created for testing using ontologies from the Oxford Ontology Repository [1]. With the advancement of LLMs, we can now investigate the extraction of description logic ontologies from LLMs within Angluin's active learning framework [2]. For mapping concept inclusions in description logic to controlled natural language, we use the Manchester

[1] https://www.cs.ox.ac.uk/isg/ontologies/

OWL Syntax [10] and our own parsing function, described in Section 3. We provide a non-trivial adaptation of the implementation for $\mathcal{EL}$ ontologies from the literature [7] that poses questions to LLMs, instead of using their synthetic teacher. To evaluate our approach, we present results showing the performance of GPT and other LLMs in determining whether concept inclusions created by an ontology engineer on prototypical $\mathcal{EL}$ ontologies are 'true' or 'false'.

2 Introduction to Description Logic

2.1 Overview

Description Logic (DL) [9] is a formal knowledge representation language used to describe and reason about the conceptualization of a domain. DL provides a set of constructs to represent knowledge in a structured and logical manner, allowing for precise and expressive modeling of concepts and their relationships. In this chapter, we will explore the fundamental concepts of DL, its syntax and semantics, reasoning mechanisms, and its applications in various domains.

2.1.1 Concepts

In DL, concepts represent sets of individuals that share common characteristics or properties. Concepts are used to describe the types or classes of objects in a domain. For example, in a medical ontology, concepts like *Person*, *Patient*, and *Doctor* can be defined to represent different types of individuals.

2.1.2 Roles

Roles (also known as properties or relations) define binary relationships between individuals or concepts. Roles are used to specify how individuals are related to each other. For instance, in a family ontology, roles such as *hasParent* and *hasChild* can be defined to represent parent-child relationships.

2.1.3 Axioms

Axioms are logical statements that define the relationships between concepts and roles in a domain. Axioms are used to specify constraints and rules that govern the domain. Common types of axioms include concept inclusion axioms (e.g., $A \sqsubseteq B$), role inclusion axioms (e.g., $R \sqsubseteq S$), and equality axioms (e.g., $A \equiv B$).

2.1.4 Syntax

The syntax of DL is defined by a set of symbols and rules for constructing valid expressions. DL ontologies are typically represented using a formal language based on first-order logic. The syntax includes symbols for concepts, roles, axioms, and logical connectives (e.g., conjunction, disjunction, negation).

2.1.5 Semantics

The semantics of DL define the meaning of DL expressions through interpretations and models. Interpretations provide meanings to the symbols in a DL ontology, while models fulfill the axioms and constraints outlined in the ontology. These semantics are grounded in set theory and model theory.

2.2 Reasoning in DL

2.2.1 Classification

Classification is the process of organizing individuals into classes or concepts based on their properties and relationships. DL reasoners use classification algorithms to determine the subsumption relationships between concepts and infer implicit knowledge from the ontology.

2.2.2 Query Answering

Query answering involves retrieving information from a DL ontology in response to user queries. DL reasoners can answer queries about the satisfiability of concepts, the consistency of the ontology, and the existence of individuals satisfying certain conditions.

2.3 Applications of DL

DL has applications in various domains, including ontology engineering, knowledge representation, semantic web technologies, biomedical informatics, and natural language processing. DL-based ontologies are used to model and reason about complex domains, facilitate data integration and interoperability, and support automated reasoning and decision-making systems.

3 The ExactLearner and Language Models

3.1 Active Learning

Active learning is a method where a learner attempts to learn some kind of knowledge by posing questions to a teacher. In computational learning theory, classically, the questions made by the learner are called *membership queries* and are answered with 'yes' or 'no' (or equivalently, with 'true' or 'false') [3].
What if the teacher does not directly have the entire ontology at its disposal but still it has knowledge about the domain?
This is the scenario when we substitute the omniscient teacher with an LLM. Membership queries do not pose additional issues except the ones presented in the section below. Instead, equivalence queries are problematic. It is not feasible, nor reasonable, to ask an LLM if the hypothesis ontology is equivalent to the target ontology. First, because the teacher does not directly have the target ontology at its disposal, and second, because the LLM could struggle to handle a really long input. For these reasons we simulate equivalence queries with the probably approximately correct (PAC) learning framework [19].

3.2 ExactLearner Framework

Aligned with Angluin's learning algorithm [2], the exact learner paradigm employs membership and equivalence queries to systematically construct ontologies. The framework transcends assumptions about the articulation clarity of domain experts, ensuring robustness across varied levels of expertise. Building upon prior research, an algorithm for learning $\mathcal{EL}$ terminologies is proposed, demonstrating exponential time complexity in concept expression and vocabulary size. Despite its computational demands, ExactLearner, implementing this algorithm, showcases success in terminating for small and medium-sized ontologies, thereby validating its efficacy in ontology construction.

Given a class of ontologies L (e.g., all ontologies in a specific DL, such as $\mathcal{EL}$ terminologies), the goal is to precisely identify a target ontology $O \in L$ by making queries to an oracle. It is assumed that the learner knows the vocabulary of the target ontology Σ_O.

