ECAI 2025
I. Lynce et al. (Eds.)

doi:10.3233/FAIA251009

Actively Learning $\mathcal{EL}$ Terminologies from Large Language Models

Matteo Magnini [a], Riccardo Squarcialupi[a], Martin T. Sterri[c] and Ana Ozaki [b,c,*]

[a]University of Bologna
[b]University of Oslo
[c]University of Bergen

Abstract. In active learning, a learner attempts to learn from a teacher by posing questions. The questions made by the learner are called *membership queries* and are answered with 'yes' or 'no'. This kind of query is often studied as part of a communication protocol that also includes *equivalence queries*. Intuitively, equivalence queries ask whether the idea of the learner about the knowledge of the teacher is correct or not. If not, then the teacher should provide a counterexample showing the difference. Here, we consider the teacher as a large language model (LLM) and study the case in which knowledge is expressed as an $\mathcal{EL}$ terminology. Membership queries ask whether concept inclusions are true or not. E.g., "Can algae be considered a subcategory of plant?". Equivalence queries are simulated by a sample with concept inclusions labelled as positive or negative. We present a non-trivial extension of the ExactLearner tool to extract $\mathcal{EL}$ terminologies from LLMs. Given the relevant symbols as input (e.g., algae, plant, etc.), the tool tries to find how these symbols should be logically connected by posing questions to LLMs. To evaluate the approach, we present performance results of the ExactLearner in the task of reconstructing existing $\mathcal{EL}$ terminologies.

1 Introduction

Large language models (LLMs) have reached a point where they have accumulated so much information, and improved on their question/answering capability, that several works are now focusing on designing methods to interact and extract knowledge from them. Prompts to these models vary from questions about general knowledge such as basic definitions and historical events, to more domain specific questions, e.g., scientific facts related to health and medicine. Recently, some efforts have been made in automating the extraction of knowledge from LLMs, where knowledge is extracted in the format of an *ontology* [11, 12, 14, 18, 30]. In the field of knowledge representation and reasoning, ontologies are the standard way of representing the relevant conceptual knowledge of a domain of interest. The most popular formalism for specifying ontologies is given by *description logic* [7]. Ontologies have been extensively used for representing hierarchies and supporting data integration, information retrieval, and automated reasoning in life sciences [27].

Example 1. We provide an expression that corresponds to a *concept inclusion* in description logic and what it means in natural language.

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

* Corresponding Author. Email: anaoz@uio.no.

What can we learn from LLMs? And, since it is known that they can give false information, is there an automated way of discovering whether responses are incorrect or at least inconsistent? To study these questions, in this work we explore an active learning approach to learn *description logic ontologies* from LLMs. Our work is based on Angluin's exact learning framework [3] from computational learning theory, where a learner attempts to learn some target knowledge from a teacher by posing questions (learning by posing queries is known as *active learning* [4]). Here we consider the teacher as an LLM and study the case in which the knowledge to be learnt is an ontology formulated in the classical $\mathcal{EL}$ description logic [6].

Example 2. $\mathcal{EL}$ ontologies allow for expressions involving conjunctions and existential quantification. The following example illustrates a concept inclusion in $\mathcal{EL}$.

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

This can be translated into the sentence: "Conifers can be considered a subcategory of Gymnosperms that bears cones."

Exact learning of lightweight description logic ontologies and concepts has been studied for theoretical purposes [16, 17, 24, 25, 28]. One of the main difficulties of employing such algorithms in practice is precisely finding a teacher that can answer the questions made by the algorithms. The only known implementation for exact learning description logic ontologies is called *ExactLearner* [13]. The tool is designed to learn $\mathcal{EL}$ terminologies [1] from a synthetic teacher that can answer *membership* and *equivalence* queries. Intuitively, membership queries ask whether a given concept inclusion is 'true' or 'false', while equivalence queries ask whether the idea of the learner about the knowledge of the teacher is correct or not. If not, then the teacher should provide a counterexample showing the difference. This synthetic teacher was created by the authors to test the implementation, using ontologies from the Oxford Ontology Repository [2] to answer the queries. Given the relevant symbols, called *vocabulary* or *signature*, as input (e.g., algae, plant, etc.), the tool tries to find how these symbols should be logically connected, by posing membership and equivalence queries to the synthetic teacher.

In this work, we investigate the automated construction of $\mathcal{EL}$ terminologies with a non-trivial adaptation of the ExactLearner tool, which prompts LLMs as teachers. We call our extended version *ExactLearner+LLM*. We use the Manchester OWL Syntax [20] and our

[1] An $\mathcal{EL}$ terminology is an ontology with some syntactic restrictions (Sec 2).
[2] https://www.cs.ox.ac.uk/isg/ontologies/

own parsing function to map concept inclusions in description logic into natural language. Membership queries can be easily mapped into a question-answering setting with LLMs, however, equivalence queries are more difficult to answer [33, 9]. This is solved by simulating equivalence queries with a sample of randomly generated concept inclusions classified as 'true' or 'false' by the LLM (a collection of membership queries with their answers) (see [3] for a theoretical analysis on this strategy within the probably approximately correct framework). To evaluate our approach, we present results showing the performance of the ExactLearner+LLM tool when prompting open source LLMs. In more detail, we give the signature as input to our ExactLearner+LLM tool and check the results for the task of reconstructing existing ontologies. In our experiments, we consider small ontologies from the benchmark originally used by the authors of the ExactLearner tool and also modules extracted from the GALEN ontology [2], with medical terms. We also indicate whether there is statistical evidence of correlation between the concept inclusions extracted with the ExactLearner+LLM tool and the original ontologies, constructed by (human) ontology engineers.

2 Actively Learning Ontologies

Here we define the notion of an $\mathcal{EL}$ terminology, provide more details about the exact and PAC learning frameworks, and the algorithm in the ExactLearner tool [13], designed to learn $\mathcal{EL}$ terminologies from a teacher that answers membership and equivalence queries.

