ROCES: Robust Class Expression Synthesis in Description Logics via Iterative Sampling

N'Dah Jean Kouagou , **Stefan Heindorf** ,
Caglar Demir and **Axel-Cyrille Ngonga Ngomo**
Data Science Research Group, Paderborn University, Germany
{ndah.jean.kouagou, heindorf, caglar.demir, axel.ngonga}@upb.de

Abstract

We consider the problem of class expression learning using cardinality-minimal sets of examples. Recent class expression learning approaches employ deep neural networks and have demonstrated tremendous performance improvements in execution time and quality of the computed solutions. However, they lack generalization capabilities when it comes to the number of examples used in a learning problem, i.e., they often perform poorly on unseen learning problems where only a few examples are given. In this work, we propose a generalization of the classical class expression learning problem to address the limitations above. In short, our generalized learning problem (GLP) forces learning systems to solve the classical class expression learning problem using the smallest possible subsets of examples, thereby improving the learning systems' ability to solve unseen learning problems with arbitrary numbers of examples. Moreover, we develop ROCES, a learning algorithm for synthesis-based approaches to solve GLP. Experimental results suggest that post training, ROCES outperforms existing synthesis-based approaches on out-of-distribution learning problems while remaining highly competitive overall.

1 Introduction

Class expression learning (CEL) in description logics (DLs) refers to the task of finding a class expression that describes a given set of (positive) examples. A class expression is human-readable and interpretable and is hence a white-box model. For instance, a class expression to describe penguins is $\texttt{Bird} \sqcap \texttt{CanSwim}) \sqcup (\texttt{Animal} \sqcap \texttt{HasWings} \sqcap \neg\texttt{CanFly})$, which stands for *birds that can swim or animals that have wings and cannot fly*. Note that this expression might also describe other birds that are not penguins, e.g., ostriches. To describe penguins exclusively, one can further refine the expression above, e.g., by using data values to specify their maximum leg length or maximum weight. In bio-medicine, class expressions can be used to formally describe diseases or chemical compounds [Hartel *et al.*, 2005; Schulz *et al.*, 2008; Boeker *et al.*, 2016] and facilitate further investigations by domain experts. Other application domains of CEL include ontology engineering [Lehmann *et al.*, 2011], and Industry 4.0 [Demir *et al.*, 2022].

Early approaches for CEL are search-based [Lehmann and Hitzler, 2010; Lehmann, 2010; Lehmann *et al.*, 2011; Rizzo *et al.*, 2020]. These approaches employ a refinement operator [Badea and Nienhuys-Cheng, 2000; Lehmann and Hitzler, 2007] to construct an infinite conceptual space, and a heuristic function to traverse the latter. Consequently, search-based approaches are time-inefficient and often yield poor solutions—especially on large datasets [Kouagou *et al.*, 2023a; Kouagou *et al.*, 2023b]. Other types of approaches have hence been proposed to overcome the limitations of search-based approaches. These include neural class expression synthesizers [Kouagou *et al.*, 2023a; Kouagou *et al.*, 2023b], approaches based on reinforcement learning (e.g., DRILL) [Demir and Ngonga Ngomo, 2023], pruning-based approaches (e.g., CLIP) [Kouagou *et al.*, 2022], meta-learners (e.g., EvoLearner) [Heindorf *et al.*, 2022], and sampling-based approaches [Baci and Heindorf, 2023]. The latter employs sampling algorithms to construct a subset of the input knowledge base which then serves as the background knowledge for the considered learning problem. EvoLearner is based on evolutionary algorithms and initializes its population (class expressions) by random walks on the input knowledge base. CLIP steers the search space of search-based approaches by using concept length predictors which are initially trained in an unsupervised manner. DRILL learns to optimally traverse the search space via deep Q-learning [Mnih *et al.*, 2013]. Neural class expression synthesizers use deep neural networks to learn mappings between sets of examples and class expressions without a search process. They have proven to be hundreds of times faster than all approaches mentioned above while maintaining the quality of the computed solutions. This makes synthesis-based approaches suitable for web-scale applications of CEL.