- **Membership Query:** This involves asking the oracle whether an inclusion $C \sqsubseteq D$, where C and D are concept expressions in the DL's vocabulary Σ_O, is entailed by O ($O \models C \sqsubseteq D$). If $O \models C \sqsubseteq D$, the inclusion is a positive example with respect to the target ontology O; otherwise, it is a negative example.

- **Equivalence Query:** This involves asking the oracle whether a hypothesis ontology H is equivalent to the target ontology O. If they are equivalent, the oracle answers "yes." If not, the oracle provides either a positive example $C \sqsubseteq D$ where $H \not\models C \sqsubseteq D$ or a negative example $E \sqsubseteq F$ where $H \models E \sqsubseteq F$. These examples are referred to as positive and negative counterexamples, respectively.

A class of ontologies L is considered exactly learnable if there exists an algorithm that, for any target ontology $O \in L$, halts and computes a hypothesis $H \in L$ using membership and equivalence queries, such that $H \equiv O$. If this algorithm operates in polynomial time, the class is deemed exactly learnable in polynomial time. Specifically, at each computational step, the time taken is bounded by a polynomial $p(|O|, |C \sqsubseteq D|)$, where O represents the target ontology and $C \sqsubseteq D$ is the largest counterexample encountered.

By utilizing membership and equivalence queries, learners navigate through the ontology space, striving to accurately pinpoint the target ontology. While classes such as $ELlhs$ and $ELrhs$ demonstrate polynomial time learnability, the broader class of $\mathcal{EL}$ ontologies presents challenges for polynomial time learning, reflecting the intricate nature of ontology space exploration. Research by Konev et al. [14] confirms that $ELlhs$ and $ELrhs$ are indeed exactly learnable in polynomial time, while the entire class of $\mathcal{EL}$ ontologies does not lend itself to polynomial time learning.

Algorithm 1 The Learning Algorithm for $\mathcal{EL}$

Require: An $\mathcal{EL}$ terminology O given to the oracle; Σ_O given to the learner.
Ensure: An $\mathcal{EL}$ terminology H computed by the learner such that $O \equiv H$.

Initialization:
Set $H = \{A \sqsubseteq B \mid O \models A \sqsubseteq B, A, B \in \Sigma_O\}$.
while $H \not\equiv O$ **do**
  While $H \not\equiv O$:
  Get Counterexample:
  Let $C \sqsubseteq D$ be the returned positive counterexample for O relative to H.
  Compute $C' \sqsubseteq D'$:
  Compute $C' \sqsubseteq D'$ with C' or D' in $\Sigma_O \cap NC$ (where NC is the set of concept names).
  Determine O-essential Concept:
  if $C' \in \Sigma_O \cap NC$ **then**
    Compute a right O-essential α from $C' \sqsubseteq D'$.
  else
    if $C' \sqsubseteq F' \in H$ **then**
      Compute a left O-essential α from $C' \sqsubseteq D'$.
    end if
  end if
  Update Hypothesis:
  Add α to H.
end while
return Hypothesis: H

The algorithm is designed for the exact learning of $\mathcal{EL}$ terminologies. The learning process exhibits exponential time complexity in the size of the largest concept expression in the ontology, denoted as $|C_O|$, and the size of the ontology vocabulary, $|\Sigma_O|$. However, it does not depend on the size of the entire ontology.
Main Loop:

- The learner repeatedly poses equivalence queries to the oracle.
- If the oracle responds with "yes," the algorithm concludes that H is equivalent to O, and H is returned.
- If the oracle provides a counterexample $C \sqsubseteq D$, the algorithm updates its hypothesis H.

Key Observations:

- At every step, the hypothesis H is such that $O \models H$. Thus, the counterexamples provided by the oracle are always positive.
- Given that O is a terminology, any complex expressions C and D in the counterexample can only be related through a concept name. This connection can be identified through membership queries.

Explanation:

- When a positive counterexample $C \sqsubseteq D$ is received, the algorithm transforms it to a simpler form $C' \sqsubseteq D'$ where either C' or D' is a concept name.
- This transformation is achieved using membership queries and is efficient, bounded by polynomial time in the size of the hypothesis H, the concept C, and the vocabulary Σ_O.

Process:

- If C' is a concept name, the algorithm merges D' with the right-hand sides of all inclusions in H with C' on the left, computing a right O-essential counterexample.
- If D' is a concept name, the algorithm computes a left O-essential counterexample.

Right O-essential Counterexample:

- **Concept Saturation for** O: If $O \models A \sqsubseteq C'$ and C' results from C by adding a concept name A' to the label of some node, then replace $A \sqsubseteq C$ with $A \sqsubseteq C'$.
- **Sibling Merging for** O: If $O \models A \sqsubseteq C'$ and C' is the result of identifying two r-successors of the same node in C, then replace $A \sqsubseteq C$ with $A \sqsubseteq C'$.

- **Decomposition on the Right for** O: If d' is an r-successor of d in C, A' is in the node label of d, and $O \models A' \sqsubseteq \exists r.C_{d'}$ with $A' \not\equiv_O A$ if d is the root of C, then replace $A \sqsubseteq C$ by:
  - $A' \sqsubseteq \exists r.C_{d'}$
  - $A \sqsubseteq C|-d$ if $H \not\models A' \sqsubseteq \exists r.C_{d'}$; or
  - $d'\downarrow$, where C_d is the concept corresponding to the subtree rooted at d, and $C|-d\downarrow$ is the concept corresponding to the result of removing the subtree rooted at d from C.