The $\mathcal{EL}$ Description Logic Let $\mathsf{N_C}$ and $\mathsf{N_R}$ be countably infinite and disjoint sets of *concept* and *role* names, respectively (see below cases in which we may abuse notation and write $\mathsf{N_C}$ and $\mathsf{N_R}$ denoting finite sets). An $\mathcal{EL}$ *concept* is an expression of the form $C, D := C \sqcap D \mid \exists r.C \mid A \mid \top$, where $A \in \mathsf{N_C}$ and $r \in \mathsf{N_R}$. An $\mathcal{EL}$ *concept inclusion* is of the form $C \sqsubseteq D$ with C, D being $\mathcal{EL}$ concepts and an $\mathcal{EL}$ *equivalence* is of the form $C \equiv D$ (which can be alternatively expressed as $C \sqsubseteq D$ and $D \sqsubseteq C$). An $\mathcal{EL}$ *ontology* is a finite set of $\mathcal{EL}$ concept inclusions $C \sqsubseteq D$. Given an $\mathcal{EL}$ ontology $\mathcal{O}$, we write $\mathsf{N_C}(\mathcal{O})$ and $\mathsf{N_R}(\mathcal{O})$ for the (finite) sets of concept and role names that occur in $\mathcal{O}$. In the literature, these sets are also called the *signature* of $\mathcal{O}$. We may omit '$(\mathcal{O})$' and simply write $\mathsf{N_C}$ and $\mathsf{N_R}$ when $\mathcal{O}$ is clear from the context. If, for all concept inclusions $C \sqsubseteq D$ in an ontology $\mathcal{O}$, at least one of C, D is in $\mathsf{N_C}$ and it has at most one inclusion of the form $A \sqsubseteq C$ for every $A \in \mathsf{N_C}$ then we say that $\mathcal{O}$ is a *terminology* [3]. The semantics of $\mathcal{EL}$ is given by interpretations, as usual [7]. Given an ontology $\mathcal{O}$ and a concept inclusion $C \sqsubseteq D$, we say that $\mathcal{O}$ *entails* $C \sqsubseteq D$, in symbols $\mathcal{O} \models C \sqsubseteq D$, if all interpretations that satisfy $\mathcal{O}$ also satisfy $C \sqsubseteq D$. Also, given two ontologies $\mathcal{O}$ and $\mathcal{H}$, we say that they are *equivalent*, in symbols $\mathcal{O} \equiv \mathcal{H}$, if they are satisfied by the same interpretations. In other words, if, for all concept inclusions $C \sqsubseteq D$, we have that $\mathcal{O} \models C \sqsubseteq D \Leftrightarrow \mathcal{H} \models C \sqsubseteq D$.

Exact Learning In Angluin's exact learning framework, a learner attempts to learn some abstract target – which in this work is an $\mathcal{EL}$ terminology – from a teacher. The most studied communication protocol between the learner and the teacher includes two kinds of queries: *membership queries*, in which the learner asks whether a given statement is true or false; and *equivalence queries*, in which the learner gives a hypothesis as input and receives as return either 'yes' (meaning the hypothesis is equivalent to the target) or 'no' together with a counterexample showing a difference between the hypothesis and the target [3]. In our setting, membership queries ask whether an $\mathcal{EL}$ concept inclusion $C \sqsubseteq D$ is true or false (e.g., ask an LLM whether $\mathsf{Algae} \sqsubseteq \mathsf{Plant}$ in Example 1 is true or false).

Simulating Equivalence Queries We use a *sampling* strategy to simulate equivalence queries. Instead of asking the LLM to evaluate the hypothesis and provide a counterexample, we randomly generate a set of $\mathcal{EL}$ concept inclusions and ask the LLM whether it is true or false. Then we check whether the hypothesis of the learner is consistent with the classification of the LLM. That is, whether the true concept inclusions are logical consequences of the hypothesis and whether false concept inclusions are not logical consequences. If consistent, then we stop the learning process, otherwise, we have found a counterexample and we can proceed as if the LLM had answered an equivalence query with 'no' and returned the counterexample. This sampling strategy and some heuristics have been used in previous work in the literature to simulate equivalence queries [33, 9]. Under certain conditions, it can be used to provide a theoretical guarantee within the probably approximately correct framework (PAC) [3, 32]. In this framework, the sample size is calculated based on the hypothesis space $\mathbf{H}$ (in our case, $\mathcal{EL}$ terminologies of a certain size and shape formulated with a given signature), and two parameters ϵ and δ quantifying error and confidence, respectively. According to the theory behind PAC learning (see e.g., [29, Cor. 2.3]), such sample should be at least $\ln(|\mathbf{H}|/\delta)/\epsilon$ for finite $\mathbf{H}$.

The Algorithm for $\mathcal{EL}$ Terminologies ExactLearner [13] is a tool for learning $\mathcal{EL}$ terminologies. In the original work, a learner interacts with a synthetic teacher that answers membership and equivalence queries. The answers are logically consistent with a target $\mathcal{EL}$ terminology and the communication is in OWL, not in natural language. The main steps of the learning procedure is given by Algorithm 1. The algorithm makes equivalence queries in Line 4 ('does $\mathcal{H} \not\equiv \mathcal{O}$'?) and membership queries ('does $\mathcal{O} \models C \sqsubseteq D$?') in Lines 6, 8 and 10. In Line 6, $C' \sqsubseteq D'$ is a counterexample extracted from $C \sqsubseteq D$ where one of the sides is in $\mathsf{N_C}(\mathcal{O})$. This line is ensured to find such counterexample because $\mathcal{O}$ is a terminology [13]. In Lines 6, 8 and 10, the algorithm performs different kinds of operations that use membership queries, namely *concept saturation for* $\mathcal{O}$, *sibling merging*, *decomposition on the right*, and *decomposition on the left*, plus two heuristics called *desaturation* and *branching*. The exhaustive application of these operations result in concept inclusions that, intuitively, maximise how informative they are and also minimise their size. These concept inclusions are called $\mathcal{O}$*-essential* and are added to the hypothesis (Line 11).

3 Language Models as Teachers

While it is easy to argue that LLMs can offer great opportunities for knowledge acquisition in the format of ontologies, there are also some challenges that need to be addressed. The main challenges are:

- the format of membership queries in the ExactLearner is in OWL, but LLMs are designed to receive queries in natural language;
- the responses may not follow the expected format (even if we ask the LLMs to just answer 'yes' or 'no');
- the length of the queries made by the ExactLearner can be long, with several logical operators, which could affect the ability of LLMs to answer queries correctly;
- even if answer has the right format and short, the responses of the LLM may be logically inconsistent with previous answers, incorrect (and even change if asked multiple times);
- the knowledge that we aim to extract is limited by the expressivity of the ontology language that we adopt.

[3] In the DL literature [7], we can see terminologies defined as a set of equivalences with one of the sides in $\mathsf{N_C}$. Our notion is slightly more flexible.

Algorithm 1: The learning algorithm for $\mathcal{EL}$ [13]

Input (to the learner): $\mathsf{N_C}(\mathcal{O}) \cup \mathsf{N_R}(\mathcal{O})$ plus access to a teacher that can answer membership and equivalence queries (or just membership queries, for when equivalence queries are simulated in a PAC setting).
Output: An $\mathcal{EL}$ terminology $\mathcal{H}$ computed by the learner such that $\mathcal{O} \equiv \mathcal{H}$
Set $\mathcal{H} = \{A \sqsubseteq B \mid \mathcal{O} \models A \sqsubseteq B, A, B \in \mathsf{N_C}(\mathcal{O})\}$
while $\mathcal{H} \not\equiv \mathcal{O}$ **do**
  Let $C \sqsubseteq D$ be the returned positive counterexample for $\mathcal{O}$ relative to $\mathcal{H}$
  Compute $C' \sqsubseteq D'$ with C' or D' in $\mathsf{N_C}(\mathcal{O})$
  if $C' \in \mathsf{N_C}(\mathcal{O})$ **then**
    Compute a right $\mathcal{O}$-essential α from $C' \sqsubseteq D' \sqcap \bigsqcap_{C' \sqsubseteq F' \in \mathcal{H}} F'$
  else
    Compute a left $\mathcal{O}$-essential α from $C' \sqsubseteq D'$
  Add α to $\mathcal{H}$
return $\mathcal{H}$

We now discuss how we address each of these challenges.