Despite their scalability, the performance of synthesis-based approaches remains highly dependent on their training data which, in fact, is generated in a self-supervised manner. The training data consists of non-redundant arbitrary class expressions coupled with their sets of positive/negative examples. Positive examples for a class expression are a subset of its instances. Likewise, negative examples for a class expression are a subset of individuals that are not instances

of that particular class expression. For the sake of scalability and to enable support for batched inputs, synthesis-based approaches require a fixed number N which represents the maximum number of examples per class expression. Hence, the generated training data induces a probability distribution over the sizes of the sets of positive and negative examples. This distribution is inevitably injected into the trained neural class expression synthesizers. As we show in our experiments, synthesis-based approaches perform poorly on learning problems with small sets of examples, which are most likely to be encountered in practical applications of CEL. For instance, in ontology engineering, knowledge engineers manually inspect and label a few individuals at a time. A naive way to improve neural class expression synthesizers in this regard would be to set the maximum number of examples N to a low value. However, this would fail because many positive/negative examples would be omitted, thereby leading to poorly trained neural synthesizers.

In this paper, we propose a new synthesis-based approach dubbed ROCES that can solve learning problems with arbitrary numbers of examples. ROCES has the same expressive power and scalability capability as NCES2 [Kouagou *et al.*, 2023a], but enjoys a better predictive performance on out-of-distribution learning problems. In a nutshell, our contributions are:

1. We propose a generalization of the classical learning problem that aims to improve the robustness of learning systems w.r.t. the number of input examples. Our new formulation constrains learning systems to find and use a subset of the provided examples to solve a given learning problem. If such a subset cannot be found, the learning system then uses all examples as in the classical case.
2. We establish connections between the classical learning problem and our generalized learning problem ($\mathcal{GLP}$) in terms of solution existence.
3. We develop a learning algorithm to solve $\mathcal{GLP}$. The algorithm is applicable to any neural synthesis-based approach. In our experiments, we apply it to the approach in [Kouagou *et al.*, 2023a] because it is the best synthesis-based approach to date.
4. We provide the source code, pretrained models, and open-access datasets for reproducible research.[1]

2 Preliminaries

This paper exploits techniques from different well-established fields of research, including description logics, deep learning, and knowledge graphs. In this section, we briefly present the prerequisites needed throughout the paper.

2.1 Description Logics

Description logics [Nardi *et al.*, 2003] are a family of languages for formal representation of knowledge in a variety of application domains such as artificial intelligence, bio-informatics, the semantic web, and automated reasoning. They are generally more expressive than propositional logic [Büning and Lettmann, 1999] but less expressive than first-order logic [Barwise, 1977]. Following [Kouagou *et al.*, 2023a], our approach targets the description logic $\mathcal{ALCHIQ}(\mathcal{D})$ whose syntax and semantics are given in Table 1.

Syntax	Construct	Semantics
	$\mathcal{ALC}$	
r	abstract role	$r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
$\top$	top concept	$\Delta^{\mathcal{I}}$
$\bot$	bottom concept	$\emptyset$
C	atomic concept	$C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
$\neg C$	negation	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
$C \sqcup D$	disjunction	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
$C \sqcap D$	conjunction	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
$\exists\, r.C$	existential restriction	$\{a^{\mathcal{I}} \in \Delta^{\mathcal{I}} \mid \exists\, b^{\mathcal{I}} \in C^{\mathcal{I}}, (a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}\}$
$\forall\, r.C$	universal restriction	$\{a^{\mathcal{I}} \in \Delta^{\mathcal{I}} \mid \forall\, b^{\mathcal{I}}, (a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}} \Rightarrow b^{\mathcal{I}} \in C^{\mathcal{I}}\}$
	$\mathcal{H}$	
$r_1 \sqsubseteq r_2$	abstract role hierarchy	$r_1^{\mathcal{I}} \subseteq r_2^{\mathcal{I}}$
	$\mathcal{I}$	
r^-	inverse abstract role	$\{(b^{\mathcal{I}}, a^{\mathcal{I}}) \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \mid (a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}\}$
	$\mathcal{Q}$	
$\leq n\, r.C$	max. card. restriction	$\{a^{\mathcal{I}} \in \Delta^{\mathcal{I}} \mid \lvert\{b^{\mathcal{I}} \in C^{\mathcal{I}} : (a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}\}\rvert \leq n\}$
$\geq n\, r.C$	min. card. restriction	$\{a^{\mathcal{I}} \in \Delta^{\mathcal{I}} \mid \lvert\{b^{\mathcal{I}} \in C^{\mathcal{I}} : (a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}\}\rvert \geq n\}$
	$(\mathcal{D})$	
d	concrete role	$d^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \mathcal{D}$
$d = v$	exact value restriction	$\{a^{\mathcal{I}} \in \Delta^{\mathcal{I}} \mid (a^{\mathcal{I}}, v) \in d^{\mathcal{I}}\}$
$d \leq v$	max. restriction	$\{a^{\mathcal{I}} \in \Delta^{\mathcal{I}} \mid \exists\, w \in \mathcal{D}, (a^{\mathcal{I}}, w) \in d^{\mathcal{I}} \wedge w \leq v\}$
$d \geq v$	min. restriction	$\{a^{\mathcal{I}} \in \Delta^{\mathcal{I}} \mid \exists\, w \in \mathcal{D}, (a^{\mathcal{I}}, w) \in d^{\mathcal{I}} \wedge w \geq v\}$