Left O-essential Counterexample:

- **Concept Saturation for** H: If $H \models C \sqsubseteq C'$ and C' results from C by adding a concept name A' to the label of some node, then replace $C \sqsubseteq A$ with $C' \sqsubseteq A$.
- **Decomposition on the Left for** O: If d is a non-root node such that $O \models C|-d\downarrow \sqsubseteq A'$ and $H \not\models C|-d\downarrow \sqsubseteq A'$ for some $A' \in \Sigma_O$, then replace $C \sqsubseteq A$ by:
  - $C|-d\downarrow \sqsubseteq A'$
  - $C_d \sqsubseteq A'$ if $O \models C_d \sqsubseteq A'$ and $H \not\models C_d \sqsubseteq A'$.

3.3 PAC for equivalence queries

In PAC learning, a learning algorithm receives a labeled set of examples drawn from the instance space X using an unknown probability distribution D and attempts to find a hypothesis that approximates the target concept. The (finite) hypothesis class H is a set of possible hypotheses that a learning algorithm considers when trying to approximate the target concept. Each $h \in H$ is a potential candidate that the algorithm might select as the best approximation of the target concept based on the given training data. The size or complexity of H directly impacts the sample complexity m, which is the number of training examples required to ensure that the learning algorithm can – with high probability – find a hypothesis that is approximately correct. The framework requires two hyper-parameters: ϵ, which is the maximum error of the hypothesis, and δ, the learning algorithm produces a hypothesis with error less than ϵ with probability $1-\delta$. The minimum number of samples needed to PAC learn depends on the size of the instance space and on the values of ϵ and δ:

$$m \geq \frac{\log\left(|H|/\gamma\right)}{\epsilon} \tag{1}$$

For a given target ontology, we estimate the instance space X by considering all possible combinations of the following statements: (i) $A \cap B \sqsubseteq C$, (ii) $B \sqsubseteq \exists r.A$, and (iii) $\exists r.A \sqsubseteq B$. We limit the instance space to these three statements because they are sufficient to cover non-trivial ontologies, in particular the ones used for the experiments. Of course, one can choose different and

additional statements (e.g., use not only statements with chains of length one) but in doing so they should keep into account the size of the input space since it heavily affects the computation.

We then consider the upper bound to the size of the hypothesis class H to be equal to the size of the instance space to the power of the size of the number of axioms in the target ontology:

$$|H| = |X|^{|A|} \tag{2}$$

where $|X|$ is the size of the instance space, and $|A|$ is the number of axioms.

An equivalence query can then be simulated by probing random samples from X. If a statement is not entailed by the hypothesis ontology and it is labeled as 'true' by the LLM, then there is no equivalence and the sample is used as a counterexample. Instead, if none counterexample can be found within m samples, we say that the equivalence query is 'true'.

3.4 From Manchester OWL Syntax to Natural Language

The Manchester OWL syntax is a formal language for describing ontologies and it is quite close to the natural language. However, because is it most likely that LLMs have been trained more on data in natural language rather than in other formalisms, they could perform better if the query is also presented in natural language. We now describe the translation from the Manchester OWL syntax to natural language that we used in our experiments. All queries in the Manchester OWL syntax are of the form "`[A] SubClassOf [B]`". We substitute it with "Can `[A]` be considered a subcategory of `[B]`? Answer only with yes or no.", where `[A]` and `[B]` are placeholders for the concept expressions on the left and right hand side of the concept inclusion. This corresponds to the third formulation by Funk et al. [8], chosen in this work as it is closer to our setting. The translation of OWL concept expressions into natural language is then recursively applied to `[A]` and `[B]`.

1. If the query is of the form "`[A] and [B]`" then it is substituted with "`[A] that is also [B]`". The mapping is then recursively applied to `[A]` and `[B]`.

2. If the query is of the form "`[r] some [A]`", where `[r]` is a role name (or object property in OWL terminology), then it is substituted with "`something that [r] [A]`" and the substitution is recursively applied to `[A]`.

We provide examples for the procedure described above.

- If the input message is "Person SubClassOf Human", the function will return: "Can Person be considered a subcategory of Human?"

- If the input message is "Carnivore SubClassOf Animal and eats some Meat" The function will return: "Can Carnivore be considered a subcategory of Animal that is also something that eats Meat?"

- If the input message is "(Bird and Fish) SubClassOf Animal": The function will return: "Can Bird that is also Fish be considered a subcategory of Animal?"

4 Probing Language Models

In this section we describe the main challenges encountered when probing LLMs with ontology axioms and how we handled them.