Input Format An important factor is the format of the query. We use both the Manchester OWL syntax [20], as this is an ontology syntax designed to be closer to natural language, and controlled natural language. All queries in the Manchester OWL syntax are of the form "`[A] SubClassOf [B]`". In natural language, 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. [18], chosen in this work because 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]`.

Example 3. The concept inclusion in OWL "Person SubClassOf Human" is transformed into the question "Can Person be considered a subcategory of Human?" As another example, the concept inclusion "Carnivore SubClassOf Animal and eats some Meat" becomes "Can Carnivore be considered a subcategory of Animal that is also something that eats Meat?". Finally, "(Bird and Fish) SubClassOf Animal" is translated to "Can Bird that is also Fish be considered a subcategory of Animal?".

Unexpected Responses Another aspect to consider is that, in principle, there are no constraints in the answer 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., temperature). Even with all these precautions, the model may still return an unexpected response:

1. the answer can have more text than just 'true' or 'false',
2. both 'true' and 'false' can appear in the answer, or
3. the answer does not have 'true' nor 'false'.

We avoided the first two cases by limiting the maximum number of tokens. We treated the last case as if the LLM had returned 'false'.

The Length of the Questions The implementation poses membership queries of the form "Can C be considered a subcategory of D?". For an ontology reasoner such as Hermit [21] or ELK [23], having complex expressions, with a number of conjuncts or a large number of nested existential quantifiers neither affects the behaviour of the reasoner nor the correctness of the responses, provided there are enough computational resources. However, this is not the case for LLMs. In our experiments, we observed that some LLMs tend to simply reply 'True' when the query gets long. To deal with this issue, we have modified the ExactLearner in the following ways:

- the first *simplifies* concept expressions by omitting concept names that are implied by other concept names in a concept expression,
- the second *splits* a long question with multiple conjuncts into several smaller questions.

That is, in the first case, if the concept expression is of the form, e.g., $A \sqcap B$ and the hypothesis of the learner entails $A \sqsubseteq B$ then one can think of B as being "redundant" and omitting it has no effect from the logical point of view, though, it does shorten the length of the question (see e.g. [5] for earlier works on concept reduction). More precisely, the *simplification* operation can be described as:

- Concept simplification for $\mathcal{H}$: if $\mathcal{H} \models C \sqsubseteq D$ and C, D are conjuncts of a node in a concept expression then we remove D from that label.

This operation is applied exhaustively to both sides of an inclusion and, even though it would work for arbitrary concepts C, D, it is currently only implemented for the case in which C, D are concept names. The simplification may happen in any node of a concept, including non-root nodes (e.g., $E \sqsubseteq \exists r.(C \sqcap D)$ would become $E \sqsubseteq \exists r.C$ and $\exists r.(C \sqcap D) \sqsubseteq E$ would become $\exists r.C \sqsubseteq E$). In the second case, rather than asking "Can C be considered a subcategory of D_1 and ... and D_n?", which would often happen as a result of apply concept saturation, the algorithm asks "Can C be considered a subcategory of D_1?", ..., "Can C be considered a subcategory of D_n?" and collects the answers to all these smaller questions. Since $\{C \sqsubseteq D_1 \sqcap \ldots \sqcap D_n\} \equiv \{C \sqsubseteq D_1, \ldots, C \sqsubseteq D_n\}$, having 'yes' to all of the smaller questions corresponds to 'yes' for the complex question. However, as e.g., $\{C \sqsubseteq \exists r.(D_1 \sqcap \ldots \sqcap D_n)\} \not\equiv \{C \sqsubseteq \exists r.D_1, \ldots, C \sqsubseteq \exists r.D_n\}$, this strategy cannot be applied to concept saturation on "non-root nodes". With these modifications, we mitigate the biased behaviour of LLMs answering 'true' to long queries.

Correctness and Logical Consistency We also need to deal with challenges regarding the correctness of the responses (assuming that the format of the responses returned by the language model is as expected). Actively learning ontologies has been investigated for various fragments of $\mathcal{EL}$ [25, 28, 13], 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 (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 [35]. So, we need to consider the following kinds of errors:

1. $C \sqsubseteq D$ is 'false' (cf. the real world) but the LLM answers 'true';
2. $C \sqsubseteq D$ is 'true' (cf. the real world) but the LLM answers 'false';
3. all concept inclusions in $\mathcal{O} = \{C_1 \sqsubseteq D_1, \ldots, C_n \sqsubseteq D_n\}$ are answered by the LLM with 'true', $\mathcal{O} \models C \sqsubseteq D$ but the LLM answers $C \sqsubseteq D$ as 'false'.

The last case is a logical inconsistency. Our strategy to handle this issue is to consider the *closure* under logical consequence [8]. That is, in Item 3, we consider $C \sqsubseteq D$ as 'true'. There is also no guarantee that an LLM will never change its answer. We deal with this using a cache system that keeps only one answer (the first) to a given query. Also, there is no perfect way for us to know what is true/false in the real world. We assume that the ontologies in the datasets used in the evaluation are mostly correct w.r.t. the real world.

Expressivity The algorithm will focus on extracting ontologies that are $\mathcal{EL}$ terminologies, while the knowledge we attempt to extract may be better expressed in richer languages. $\mathcal{EL}$ is a popular description logic, used as basis for many real-world ontologies in the medical domain [2, 1], meaning that interesting concept inclusions can still be expressed in this language. Previous works have explored the idea of extracting an approximation of an ontology in a richer language into Horn logic [10], which $\mathcal{EL}$ falls into as a special case. Some of the benefits of working with $\mathcal{EL}$ are the complexity of reasoning [6], which is in polynomial time, and the existence of tools that can efficiently perform automated reasoning in $\mathcal{EL}$ ontologies [23], a feature that we extensively use for building them.

4 Experiments

The code of the experiments is available on GitHub [4]. We conducted all experiments on a workstation equipped with two NVIDIA Tesla V100S-PCIE-32GB GPUs and dual Intel© Xeon© Gold 6226R CPUs (64 threads in total). The system was running CUDA 12.5 with NVIDIA driver version 535.183.01.