Table 1: Syntax and semantics of $\mathcal{ALCHIQ}(\mathcal{D})$. $\mathcal{I} = (\Delta^I, \cdot^{\mathcal{I}})$ is an interpretation where $\Delta^{\mathcal{I}}$ is its domain and $\cdot^{\mathcal{I}}$ is the interpretation function. $\mathcal{D}$ denotes the set of data values, e.g., strings, numbers, Boolean values, etc.

2.2 Refinement Operators

Definition 1 ([Lehmann and Hitzler, 2010; Kouagou *et al.*, 2023b]). *Given a quasi-ordered space $(\mathcal{S}, \preceq)$, a downward (respectively upward) refinement operator on $\mathcal{S}$ is a mapping $\rho : \mathcal{S} \rightarrow 2^{\mathcal{S}}$ such that for all $C \in \mathcal{S}$, $C' \in \rho(C)$ implies $C' \preceq C$ (respectively $C \preceq C'$).*

Search-based approaches, e.g. CELOE, employ a refinement operator and a heuristic function to construct and traverse their search space. In contrast, our proposed approach only needs a refinement operator to generate its training data.

2.3 Class Expression Learning

We now give the classical definition of CEL in description logics (Definition 2), and propose a generalized version thereof (Definition 3) in Section 3.

Definition 2 (Classical Learning Problem ($\mathcal{CLP}$)). *Given a knowledge base $\mathcal{K}$, a target concept T, a set of positive examples $E^+ = \{e_1^+, e_2^+, \ldots, e_{n_1}^+\}$, and a set of negative examples $E^- = \{e_1^-, e_2^-, \ldots, e_{n_2}^-\}$, the learning problem is to find a class expression C such that T does not occur in C and for $\mathcal{K}'_C = \mathcal{K} \cup \{T \equiv C\}$, we have that $\mathcal{K}'_C \models C(E^+)$ and $\mathcal{K}'_C \not\models C(E^-)$.*

[1] https://github.com/dice-group/ROCES

We write $\mathcal{K}'_C \models C(E)$ (respectively, $\mathcal{K}'_C \not\models C(E)$) to express that $\forall\ e \in E,\ \mathcal{K}'_C \models C(e)$ (respectively, $\forall\ e \in E,\ \mathcal{K}'_C \not\models C(e)$). Here, $\mathcal{K}'_C \models C(e)$ means e *is an instance of* C *according to* $\mathcal{K}'_C$. We denote the classical learning problem defined by $\mathcal{K}$, T, E^+, and E^- by $\mathcal{CLP}(\mathcal{K}, T, E^+, E^-)$. Our proposed approach uses continuous vector representations of examples to compute a solution. These representations are provided by a knowledge graph embedding model—a notion that we elucidate in the next subsection.

2.4 Knowledge Graph Embedding

A Knowledge graph (KG) is a "graph of data intended to accumulate and convey knowledge of the real world, whose nodes represent entities of interest and whose edges represent relations between these entities" [Hogan *et al.*, 2022]. They can also be defined as collections of assertions in the form of triples. To facilitate downstream applications, KGs are often projected onto continuous vector spaces such as $\mathbb{R}^d$. This process is known as *knowledge graph embedding*.