4.1 Input Format and Unexpected Responses

One important factor is the format of the query. To systematically query an LLM with the goal of learning an ontology, it is useful to standardise the questions, so as to improve the transparency of the whole process and increase the level of reproducibility. For the membership queries task, we investigate the use of the Manchester OWL syntax [10], as this is an ontology syntax designed to be closer to natural language. Another aspect to consider is that, in principle, there are no constraints in the answers returned by the language model. An LLM may answer with an arbitrary and unexpected response, even if the expected answer is just a single word like in the case of membership queries in the exact learning model. To mitigate this issue, one can explicitly tell the LLM to answer with 'true' or 'false'. This particular request can be done in the question itself (e.g., appending "Answer with 'true' or 'false'." after the query) or by exploiting some hyper-parameters of the API of the LLM. In the second case, one can use a *system prompt* – a.k.a., integrated text into each query within the chat session – to enrich the model with additional information and useful to harness the response. We highlight that there are also other hyper-parameters that could help driving the LLM's response into the desired format (e.g., maximum number of tokens, temperature, etc.). Even with all these precautions the model may return an unexpected response. For example: (i) the answer can have more text than just 'true' or 'false', (ii) both 'true' and 'false' can appear in the answer, (iii) the answer does not have 'true' nor 'false'. While in the first scenario a trivial parsing would determine the correct classification, in the remaining cases, since there is some ambiguity, we considered a third value, which we called 'unknown'.

4.2 Correctness and Logical Consistency

We also need to deal with challenges regarding the correctness of the responses (assuming the format of the responses returned by the language model are as expected, see Section 4.1). Actively learning ontologies has been investigated for various fragments of $\mathcal{EL}$ [13, 16, 7], though, without using LLMs as teachers. If an LLM is playing the role of the teacher then there is no guarantee that the responses are correct [8] (in the sense of reflecting the 'truth' about the real world) and, moreover, that they are logically consistent with any $\mathcal{EL}$ ontology. Indeed, it is known that LLMs can learn statistical features instead of performing logical reasoning [21]. So, we need to consider the following kinds of errors:

1. $C \sqsubseteq D$ should be 'false' (cf. the real world) but the LLM answers 'true';
2. $C \sqsubseteq D$ should be 'true' (cf. the real world) but the LLM answers 'false';

3. all concept inclusions in $= \{C_1 \sqsubseteq D_1, \ldots, C_n \sqsubseteq D_n\}$ are answered with 'true', $\models C \sqsubseteq D$ but $C \sqsubseteq D$ is classified as 'false'.

The last case is a logical inconsistency. One strategy to handle this issue is to consider the *closure* under logical consequence [6]. That is, in Point 3, one could consider $C \sqsubseteq D$ as 'true'.

5 Experiments

We first consider the following ontologies (these ontologies were used in the work that presented ExactLearner [7]):

1. *Animals* contains knowledge related to the animal realm, including actual animals, subphyla, classes, orders, etc. The ontology has 12 (explicit) logical axioms in $\mathcal{EL}$ and 20 logical axioms in the logical closure (that is, taking into account inferred axioms).

2. *Bio-primitive* has information about different kinds of measures in the biological domain such as weight measure, blood pressure and specific metrics (e.g., Hamilton anxiety score). The ontology has 122 logical axioms (no new axioms are inferred by the engine).

3. *Biosphere* is a rich ontology providing information about animals and plants. The ontology has 86 logical axioms (no new axioms are inferred by the engine).

4. *Cell* provides information about different cells based on their type, development stage and organism. The ontology has 24 logical axioms in $\mathcal{EL}$ and 24 in the logical closure.

5. *Football* is a minimal ontology that describes the relations between football game, teams, players and managers. It has 9 logical axioms in $\mathcal{EL}$ and 12 in the logical closure.

6. *Generations* describes the members and relations within a family. This ontology has 18 (explicit) logical axioms in $\mathcal{EL}$ and 42 in the logical closure.

7. *University* is a small ontology, focusing on the professor role, with 4 logical axioms in the logical axioms in $\mathcal{EL}$ and 8 in the logical closure.

We use 5 open LLMs: Mistral [1] with 7 billion parameters, Mixtral [11] with 47 billion parameters, Llama2 with 7 and 13 billion parameters and the very recent LLama3 with 8 billion parameters [18] (we use Ollama's API[2]). We try two different formalism for the queries: the Manchester OWL syntax and the natural language (see Section 2.2). Finally, we use two different system prompts to provide contextual information to LLMs.

```
Answer with only True or False.
```

This first prompt is minimal and it simply suggests the LLM to answer with a single word, i.e., "True" or "False".

```
You need to classify the following statements as True or False.
The statement will be provided in either Manchester OWL syntax or
natural language. Follow strictly these guidelines:
```

[2]https://github.com/ollama/ollama

Animals	Bio-primitive	Biosphere	Cell	Football	Generation	University
1,082	17,629	11,508	2,242	655	1,669	274

Table 1: Minimum number of axioms needed to PAC learn an ontology.

```
1. answer with only True or False;
2. entities from different categories are not in a subclass
   relationship;
3. statements or question containing numerous entities are most
   probably False;
4. take a deep breath before answering;
5. if you are unsure about the classification, answer with False.
```

The second prompt instead provides much more context information and tells the LLM to follow some guidelines. Point 1 is the same sentence used in the first prompt. Point 2 is a remark for the LLM to stress out that there could be concept names that refer to different categories and cannot be put in a subclass relation. For instance, "Fish" (animal) lives in "Water" (environment) and the two concept names should not be in a subclass relationship like "Fish" $\sqsubseteq$ "Water". Point 3 has the purpose to prevent the LLM to answer "True" to long queries (i.e., queries involving several concept and role names). Long queries may appear when the learner tries to apply the rules in sec:active-learning-ontologies. Point 4 is an expedient that has been found to be effective to improve LLM performance [20]. The last point is a final remark to be conservative in the replies.