In more detail, we describe our evaluation approach. Algorithm 1 takes as input the signature (finite sets of concept and role names) and builds an $\mathcal{EL}$ terminology by posing membership and equivalence queries. As mentioned, we simulate equivalence queries with a sample of concept inclusions randomly generated and answered by an LLM, we refer to this as *PAC sample*. For a given ontology $\mathcal{O}$, with n logical axioms (concept inclusions and equivalences), the PAC sample size is $\ln(A^n/\delta)/\epsilon$, where $\delta = 0.1$, $\epsilon = 0.2$, and A is the number of all possible axioms of the form

$$A \sqsubseteq B, A \sqcap B \sqsubseteq C, B \sqsubseteq \exists r.A, \text{and } \exists r.A \sqsubseteq B$$

with $A, B, C \in \mathsf{N_C}$ and $r \in \mathsf{N_R}$, that can be formulated with $\mathsf{N_C}(\mathcal{O})$ and $\mathsf{N_R}(\mathcal{O})$. We call *normalized* the axioms in these 4 formats. We test the ability of ExactLearner+LLM to reconstruct existing ontologies by posing queries to LLMs.

[4] https://github.com/MatteoMagnini/ExactLearner-LLM/

Hypotheses In our experiments, we test the following hypotheses: **(1)** even though LLMs may make mistakes, and the simulation strategy does not guarantee exact learnability, we expect that there is a correlation between the ontologies built by ExactLearner+LLM and the original ontologies; **(2)** we expect that queries with our own parsing function to (controlled) natural language will give better results than just using the OWL Manchester syntax; **(3)** we expect that bigger LLMs will perform better than smaller ones; **(4)** we expect that ontologies expressing common knowledge such as family relations will be easier to learn than ontologies with specialized knowledge (such as medical ontologies); and **(5)** we expect that the simplification that we introduced (see Section 3, about the length of the questions) will improve the performance of the LLMs (and thus improve the performance of the ontologies constructed by our ExactLearner+LLM tool).

To test our hypotheses, we have considered the following datasets of ontologies, LLMs, and system prompts.

Datasets We first consider 5 small ontologies used as benchmark in the ExactLearner tool [13]. Then, we include 10 modules extracted from the GALEN ontology [2], with medical information. The ontologies in the first case are:

1. *Animals* contains knowledge related to the animal realm, including actual animals, subphyla, classes, orders, etc.;
2. *Cell* provides information about different cells based on their type, development stage and organism;
3. *Football* is a minimal ontology that describes the relations between the football game, the teams, the players and the manager;
4. *Generations* describes the members and relations within a family;
5. *University* is a small ontology, focusing on the professor role.

Table 1 presents the size of the signature (number of concept and role names) in these ontologies. We also have the number of logical axioms (concept inclusions and equivalences), PAC sample size, and the number of possible normalised axioms with the signature.

Table 1: Ontology statistics and PAC sample sizes with $\epsilon = 0.2$ and $\gamma = 0.1$. $\mathsf{N_C}$ and $\mathsf{N_R}$ are the number of concept and role names occurring in the ontologies.

Ontology	$\mathsf{N_C}$	$\mathsf{N_R}$	Log. Ax.	PAC Sample	Poss. Ax.
Animals	17	4	12	542	6,936
Cell	22	0	24	1,119	10,164
Football	10	3	9	341	1,500
Generations	20	4	18	847	10,800
University	7	3	4	139	588

As mentioned, we also perform experiments with modules extracted from the GALEN ontology. This is a large ontology, with $22,286$ concept names, 950 role names, and $46,558$ logical axioms. Even though the ontology is large, not all axioms are 'connected' to each other. That is, this ontology has information about diabetes, mental health, etc., and e.g. the axioms related to diabetes have no reference to those on mental health. In practice, one can separate GALEN into smaller ontologies, preserving the entailments. We use modularisation not only to tame the size of the PAC sample but also to select a signature with terms that are within the 'same topic' (e.g., the set of terms related to diabetes). To do so, we use the modularisation method described by Grau et al. (2008) with the bottom locality strategy [5]. This method generates $22,286$ modules, one for each concept name. From these, we randomly selected 10 modules after filtering them with the following criteria:

[5] https://github.com/ernestojimenezruiz/locality-module-extractor

- a module must have at least one concept name and one role name;
- a module cannot have more than 50 concept names or role names.

Our experiments are intended to give a proof concept on how one can work with large ontologies in our setting. However, to cover larger modules, we would need a better strategy, which we leave as future work. The modules used in the experiments are presented in Table 2 (module names are concept names used as seeds for the extraction).

Table 2: Ontology statistics and PAC sample sizes with $\epsilon = 0.2$ and $\gamma = 0.1$ for medical ontologies.

Ontology	N_C	N_R	Log. Ax.	PAC Sample	Pos. Ax.
Ab. Elb. J. C.	27	14	43	2,286	39,366
BNF Sec.	36	24	80	4,646	107,568
Chlorhexidine	23	14	38	1,946	26,450
Cone of Tissue	42	42	100	6,163	220,500
Kalli Krein	18	10	27	1,279	11,988
Neon	16	10	25	1,149	8,960
Pin	43	40	99	6,113	225,578
Pros. Drug	29	14	47	2,540	47,096
Zopiclone	32	36	77	4,465	105,472
Zuccini	33	22	58	3,295	82,764

LLMs, query format, and prompts We use 4 open LLMs: Mistral [15] with 7 billion parameters, Mixtral [22] with 47 billion parameters, Llama2 with 13 billion parameters and LLama3 with 8 billion parameters [31] (we use Ollama's API[6]). Also, we use two different formalisms for the queries: the Manchester OWL syntax and our parsing function to (controlled) *natural language* (see Section 3). Finally, we explore two different system prompts: a basic one and a more instructive one, called *advanced*, that provides contextual information to LLMs. For each combination of LLM, query format & prompt, and ontology, the computation time varied between a few minutes up to at most 2 h.

Simple system prompt

Answer with only True or False.

This first prompt is minimal; it simply asks the LLM to answer with a single word, that is, "True" or "False".

Advanced system prompt

You need to classify the following statements as True or False. The statement will be provided in either Manchester OWL syntax or natural language. Strictly follow these guidelines:
1. answer with only True or False;
2. entities with has part relation are not in a subclass relation;
3. statements or questions 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 provides contextual 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 a part of another concept name and should not be put in a subclass relation [18]. For instance, "Bird" has part "Wing" should not be in a subclass relationship as "Bird" $\sqsubseteq$ "Wing". Point 3 has the purpose to mitigate a behaviour we observed of LLMs answering "True" to long queries (that is, queries involving several concept names, role names, and logical operators). Long queries may appear when the learner tries to compute $\mathcal{O}$-essential concept inclusions (Section 2). Point 4 is an expedient found to be effective in improving LLM performance [34]. The last point is a remark to be conservative in the replies.