Several KG embedding techniques have been developed ever since the introduction of the pioneering approaches. Roughly, embeddings are computed by learning the vector representations of nodes and relation types in a way that the relationships between nodes can be established by using the learned vectors. KG embeddings are useful for downstream tasks such as link prediction [Bordes *et al.*, 2013], knowledge completion [Lin *et al.*, 2015], recommendation systems [Zhang *et al.*, 2016], and natural language processing [Chen and Zaniolo, 2017]. Some KG embedding approaches solely use facts observed in the input knowledge graph [Nickel *et al.*, 2012; Bordes *et al.*, 2014] while others leverage additional information about entities and relations, such as textual descriptions [Xie *et al.*, 2016; Wang and Li, 2016] or `sameAs` links to external sources [Kouagou *et al.*, 2024]. Our proposed neural network-based approach for CEL uses the embedding method ConEx [Demir and Ngonga Ngomo, 2021] whose specificity is to apply convolutions on complex-valued vector representations of nodes and relation types to model interactions. We chose this model because it was compared to other models for CEL and was found to be the most efficient, see for example [Kouagou *et al.*, 2023b].

2.5 Set Transformer

One desired property for the approach we aim to build in this paper is *permutation invariance* w.r.t. the input examples. In other words, the prediction of our approach must remain the same for any re-ordering of the input examples. Set Transformer [Lee *et al.*, 2019], which we use in ROCES, is a permutation-invariant deep learning architecture which can also capture pairwise and higher-order interactions between the elements of the input set via a self-attention mechanism. Set Transformer achieves state-of-the-art results on set-input tasks such as *maximum value regression, unique character counting*, and *point cloud classification*. The Multi-head Attention Block (MAB), the Set Attention Block (SAB), the Induced Set Attention Block (ISAB), and the Pooling by Multi-head Attention (PMA) are the building blocks of Set Transformer. For more details, we refer to the original paper [Lee *et al.*, 2019].

3 Iterative Sampling for Class Expression Learning

We first give a modification of Definition 2 that encourages learning systems to compute solutions without using all the provided examples. Second, we establish the connections between classical solutions and those from our new formulation. Finally, we describe our learning algorithm in detail and discuss potential limitations.

3.1 Generalized Learning Problem

Our proposed definition below is motivated by the fact that in real-world applications of CEL, one deals with small sets of examples (see an example in the introduction). In this context, a good learning system should be able to compute a solution that is as specific as possible but general enough to cover other relevant examples that are not given. In the next paragraphs, we denote the set of all solutions to the classical learning problem $\mathcal{CLP}(\mathcal{K}, T, E^+, E^-)$ by $\mathcal{S}_{\mathcal{CLP}}(\mathcal{K}, T, E^+, E^-)$.

Definition 3 (Generalized Learning Problem ($\mathcal{GLP}$))**.** *Given a knowledge base $\mathcal{K}$, a target concept T, and sets of positive/negative examples $E^+ = \{e_1^+, e_2^+, \ldots, e_{n_1}^+\}$ and $E^- = \{e_1^-, e_2^-, \ldots, e_{n_2}^-\}$, the learning problem is to find non-empty subsets $\mathcal{E}^+ \subseteq E^+$, $\mathcal{E}^- \subseteq E^-$ with the following properties*

1. $\mathcal{S}_{\mathcal{CLP}}(\mathcal{K}, T, \mathcal{E}^+, \mathcal{E}^-) \neq \emptyset$
2. $\mathcal{S}_{\mathcal{CLP}}(\mathcal{K}, T, \mathcal{E}^+, \mathcal{E}^-) \subseteq \mathcal{S}_{\mathcal{CLP}}(\mathcal{K}, T, E^+, E^-)$
3. *There do not exist non-empty subsets $\mathcal{E}'^+ \subseteq E^+$, $\mathcal{E}'^- \subseteq E^-$ such that $|\mathcal{E}'^+| + |\mathcal{E}'^-| < |\mathcal{E}^+| + |\mathcal{E}^-|$ and $\emptyset \neq \mathcal{S}_{\mathcal{CLP}}(\mathcal{K}, T, \mathcal{E}'^+, \mathcal{E}'^-) \subseteq \mathcal{S}_{\mathcal{CLP}}(\mathcal{K}, T, E^+, E^-)$,*

where $|.|$ denotes the cardinality of a set. Properties *1.* and *2.* in the above definition require that the set of all solutions (class expressions) to the learning problem $\mathcal{CLP}(\mathcal{K}, T, \mathcal{E}^+, \mathcal{E}^-)$ is non-empty and each of its elements is a solution to $\mathcal{CLP}(\mathcal{K}, T, E^+, E^-)$. Property *3.* further ensures that such subsets $\mathcal{E}^-$, $\mathcal{E}^+$ are minimal in size. We say that a solution $(\mathcal{E}^+, \mathcal{E}^-)$ to $\mathcal{GLP}$ is *ideal* if $\mathcal{E}^+ \neq E^+$ or $\mathcal{E}^- \neq E^-$.