Concerning the simulation of equivalence query using the PAC framework, we set to 0.1 the value of ϵ and we set to 0.2 the value of γ. We report in 1 the minimum number of axioms needed to PAC learn the chosen ontologies.

5.1 Evaluation

We evaluate the results of experiments by computing the following metrics: accuracy, precision, recall and F1-score. The metrics are computed considering the total number of axioms is the size of the instance space $|X|$. Therefore, every possible depth-one axiom in the form of $A \sqcap B \sqsubseteq C$, $B \sqsubseteq \exists r.A$, and $\exists r.A \sqsubseteq B$. The axioms that are both (resp. not) entailed by the original ontology and the learned ontology are labeled as true positive (resp. true negative). Axioms entailed by the original ontology but not by the learned ontology are labeled as false negative, vice-versa they are labeled as false positive.

Model	Prompt & Query	Accuracy	Recall	Precision	F1-Score
Llama2 (13b)	M. OWL Syntax	0.148	0.148	1.0	0.258
	Natural Language	0.394	0.196	0.999	0.328
	E. M. OWL Syntax	0.2	0.156	1.0	0.27
	E. Natural Language	0.995	0.981	0.987	0.984
Llama2 (7b)	M. OWL Syntax	0.148	0.148	1.0	0.258
	Natural Language	0.148	0.148	1.0	0.258
	E. M. OWL Syntax	0.148	0.148	1.0	0.258
	E. Natural Language	0.148	0.148	1.0	0.258
Llama3 (8b)	M. OWL Syntax	0.148	0.148	1.0	0.258
	Natural Language	0.264	0.167	0.999	0.287
	E. M. OWL Syntax	0.306	0.176	0.998	0.299
	E. Natural Language	0.956	0.775	0.993	0.87
Mistral (7b)	M. OWL Syntax	0.222	0.16	0.999	0.276
	Natural Language	0.867	0.528	0.952	0.68
	E. M. OWL Syntax	0.883	0.56	0.987	0.714
	E. Natural Language	0.95	0.752	0.987	0.853
Mixtral (47b)	M. OWL Syntax	0.202	0.156	0.999	0.271
	Natural Language	0.87	0.533	0.993	0.694
	E. M. OWL Syntax	0.979	0.888	0.985	0.934
	E. Natural Language	0.998	1.0	0.985	0.992

Table 2: Metrics for Animals ontology

Model	Prompt & Query	Accuracy	Recall	Precision	F1-Score
Llama2 (13b)	M. OWL Syntax	0.055	0.055	1.0	0.104
	Natural Language	0.103	0.058	1.0	0.109
	E. M. OWL Syntax	0.262	0.069	1.0	0.13
	E. Natural Language	1.0	1.0	1.0	1.0
Llama2 (7b)	M. OWL Syntax	0.055	0.055	1.0	0.104
	Natural Language	0.055	0.055	1.0	0.104
	E. M. OWL Syntax	0.055	0.055	1.0	0.104
	E. Natural Language	0.055	0.055	1.0	0.104
Llama3 (8b)	M. OWL Syntax	0.055	0.055	1.0	0.104
	Natural Language	0.063	0.055	1.0	0.105
	E. M. OWL Syntax	0.159	0.061	1.0	0.115
	E. Natural Language	0.979	0.72	1.0	0.837
Mistral (7b)	M. OWL Syntax	0.65	0.136	1.0	0.239
	Natural Language	0.93	0.44	1.0	0.611
	E. M. OWL Syntax	0.215	0.065	1.0	0.123
	E. Natural Language	1.0	1.0	1.0	1.0
Mixtral (47b)	M. OWL Syntax	0.286	0.071	1.0	0.133
	Natural Language	0.814	0.228	1.0	0.371
	E. M. OWL Syntax	0.971	0.652	1.0	0.789
	E. Natural Language	0.998	0.96	1.0	0.98

Table 3: Metrics for Bio-primitive ontology

Model	Prompt & Query	Accuracy	Recall	Precision	F1-Score
Llama2 (13b)	M. OWL Syntax	0.055	0.054	1.0	0.102
	Natural Language	0.506	0.098	0.999	0.179
	E. M. OWL Syntax	0.133	0.059	1.0	0.111
	E. Natural Language	0.997	0.946	1.0	0.972
Llama2 (7b)	M. OWL Syntax	0.054	0.054	1.0	0.102
	Natural Language	0.054	0.054	1.0	0.102
	E. M. OWL Syntax	0.054	0.054	1.0	0.102
	E. Natural Language	0.054	0.054	1.0	0.102
Llama3 (8b)	M. OWL Syntax	0.054	0.054	1.0	0.102
	Natural Language	0.51	0.099	1.0	0.18
	E. M. OWL Syntax	0.146	0.059	1.0	0.112
	E. Natural Language	0.726	0.164	1.0	0.282
Mistral (7b)	M. OWL Syntax	0.627	0.126	1.0	0.224
	Natural Language	0.691	0.148	1.0	0.258
	E. M. OWL Syntax	0.247	0.067	1.0	0.125
	E. Natural Language	0.915	0.387	1.0	0.558
Mixtral (47b)	M. OWL Syntax	0.283	0.07	1.0	0.131
	Natural Language	0.764	0.186	1.0	0.314
	E. M. OWL Syntax	0.998	0.964	1.0	0.982
	E. Natural Language	1.0	0.997	1.0	0.998