We also like to detail two shrewdnesses that we used in the experiments. First, in order to constrain the response of the LLM, we set the maximum number of tokens [7] equal to 2, since we are interested only in short answers, 'True' or 'False'. Second, we employ a cache mechanism to store the answers of the LLMs. This is crucial to reduce the computation time, in particular, because there are situations in which the algorithm would ask the same query (with the same prompt) multiple times, and it is also a way of dealing with the case in which the LLM would change the answer to the same query.

5 Evaluation and Results

We now present an evaluation of the results discussing whether our initial hypotheses could be confirmed or not. We evaluate the results of experiments by computing the following metrics: accuracy, precision, recall and F1-score. The metrics are computed considering all possible axioms of the form

$$A \sqsubseteq B, A \sqcap B \sqsubseteq C, B \sqsubseteq \exists r.A, \text{ and } \exists r.A \sqsubseteq B$$

that can be formulated with a finite signature. We remove tautologies, which can be created in our case with axioms of the form $A \sqsubseteq A$, $A \sqcap B \sqsubseteq B$, and $A \sqcap B \sqsubseteq A$ as they would trivially increase the number of true positives, giving a false impression of a better performance of the LLMs. Our language does not allow for the expression of contradictions. The axioms that are both (resp. not) entailed by the original ontology and the learnt ontology are labelled as true positive (resp. true negative). Axioms entailed by the original ontology but not by the learnt ontology are labelled as false negative, vice-versa are labelled as false positive.

5.1 Learning Performance

Looking into the Animals ontology learned from Mistral, for example, we see that it correctly learns that e.g. "Carnivore is an animal that eats some meat" just as in the original ontology. However, it also learns that "Carnivore is a mammal", whereas in the real world there are non-mammal carnivores. We also find that it learns that 'Carnivore is an animal that eats some animal", which is correct w.r.t. to the real world but not entailed by the ontology in the dataset, used as ground truth. Since it would be unfeasible to comment on all concept inclusions that were learned or not, in the following, we present aggregated results. Tables 3 to 5 contain the average results obtained by running ExactLearner+LLM on the small ontologies with all the LLMs, query formats, and prompt combinations.

Table 3: Results of ExactLearner+LLM grouped by ontologies.

Ontology	Accuracy	Recall	Precision	F1-Score
Animals	0.737	0.858	0.381	0.428
Cell	0.391	0.733	0.206	0.284
Football	0.553	0.89	0.422	0.477
Generations	0.691	0.658	0.564	0.476
University	0.622	0.629	0.313	0.302

In particular, Table 3 summarises the results grouped by ontologies. We can notice that different domains can affect the overall performance of the algorithm using an LLM as a teacher. The cell ontology, which is tailored in a more technical domain w.r.t. the other

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

[7] A token is a unit notion https://help.openai.com/en/articles/4936856. Usually it corresponds to a few characters – 1 token $\approx$ 4 characters – but the size can vary a bit (e.g., a token can be a single punctuation symbol, a short word, and so on). We need 2 tokens for the word 'False'.

Table 4: Results of ExactLearner+LLM grouped by models.

Model	Accuracy	Recall	Precision	F1-Score
Llama2 (13b)	0.521	0.71	0.294	0.314
Llama3 (8b)	0.43	0.947	0.218	0.333
Mistral (7b)	0.741	0.747	0.45	0.49
Mixtral (47b)	0.705	0.611	0.547	0.436

Table 5: Results of ExactLearner+LLM grouped by prompts.

Prompt Type	Accuracy	Recall	Precision	F1-Score
M. OWL Syntax	0.34	0.93	0.165	0.262
Natural Language	0.751	0.811	0.414	0.511
A. M. OWL Syntax	0.537	0.767	0.326	0.347
A. Natural Language	0.767	0.506	0.603	0.454

ontologies, has an average F1-score of 0.284. Ontologies tailored in non-technical domains have higher average F1-score (e.g., animals 0.428, football 0.477, generations 0.476). This confirms our **hypothesis (4)**. Table 4 shows the aggregated results from LLMs. We observe that the choice of the LLM can have a significant impact on the learning process. Mistral has the highest average F1-score (0.49); models from the Llama family have a much lower average F1-score (0.314 and 0.333). This does not confirm our **hypothesis (3)** because, e.g., Mistral is smaller than Mixtral. Finally, Table 5 reports the results grouped by query format & prompt type. We notice a significant performance gap between the prompts that use natural language (0.511 and 0.454 average F1-score) w.r.t. the ones that use Manchester OWL Syntax (0.262 and 0.347 average F1-score). This confirms our **hypothesis (2)**.

We showcase the (non-aggregated) results on an ontology where the learner achieves good performance (Table 6) and one where the performance is not good (Table 7). The remaining results for the other small ontologies are reported in the Appendix [26]. Table 6 shows the individual performance on the Animals ontology. The best F1-score (0.779) is obtained by Mixtral using the natural language prompt. Also Mistral and Llama2 can achieve high F1-score (i.e., more than 0.7) in some configurations. Table 7 shows the individual results on the Cell ontology, which has more technical symbols and worse performance.

For our experiments using modules extracted from the GALEN ontology, we select the best performing LLM according to Table 4 together with the advanced natural language prompt. We limit our experiments to only one LLM because the modules are considerably larger than the small ontologies and require more time to be learnt (see the PAC Axioms column in Tables 1 and 2). Table 8 shows the performance results. We can observe that some modules have a low F1-score value (e.g., Neon 0.253, Zopiclone 0.263) and others have higher values (e.g. Prostaglandin Drug 0.434, Kallikrein 0.414). Such heterogeneous results can be explained by the fact that the LLMs were not fine-tuned to answer questions in the medical domain. However, we point out that all experiments with these ontologies passed a Chi-Square correlation test. This confirms our **hypothesis (1)**.

To test our implementation on a teacher that perfectly answers membership queries, we also run ExactLearner+LLM using the synthetic teacher previously presented in the ExactLearner work [13]. Table 9 reports the statistics of the learnt ontologies (both the small ontologies and the modules). Because the synthetic teacher has full knowledge of the target ontology, the precision column is always 1. The other metrics do not necessarily reach 1, since our stopping criterium is based on PAC (with $\epsilon = 0.2$ and $\delta = 0.1$).