Theorem 1. *$\mathcal{GLP}$ has a solution if and only if $\mathcal{CLP}$ has one.*

Proof. First, assume that $\mathcal{GLP}$ has a solution. Then (by definition), there exist non-empty subsets $\mathcal{E}^+ \subseteq E^+$, $\mathcal{E}^- \subseteq E^-$ such that $\mathcal{S}_{\mathcal{CLP}}(\mathcal{K}, T, \mathcal{E}^+, \mathcal{E}^-) \neq \emptyset$ and $\mathcal{S}_{\mathcal{CLP}}(\mathcal{K}, T, \mathcal{E}^+, \mathcal{E}^-) \subseteq \mathcal{S}_{\mathcal{CLP}}(\mathcal{K}, T, E^+, E^-)$ (properties *1.* and *2.*). Let C be an arbitrary element in $\mathcal{S}_{\mathcal{CLP}}(\mathcal{K}, T, \mathcal{E}^+, \mathcal{E}^-)$. Then, $C \in \mathcal{S}_{\mathcal{CLP}}(\mathcal{K}, T, E^+, E^-)$ and therefore C is a solution to $\mathcal{CLP}$.

It remains to prove that if $\mathcal{CLP}$ has a solution, then $\mathcal{GLP}$ also has one. For this, we provide a proof sketch and refer the reader to the supplemental material[2] for a complete proof. We define the size of $(\mathcal{E}^+, \mathcal{E}^-)$ as $|(\mathcal{E}^+, \mathcal{E}^-)| := |\mathcal{E}^+| + |\mathcal{E}^-|$. Let C be a solution to $\mathcal{CLP}$. Then, there is an example set pair $(\mathcal{E}^+, \mathcal{E}^-)$ that satisfies properties *1* and *2* (e.g., take $\mathcal{E}^+ = E^+$ and $\mathcal{E}^- = E^-$). Example set pair sizes are integers and are bounded by 2 from below since example sets are non-empty. Hence, there exist a size-minimal example set pair

[2] `https://github.com/dice-group/ROCES/blob/master/supplemental_material.pdf`

that satisfies properties *1* and *2*. Such an example set pair also satisfies property *3*, i.e., it is a solution to $\mathcal{GLCP}$. □

Theorem 1 provides sufficient and necessary conditions for our formulated learning problem ($\mathcal{GLCP}$) to have a solution. In particular, we have shown (see complete proof of Theorem 1 in the supplemental material) how ideal solutions to $\mathcal{GLCP}$ can be sought starting from an arbitrary solution to $\mathcal{CLCP}$. Given the sequential nature and complexity of this search process—see for example the use of existential quantifiers which often require search over the complete space—more efficient search strategies are necessary. In the following subsection, we propose an iterative stochastic search method coupled with gradient-based optimization to construct an approach that approximates solutions to $\mathcal{GLCP}$.

3.2 Learning Algorithm

Our approach uses the ConEx embedding model [Demir and Ngonga Ngomo, 2021] to obtain embeddings for input examples, and the Set Transformer [Lee *et al.*, 2019] to synthesize class expressions from embeddings. The embedding model and the Set Transformer instance are fused into a learner f_Θ and trained jointly. Algorithm 1 below describes how f_Θ can be used together with our iterative sampling technique to solve $\mathcal{GLCP}$.