Table 4: Metrics for Biosphere ontology

Model	Prompt & Query	Accuracy	Recall	Precision	F1-Score
Llama2 (13b)	M. OWL Syntax	0.239	0.239	1.0	0.386
	Natural Language	0.239	0.239	1.0	0.386
	E. M. OWL Syntax	0.239	0.239	1.0	0.386
	E. Natural Language	1.0	1.0	1.0	1.0
Llama2 (7b)	M. OWL Syntax	0.239	0.239	1.0	0.386
	Natural Language	0.239	0.239	1.0	0.386
	E. M. OWL Syntax	0.239	0.239	1.0	0.386
	E. Natural Language	0.247	0.241	1.0	0.388
Llama3 (8b)	M. OWL Syntax	0.239	0.239	1.0	0.386
	Natural Language	0.239	0.239	1.0	0.386
	E. M. OWL Syntax	0.239	0.239	1.0	0.386
	E. Natural Language	0.446	0.302	1.0	0.464
Mistral (7b)	M. OWL Syntax	0.36	0.272	1.0	0.428
	Natural Language	0.594	0.371	1.0	0.541
	E. M. OWL Syntax	0.281	0.25	1.0	0.399
	E. Natural Language	0.986	0.944	1.0	0.971
Mixtral (47b)	M. OWL Syntax	0.239	0.239	1.0	0.386
	Natural Language	0.632	0.394	1.0	0.565
	E. M. OWL Syntax	0.945	0.813	1.0	0.897
	E. Natural Language	1.0	1.0	1.0	1.0

Table 5: Metrics for Cell ontology

Model	Prompt & Query	Accuracy	Recall	Precision	F1-Score
Llama2 (13b)	M. OWL Syntax	0.229	0.229	1.0	0.372
	Natural Language	0.319	0.251	1.0	0.402
	E. M. OWL Syntax	0.229	0.229	1.0	0.372
	E. Natural Language	0.97	0.903	0.971	0.936
Llama2 (7b)	M. OWL Syntax	0.229	0.229	1.0	0.372
	Natural Language	0.229	0.229	1.0	0.372
	E. M. OWL Syntax	0.229	0.229	1.0	0.372
	E. Natural Language	0.229	0.229	1.0	0.372
Llama3 (8b)	M. OWL Syntax	0.229	0.229	1.0	0.372
	Natural Language	0.229	0.229	1.0	0.372
	E. M. OWL Syntax	0.229	0.229	1.0	0.372
	E. Natural Language	0.422	0.282	0.989	0.439
Mistral (7b)	M. OWL Syntax	0.229	0.229	1.0	0.372
	Natural Language	0.229	0.229	1.0	0.372
	E. M. OWL Syntax	0.768	0.496	0.953	0.653
	E. Natural Language	0.989	1.0	0.953	0.976
Mixtral (47b)	M. OWL Syntax	0.229	0.229	1.0	0.372
	Natural Language	0.311	0.249	1.0	0.399
	E. M. OWL Syntax	0.989	1.0	0.953	0.976
	E. Natural Language	0.989	1.0	0.953	0.976

Table 6: Metrics for Football ontology

Model	Prompt & Query	Accuracy	Recall	Precision	F1-Score
Llama2 (13b)	M. OWL Syntax	0.234	0.234	1.0	0.379
	Natural Language	0.476	0.305	0.968	0.464
	E. M. OWL Syntax	0.279	0.244	0.991	0.391
	E. Natural Language	0.973	1.0	0.883	0.938
Llama2 (7b)	M. OWL Syntax	0.234	0.234	1.0	0.379
	Natural Language	0.234	0.234	1.0	0.379
	E. M. OWL Syntax	0.234	0.234	1.0	0.379
	E. Natural Language	0.234	0.234	1.0	0.379
Llama3 (8b)	M. OWL Syntax	0.234	0.234	1.0	0.379
	Natural Language	0.234	0.234	1.0	0.379
	E. M. OWL Syntax	0.234	0.234	1.0	0.379
	E. Natural Language	0.903	0.735	0.913	0.815
Mistral (7b)	M. OWL Syntax	0.714	0.447	0.939	0.605
	Natural Language	0.549	0.337	0.957	0.498
	E. M. OWL Syntax	0.964	0.953	0.89	0.921
	E. Natural Language	0.974	0.979	0.907	0.941
Mixtral (47b)	M. OWL Syntax	0.234	0.234	1.0	0.379
	Natural Language	0.472	0.303	0.968	0.462
	E. M. OWL Syntax	0.973	1.0	0.883	0.938
	E. Natural Language	0.973	1.0	0.883	0.938