5.2 *Learner Behaviour*

We analyse the operations performed by the learner during the learning process. Figure 1 shows the percentages of the operations

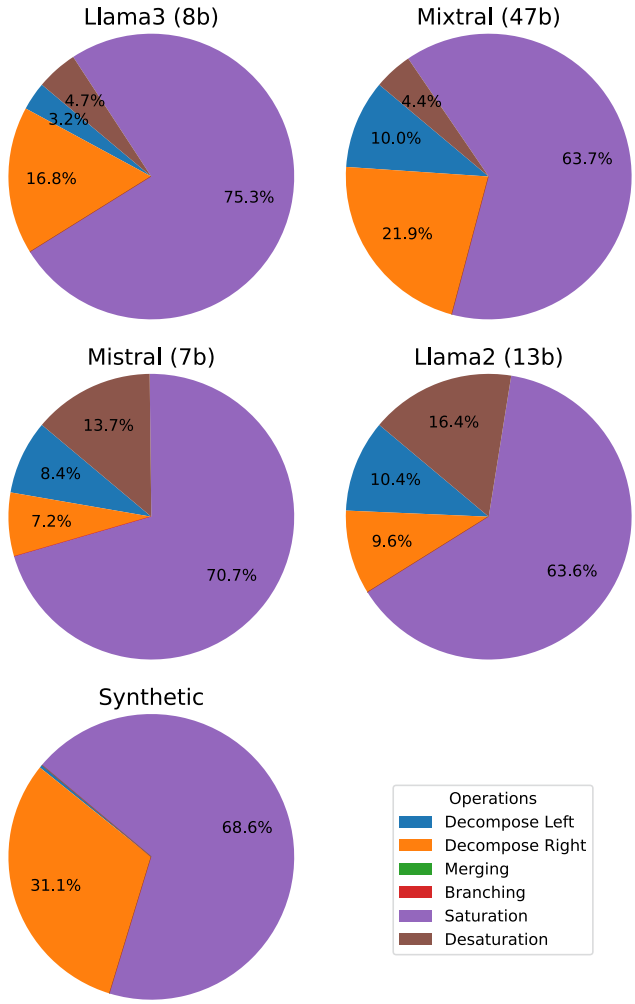

Figure 1: Aggregated results of the operations performed by the learner during the PAC learning of all the ontologies grouped by teacher type (LLMs and synthetic).

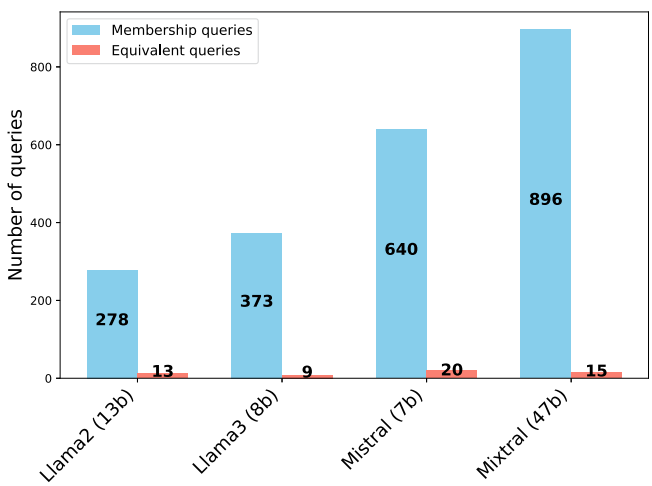

Figure 2: Average number of membership and (simulated) equivalence queries grouped by LLM. Each equivalence query is simulated by a batch of randomly generated queries.

Table 6: Individual results of ExactLearner+LLM on the Animals ontology. All the results have been analysed with the Chi-Squared test. The null hypothesis is that there is no correlation between the ontology constructed by ExactLearner+LLM and the Animals ontology. They all have p-value lower than 0.05 (i.e., refused null hypothesis).

Model	Prompt & Query	Accuracy	Recall	Precision	F1-Score
Llama2 (13b)	M. OWL Syntax	0.276	0.982	0.074	0.138
	Natural Language	0.859	0.930	0.286	0.437
	E. M. OWL Syntax	0.380	0.945	0.083	0.152
	E. Natural Language	0.970	0.651	0.801	0.718
Llama3 (8b)	M. OWL Syntax	0.309	1.000	0.078	0.145
	Natural Language	0.920	0.890	0.414	0.565
	E. M. OWL Syntax	0.314	0.985	0.078	0.145
	E. Natural Language	0.894	0.842	0.339	0.483
Mistral (7b)	M. OWL Syntax	0.552	0.996	0.116	0.207
	Natural Language	0.955	0.912	0.575	0.706
	E. M. OWL Syntax	0.830	0.945	0.250	0.396
	E. Natural Language	0.943	0.890	0.511	0.649
Mixtral (47b)	M. OWL Syntax	0.712	0.904	0.158	0.269
	Natural Language	0.970	0.893	0.690	0.779
	E. M. OWL Syntax	0.960	0.739	0.636	0.684
	E. Natural Language	0.955	0.228	1.000	0.371

Table 7: Individual results of ExactLearner+LLM on the Cell ontology. All the results have been analysed with the Chi-Squared test. The null hypothesis is that there is no correlation between the ontology constructed by ExactLearner+LLM and the Cell ontology. A yellow line means that the p-value is greater than 0.05 (i.e., the null hypothesis cannot be refused). The red line means that no axiom has been learnt (i.e., empty ontology).

Model	Prompt & Query	Accuracy	Recall	Precision	F1-Score
Llama2 (13b)	M. OWL Syntax	0.163	1.0	0.163	0.280
	Natural Language	0.586	0.715	0.241	0.360
	E. M. OWL Syntax	0.163	1.0	0.163	0.280
	E. Natural Language	0.830	0.002	0.049	0.005
Llama3 (8b)	M. OWL Syntax	0.163	1.0	0.163	0.280
	Natural Language	0.208	1.0	0.171	0.292
	E. M. OWL Syntax	0.208	1.0	0.171	0.292
	E. Natural Language	0.254	1.0	0.179	0.304
Mistral (7b)	M. OWL Syntax	0.254	1.0	0.179	0.304
	Natural Language	0.779	0.432	0.354	0.390
	E. M. OWL Syntax	0.224	0.946	0.167	0.284
	E. Natural Language	0.769	0.337	0.308	0.322
Mixtral (47b)	M. OWL Syntax	0.163	1.0	0.163	0.280
	Natural Language	0.654	0.970	0.317	0.477
	E. M. OWL Syntax	0.840	0.320	0.514	0.394
	E. Natural Language	-	-	-	-

grouped by LLMs and synthetic teacher. We observe that the LLMs share a similar pattern. Saturation is the most frequent operation and occurs more than half the time. Then, the remaining operations performed by the learner are decompose right, decompose left and desaturation. Instead, while using the synthetic teacher, the learner only performs saturation and right decomposition. We also report in Figure 2 the average number of membership and (simulated) equivalence queries made by the learner grouped by LLM. The LLM plays an important role in affecting the average number of queries that the learner poses during the learning process.

The combination of the simplification and splitting operations improved the results of the F1-score for most combinations of LLMs plus prompt. This confirmed our **hypothesis (5)**. This was particularly significant for the Mistral model with the NLP prompt, where the F1-score went from approx 0.5 to approx 0.7. We report an ablation study of the simplification and splitting operations for all the combinations of LLM and prompt in the small ontologies of the ExactLearner benchmark in the Appendix [26].

Table 8: Results of the Exact Learner using Mistral with the advanced system prompt and the natural language query format on the modules of Galen. The orange colour line means that we ran out of memory.