Table 7: Metrics for Generations ontology

Model	Prompt & Query	Accuracy	Recall	Precision	F1-Score
Llama2 (13b)	M. OWL Syntax	0.231	0.231	1.0	0.375
	Natural Language	0.936	0.805	0.952	0.872
	E. M. OWL Syntax	0.301	0.247	0.986	0.395
	E. Natural Language	0.989	1.0	0.952	0.976
Llama2 (7b)	M. OWL Syntax	0.231	0.231	1.0	0.375
	Natural Language	0.231	0.231	1.0	0.375
	E. M. OWL Syntax	0.231	0.231	1.0	0.375
	E. Natural Language	0.231	0.231	1.0	0.375
Llama3 (8b)	M. OWL Syntax	0.231	0.231	1.0	0.375
	Natural Language	0.231	0.231	1.0	0.375
	E. M. OWL Syntax	0.231	0.231	1.0	0.375
	E. Natural Language	0.986	0.986	0.952	0.969
Mistral (7b)	M. OWL Syntax	0.587	0.357	0.986	0.524
	Natural Language	0.783	0.517	0.952	0.67
	E. M. OWL Syntax	0.89	0.69	0.952	0.8
	E. Natural Language	0.958	0.875	0.952	0.912
Mixtral (47b)	M. OWL Syntax	0.231	0.231	1.0	0.375
	Natural Language	0.925	0.773	0.952	0.854
	E. M. OWL Syntax	0.989	1.0	0.952	0.976
	E. Natural Language	0.989	1.0	0.952	0.976

Table 8: Metrics for University ontology

Ontology	Accuracy	Recall	Precision	F1-Score
animals.owl	0.499	0.396	0.993	0.5
biological-measure-primitive.owl	0.438	0.292	1.0	0.358
biosphere.owl	0.443	0.235	1.0	0.302
cl.owl	0.444	0.399	1.0	0.526
football.owl	0.425	0.396	0.989	0.511
generations.owl	0.518	0.47	0.959	0.566
university.owl	0.571	0.516	0.977	0.615

Table 9: Average Metrics by Ontology

Model	Accuracy	Recall	Precision	F1-Score
llama2:13b	0.455	0.401	0.989	0.485
llama2	0.17	0.17	1.0	0.282
llama3	0.354	0.273	0.994	0.385
mistral	0.68	0.476	0.977	0.58
mixtral	0.724	0.613	0.981	0.681

Table 10: Average Metrics by Model

Metric Type	Accuracy	Recall	Precision	F1-Score
M. OWL Syntax	0.248	0.186	0.998	0.304
Natural Language	0.439	0.275	0.991	0.403
E. M. OWL Syntax	0.442	0.373	0.987	0.466
E. Natural Language	0.779	0.711	0.977	0.758

Table 11: Average Metrics by Metric Type

6 Conclusions

The findings presented in the results tables provide valuable insights into the performance of various models. Key observations include:

1. Mixtral shows superior performance compared to other models, likely due to its significantly larger number of parameters.

2. Across almost all ontologies, models achieve better metric scores when queries are expressed in natural language. This highlights the effectiveness of LLMs which utilize extensive natural language trained data to achieve a deeper understanding compared to traditional Manchester Syntax OWL queries.

3. There is a significant performance boost when using custom system prompts. This enhancement leads to performance gains ranging from 70% to 230% across various scenarios, indicating the importance of tailored prompts in guiding model behavior.

4. Ontologies with a high number of object properties or specialized knowledge present greater challenges for the learning process, whereas ontologies with only concept names are more feasible to learn with high accuracy. This is discussed in Section 5, where we describe the types of statements considered (every possible depth-one axiom in the form of $A \cap B \sqsubseteq C$, $B \sqsubseteq \exists r.A$, and $\exists r.A \sqsubseteq B$). These domains often require more nuanced understanding and present hurdles for model adaptation.

Overall, these findings illuminate the nuanced dynamics of LLM performance across different contexts, highlighting both strengths and areas for improvement in their application to ontology learning tasks. The results were obtained without proper training or fine-tuning of the models, demonstrating surprising zero-shot learning performance in most cases.

References

[1] Albert Q. Jiang et al. "Mistral 7B". In: *CoRR* abs/2310.06825 (2023). DOI: 10.48550/ARXIV.2310.06825. arXiv: 2310.06825.

[2] Dana Angluin. "Learning regular sets from queries and counterexamples". In: *Information and Computation* 75.2 (1987), pp. 87–106. ISSN: 0890-5401. DOI: 10.1016/0890-5401(87)90052-6. URL: https://www.sciencedirect.com/science/article/pii/0890540187900526.

[3] Dana Angluin. "Queries and Concept Learning". In: *Mach. Learn.* 2.4 (1987), pp. 319–342. DOI: 10.1007/BF00116828.

[4] Franz Baader, Sebastian Brandt, and Carsten Lutz. "Pushing the EL Envelope". In: *IJCAI-05, Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence, Edinburgh, Scotland, UK, July 30 - August 5, 2005*. Ed. by Leslie Pack Kaelbling and Alessandro Saffiotti. Professional Book Center, 2005, pp. 364–369. URL: http://ijcai.org/Proceedings/05/Papers/0372.pdf.

[5] Franz Baader et al. *An Introduction to Description Logic*. Cambridge University Press, 2017. ISBN: 978-0-521-69542-8. URL: http://www.cambridge.org/de/academic/subjects/computer-science/knowledge-management-databases-and-data-mining/introduction-description-logic?format=PB#17zVGeWD2TZUeu6s.97.

[6] Sophie Blum et al. "Learning Horn envelopes via queries from language models". In: *International Journal of Approximate Reasoning* (2023), p. 109026. ISSN: 0888-613X. DOI: 10.1016/j.ijar.2023.109026.

[7] Mario Ricardo Cruz Duarte, Boris Konev, and Ana Ozaki. "ExactLearner: A Tool for Exact Learning of EL Ontologies". In: *KR*. Ed. by Michael Thielscher, Francesca Toni, and Frank Wolter. AAAI Press, 2018, pp. 409–414. URL: https://aaai.org/ocs/index.php/KR/KR18/paper/view/18006.

[8] Maurice Funk et al. "Towards Ontology Construction with Language Models". In: *Joint proceedings of the 1st workshop on Knowledge Base Construction from Pre-Trained Language Models (KBC-LM) and the 2nd challenge on Language Models for Knowledge Base Construction (LM-KBC) co-located with the 22nd International Semantic Web Conference (ISWC 2023), Athens, Greece, November 6, 2023*. Ed. by Simon Razniewski et al. Vol. 3577. CEUR Workshop Proceedings. CEUR-WS.org, 2023. URL: https://ceur-ws.org/Vol-3577/paper16.pdf.

[9] Bernardo Cuenca Grau et al. "OWL 2: The Next Step for OWL". In: *Journal of Web Semantics* 6.4 (2008), pp. 309–322. URL: http://web.comlab.ox.ac.uk/people/Boris.Motik/pubs/ghmppss08next-steps.pdf.

[10] Matthew Horridge et al. "The Manchester OWL Syntax". In: *Proceedings of the OWLED*06 Workshop on OWL: Experiences and Directions, Athens, Georgia, USA, November 10-11, 2006*. Ed. by Bernardo Cuenca Grau et al. Vol. 216. CEUR Workshop Proceedings. CEUR-WS.org, 2006. URL: https://ceur-ws.org/Vol-216/submission_9.pdf.

[11] Albert Q. Jiang et al. "Mixtral of Experts". In: *CoRR* abs/2401.04088 (2024). DOI: 10.48550/ARXIV.2401.04088. arXiv: 2401.04088.

[12] Boris Konev, Ana Ozaki, and Frank Wolter. "A Model for Learning Description Logic Ontologies Based on Exact Learning". In: *AAAI*. Ed. by Dale Schuurmans and Michael P. Wellman. AAAI Press, 2016, pp. 1008–1015. DOI: 10.1609/AAAI.V30I1.10087.

[13] Boris Konev et al. "Exact Learning of Lightweight Description Logic Ontologies". In: *J. Mach. Learn. Res.* 18 (2017), 201:1–201:63. URL: http://jmlr.org/papers/v18/16-256.html.

[14] Boris Konev et al. "Exact Learning of Lightweight Description Logic Ontologies". In: *Journal of Machine Learning Research* 18.201 (2018), pp. 1–63. URL: http://jmlr.org/papers/v18/16-256.html.

[15] Natalya Fridman Noy et al. "BioPortal: A Web Repository for Biomedical Ontologies and Data Resources". In: *ISWC*. Ed. by Christian Bizer and Anupam Joshi. Vol. 401. CEUR Workshop Proceedings. CEUR-WS.org, 2008. URL: https://ceur-ws.org/Vol-401/iswc2008pd_submission_25.pdf.

[16] Ana Ozaki, Cosimo Persia, and Andrea Mazzullo. "Learning Query Inseparable ELH Ontologies". In: *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*. AAAI Press, 2020, pp. 2959–2966. DOI: 10.1609/AAAI.V34I03.5688.

[17] Simon Razniewski et al., eds. *Joint proceedings of the 1st workshop on Knowledge Base Construction from Pre-Trained Language Models (KBC-LM) and the 2nd challenge on Language Models for Knowledge Base Construction (LM-KBC) co-located with the 22nd International Semantic Web Conference (ISWC 2023), Athens, Greece, November 6, 2023*. Vol. 3577. CEUR Workshop Proceedings. CEUR-WS.org, 2023. URL: https://ceur-ws.org/Vol-3577.

[18] Hugo Touvron et al. "Llama 2: Open Foundation and Fine-Tuned Chat Models". In: *CoRR* abs/2307.09288 (2023). DOI: 10.48550/ARXIV.2307.09288. arXiv: 2307.09288.

[19] Leslie G. Valiant. "A Theory of the Learnable". In: *Commun. ACM* 27.11 (1984), pp. 1134–1142. DOI: 10.1145/1968.1972. URL: https://doi.org/10.1145/1968.1972.

[20] Chengrun Yang et al. “Large Language Models as Optimizers”. In: *The Twelfth International Conference on Learning Representations*. 2024. URL: https://openreview.net/forum?id=Bb4VGOWELI.

[21] Honghua Zhang et al. “On the Paradox of Learning to Reason from Data”. In: *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI 2023, 19th-25th August 2023, Macao, SAR, China*. ijcai.org, 2023, pp. 3365–3373. DOI: 10.24963/ijcai.2023/375.