Ontology	Accuracy	Recall	Precision	F1-Score
Above Elbow Jacket Cast	0.86	0.185	0.441	0.261
BNF Section 13.11	0.834	0.394	0.341	0.365
Chlorhexidine	0.891	0.199	0.522	0.288
Cone of Tissue	-	-	-	-
Kallikrein	0.877	0.283	0.771	0.414
Neon	0.868	0.163	0.564	0.253
Pin	0.87	0.482	0.204	0.287
Prostaglandin Drug	0.887	0.36	0.546	0.434
Zopiclone	0.923	0.233	0.303	0.263
Zuccini	0.915	0.182	0.588	0.278

Table 9: Results of ExactLearner+LLM with the synthetic teacher on small ontologies and modules extracted from Galen.

Ontology	Accuracy	Recall	Precision	F1-Score
Animals	0.997	0.941	1.0	0.97
Cell	1.0	1.0	1.0	1.0
Football	0.995	0.969	1.0	0.984
Generations	0.977	0.859	1.0	0.924
University	0.984	0.794	1.0	0.885
Above Elbow Jacket Cast	0.991	0.932	1.0	0.965
BNF Section 13.11	0.992	0.935	1.0	0.967
Chlorhexidine	0.993	0.936	1.0	0.967
Cone of Tissue	0.996	0.931	1.0	0.964
Kallikrein	0.994	0.962	1.0	0.981
Neon	1.0	1.0	1.0	1.0
Pin	0.997	0.94	1.0	0.969
Prostaglandin Drug	0.993	0.941	1.0	0.97
Zopiclone	0.995	0.916	1.0	0.956
Zuccini	0.993	0.918	1.0	0.957

6 Conclusion and Discussion

We present *ExactLearner+LLM*, a novel extension of the ExactLearner tool that exploits an LLM as a teacher to learn $\mathcal{EL}$ ontologies. During the *active learning* process, we employ the PAC learning framework to simulate *equivalence queries*. We adapt tool to deal with the challenges posed by the use of an LLM as a teacher, such as the *input format* of the queries, the handling of *unexpected responses*, and possible lack of *logical consistency*.

We introduce mechanisms to shorten the length of the query (simplification and splitting), to tail our solution to the behaviour of LLMs. Finally, we perform extensive experiments with different ontologies and LLMs to validate our tool. The results indicate that, in most cases, there is statistical evidence of correlation between the extracted concept inclusions and those on the original ontologies.

In summary, the present findings provide valuable insight into the performance of various models. Key observations include:

1. concept inclusions of the form $A \sqsubseteq B$ are much easier to learn than $A \sqsubseteq \exists r.B$ or $\exists r.B \sqsubseteq A$, with $A, B \in \mathsf{N_C}$, so ontologies with proportionally more of the two latter types of concept inclusions had worse results;
2. operations such as saturation, desaturation and right decomposition were more successfully applied;
3. Mistral has superior performance w.r.t. other models in our experiments, including the larger Mixtral model;
4. across almost all ontologies, models achieve better performance when queries are expressed with a natural language prompt, customized system prompts also significantly changed the results.

The results were obtained without proper training or fine-tuning of the models, with zero-shot good learning performance in most cases. In contrast to a simple generative approach to create ontologies, our approach is guaranteed to build a terminology in $\mathcal{EL}$, with the desired vocabulary. As future work, it would be interesting to investigate more efficient ways of finding counterexamples than random sampling, which could improve scalability. Also, we would like to improve on the expressivity of the ontology language that we consider. Finally, for ontologies with technical terms, such as medical ontologies, we would like to explore LLMs tailored to answer questions on these specific domains.

Acknowledgements Ozaki is supported by the Research Council of Norway, projects 316022, 322480. This work was partly supported by the Research Council of Norway through its Centre of Excellence Integreat - The Norwegian Centre for knowledge-driven machine learning, project 332645.

References

[1] Bioportal. http://bioportal.bioontology.org. Accessed: 2025-05-03.

[2] P. P. Alan L. Rector, J.E. Rogers. The galen high level ontology. In *Medical Informatics Europe*, page 174–178. IOS Press, 1996. doi: 10.3233/978-1-60750-878-6-174.

[3] D. Angluin. Queries and concept learning. *Mach. Learn.*, 2(4):319–342, 1987. doi: 10.1007/BF00116828.

[4] D. Angluin. Computational learning theory: Survey and selected bibliography. In S. R. Kosaraju, M. Fellows, A. Wigderson, and J. A. Ellis, editors, *Proceedings of the 24th Annual ACM Symposium on Theory of Computing*, pages 351–369. ACM, 1992. doi: 10.1145/129712.129746.

[5] F. Baader, R. Küsters, and R. Molitor. Rewriting concepts using terminologies. In A. G. Cohn, F. Giunchiglia, and B. Selman, editors, *KR*, pages 297–308. Morgan Kaufmann, 2000.

[6] F. Baader, S. Brandt, and C. Lutz. Pushing the EL envelope. In L. P. Kaelbling and A. Saffiotti, editors, *IJCAI-05, Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence, Edinburgh, Scotland, UK, July 30 - August 5, 2005*, pages 364–369. Professional Book Center, 2005. URL http://ijcai.org/Proceedings/05/Papers/0372.pdf.

[7] F. Baader, I. Horrocks, C. Lutz, and U. Sattler. *An Introduction to Description Logic*. Cambridge University Press, 2017. ISBN 978-0-521-69542-8. URL https://dl.acm.org/doi/book/10.5555/3158709.

[8] S. Blum, R. Koudijs, A. Ozaki, and S. Touileb. Learning horn envelopes via queries from language models. *Int. J. Approx. Reason.*, 171:109026, 2024. doi: 10.1016/J.IJAR.2023.109026.

[9] S. Blum, R. Koudijs, A. Ozaki, and S. Touileb. Learning horn envelopes via queries from language models. *Int. J. Approx. Reason.*, 171:109026, 2024. doi: 10.1016/J.IJAR.2023.109026.

[10] A. Bötcher, C. Lutz, and F. Wolter. Ontology approximation in horn description logics. In S. Kraus, editor, *IJCAI*, pages 1574–1580. ijcai.org, 2019. doi: 10.24963/IJCAI.2019/218.

[11] J. H. Caufield, H. Hegde, V. Emonet, N. L. Harris, M. P. Joachimiak, N. Matentzoglu, H. Kim, S. Moxon, J. T. Reese, M. A. Haendel, P. N. Robinson, and C. J. Mungall. Structured prompt interrogation and recursive extraction of semantics (spires): a method for populating knowledge bases using zero-shot learning. *Bioinformatics*, 40(3):btae104, 2024. ISSN 1367-4811. doi: 10.1093/bioinformatics/btae104.

[12] G. Ciatto, A. Agiollo, M. Magnini, and A. Omicini. Large language models as oracles for instantiating ontologies with domain-specific knowledge. *Knowl. Based Syst.*, 310:112940, 2025. doi: 10.1016/J.KNOSYS.2024.112940.

[13] M. R. C. Duarte, B. Konev, and A. Ozaki. Exactlearner: A tool for exact learning of EL ontologies. In M. Thielscher, F. Toni, and F. Wolter, editors, *KR*, pages 409–414. AAAI Press, 2018. URL https://aaai.org/ocs/index.php/KR/KR18/paper/view/18006.

[14] D. Duranti, P. Giorgini, A. Mazzullo, M. Robol, and M. Roveri. Llm-driven knowledge extraction in temporal and description logics. In M. Alam, M. Rospocher, M. van Erp, L. Hollink, and G. A. Gesese, editors, *EKAW*, volume 15370 of *Lecture Notes in Computer Science*, pages 190–208. Springer, 2024. doi: 10.1007/978-3-031-77792-9_12.

[15] A. Q. J. et al. Mistral 7b. *CoRR*, abs/2310.06825, 2023. doi: 10.48550/ARXIV.2310.06825.

[16] M. Funk, J. C. Jung, C. Lutz, H. Pulcini, and F. Wolter. Learning description logic concepts: When can positive and negative examples be separated? In S. Kraus, editor, *IJCAI*, pages 1682–1688. ijcai.org, 2019. doi: 10.24963/IJCAI.2019/233.

[17] M. Funk, J. C. Jung, and C. Lutz. Actively learning concepts and conjunctive queries under elr-ontologies. In Z. Zhou, editor, *IJCAI*, pages 1887–1893. ijcai.org, 2021. doi: 10.24963/IJCAI.2021/260.

[18] M. Funk, S. Hosemann, J. C. Jung, and C. Lutz. Towards ontology construction with language models. In S. Razniewski, J. Kalo, S. Singhania, and J. Z. Pan, editors, *(KBC-LM) and (LM-KBC) co-located with ISWC*, volume 3577 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2023. URL https://ceur-ws.org/Vol-3577/paper16.pdf.

[19] B. C. Grau, I. Horrocks, Y. Kazakov, and U. Sattler. Modular reuse of ontologies: Theory and practice. *J. Artif. Intell. Res.*, 31:273–318, 2008. doi: 10.1613/JAIR.2375.

[20] M. Horridge, N. Drummond, J. Goodwin, A. L. Rector, R. Stevens, and H. Wang. The manchester OWL syntax. In B. C. Grau, P. Hitzler, C. Shankey, and E. Wallace, editors, *OWLED*, volume 216 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2006. URL https://ceur-ws.org/Vol-216/submission_9.pdf.

[21] I. Horrocks, B. Motik, and Z. Wang. The hermit OWL reasoner. In I. Horrocks, M. Yatskevich, and E. Jiménez-Ruiz, editors, *ORE*, volume 858 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2012. URL https://ceur-ws.org/Vol-858/ore2012_paper13.pdf.

[22] A. Q. Jiang, A. Sablayrolles, A. Roux, A. Mensch, B. Savary, C. Bamford, D. S. Chaplot, D. de Las Casas, E. B. Hanna, F. Bressand, G. Lengyel, G. Bour, G. Lample, L. R. Lavaud, L. Saulnier, M. Lachaux, P. Stock, S. Subramanian, S. Yang, S. Antoniak, T. L. Scao, T. Gervet, T. Lavril, T. Wang, T. Lacroix, and W. E. Sayed. Mixtral of experts. *CoRR*, abs/2401.04088, 2024. doi: 10.48550/ARXIV.2401.04088.

[23] Y. Kazakov, M. Krötzsch, and F. Simancik. The incredible ELK - from polynomial procedures to efficient reasoning with EL ontologies. *J. Autom. Reason.*, 53(1):1–61, 2014. doi: 10.1007/S10817-013-9296-3.

[24] B. Konev, A. Ozaki, and F. Wolter. A model for learning description logic ontologies based on exact learning. In D. Schuurmans and M. P. Wellman, editors, *AAAI*, pages 1008–1015. AAAI Press, 2016. doi: 10.1609/AAAI.V30I1.10087.

[25] B. Konev, C. Lutz, A. Ozaki, and F. Wolter. Exact learning of lightweight description logic ontologies. *J. Mach. Learn. Res.*, 18:201:1–201:63, 2017. URL http://jmlr.org/papers/v18/16-256.html.

[26] M. Magnini, R. Squarcialupi, M. T. Sterri, and A. Ozaki. Actively learning el terminologies from large language models (extended version), Aug. 2025. URL https://doi.org/10.5281/zenodo.16950849.

[27] N. F. Noy, N. Shah, B. Dai, M. Dorf, N. Griffith, C. Jonquet, M. Montegut, D. L. Rubin, C. Youn, and M. A. Musen. Bioportal: A web repository for biomedical ontologies and data resources. In C. Bizer and A. Joshi, editors, *ISWC*, volume 401 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2008. doi: 10.1017/9781139025355.

[28] A. Ozaki, C. Persia, and A. Mazzullo. Learning query inseparable ELH ontologies (extended abstract). In S. Borgwardt and T. Meyer, editors, *DL*, volume 2663 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2020. doi: 10.1609/AAAI.V34I03.5688.

[29] S. Shalev-Shwartz and S. Ben-David. *Understanding Machine Learning - From Theory to Algorithms*. Cambridge University Press, 2014. ISBN 978-1-10-705713-5. doi: 10.1017/CBO9781107298019.

[30] V. M. A. Soares and R. Wassermann. Ontology extraction and evaluation for the blue amazon. In C. M. Fonseca and J. L. Emygdio, editors, *ONTOBRAS*, volume 3905 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2024. URL https://ceur-ws.org/Vol-3905/master5.pdf.

[31] H. Touvron et al. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288, 2023. doi: 10.48550/arxiv.2307.09288.

[32] L. G. Valiant. A theory of the learnable. *Commun. ACM*, 27(11):1134–1142, 1984. doi: 10.1145/1968.1972.

[33] G. Weiss, Y. Goldberg, and E. Yahav. Extracting automata from recurrent neural networks using queries and counterexamples (extended version). *Mach. Learn.*, 113(5):2877–2919, 2024. doi: 10.1007/S10994-022-06163-2.

[34] C. Yang, X. Wang, Y. Lu, H. Liu, Q. V. Le, D. Zhou, and X. Chen. Large language models as optimizers. In *ICLR*. OpenReview.net, 2024. URL https://openreview.net/forum?id=Bb4VGOWELI.

[35] H. Zhang, L. H. Li, T. Meng, K. Chang, and G. V. den Broeck. On the paradox of learning to reason from data. In *IJCAI*, pages 3365–3373. ijcai.org, 2023. doi: 10.24963/IJCAI.2023/375.