Algorithm 1 Learning Step

Input: (C, E^+, E^-), f_Θ
Hyper-parameters: k, Opt (optimization algorithm)
Output: f_Θ

1: $S^+ \leftarrow [\min(k, |E^+|), 2k, 3k, \ldots, |E^+|]$
2: $S^- \leftarrow [\min(k, |E^-|), 2k, 3k, \ldots, |E^-|]$
 # Define probability functions over S^+ and S^-; $[s]_S$ denotes the position of s in S
3: $\forall\, x \in S^+, p^+(x) = \dfrac{1/[x]_{S^+}}{\sum_{s \in S^+}(1/[s]_{S^+})}$
4: $\forall\, x \in S^-, p^-(x) = \dfrac{1/[x]_{S^-}}{\sum_{s \in S^-}(1/[s]_{S^-})}$
 # Determine the number of examples to sample
5: Draw k^+ from S^+ following p^+
6: Draw k^- from S^- following p^-
7: $\mathcal{E}^+ \leftarrow \mathcal{U}(E^+, k^+)$ *# Uniform sampling*
8: $\mathcal{E}^- \leftarrow \mathcal{U}(E^-, k^-)$
9: $\hat{C} = f_\Theta(\mathcal{E}^+, \mathcal{E}^-)$ *# Synthesize a class expression*
10: $\mathcal{L} = \text{Loss}(\hat{C}, C)$ *# Compute the loss*
11: $\Theta \leftarrow Opt(\Theta, \nabla_\Theta \mathcal{L})$ *# Update the parameters Θ*
12: **return** f_Θ

Algorithm Description. In lines 1 and 2, we construct possible sizes S^+ and S^- for the subsets $\mathcal{E}^+$ and $\mathcal{E}^-$ (see Definition 3), respectively. Next, we define discrete probability functions[3] p^+ and p^- over S^+ and S^- (lines 3 and 4). In lines 5–8, we sample candidate example set sizes k^+ and k^- following p^+ and p^-, and construct candidate subsets of examples $\mathcal{E}^+$ and $\mathcal{E}^-$ by uniformly sampling k^+ positive and k^- negative examples, respectively. Finally, the parameterized learner f_Θ synthesizes an expression $\hat{C}$ which we compare to the target C, compute the loss and backpropagate to update the parameters Θ (lines 9–11). Note that Algorithm 1 describes a single learning step with one training data point for the sake of simplicity. In practice (e.g. in our experiments), a batch of training data points is given and parameter updates are performed based on the input batch. Once the learner f_Θ is trained using Algorithm 1, it can be employed to solve learning problems with arbitrary example set sizes; we provide more details in Section 4. In the rest of the paper, we refer to the parameterized learner f_Θ trained using Algorithm 1 as our approach ROCES.

Limitations. As mentioned earlier, our learning algorithm computes approximate solutions to $\mathcal{GLCP}$. The search for the exact solution can require up to $2^{|E^+|} \times 2^{|E^-|}$ checks (for $|E^+| = |E^-| = 100$, this is approximately 10^{60} checks). Hence, the optimal subsets of examples $(\mathcal{E}^+, \mathcal{E}^-)$ computed in Algorithm 1 might fail to satisfy property *3.* (minimality) in Definition 3 but they remain, in most cases, strictly smaller than the initial sets E^+ and E^- as suggested by the results in Table 3.

[3]The density functions p^+ and p^- are defined in such a way that higher probabilities are given to smaller values. In this way, we encourage the learner f_Θ to learn target expressions with small sets of examples. Of course, the search for the most suitable probability functions remains an interesting topic which we leave for future work.

4 Experiments

4.1 Datasets

We used four benchmark datasets in our experiments: Semantic Bible[4], Vicodi [Nagypál, 2005], Carcinogenesis [Westphal *et al.*, 2019], and Mutagenesis [Westphal *et al.*, 2019]. Vicodi describes the European history, and Semantic Bible, the New Testament. Carcinogenesis and Mutagenesis describe chemical compounds and how they relate to each other. On the last two datasets, CEL can provide insights into hidden properties shared by different compounds and facilitate further investigations by domain experts. For instance, a class expression learner in [Bühmann *et al.*, 2016] found that a chemical compound is carcinogenic if it can be described by the expression: $\neg(\exists$ `hasAtom.(Nitrogen-35` $\sqcup$ `Phosphorus-60` $\sqcup$ `Phosphorus-61` $\sqcup$ `Titanium-134`$)) \sqcap (\geq 3$ `hasStructure.(Halide` $\sqcap \neg$`Halide10`$) \sqcup \exists$ `amesTestPositive.`$\{$True$\} \sqcap \geq 5$ `hasBond.(`$\neg$`Bond-7`$))$. In natural language, this would translate into: <<A chemical compound is carcinogenic if and only if it does not contain a Nitrogen-35, Phosphorus-60, Phosphorus-61, or Titanium-134 atom and it has at least three Halide—excluding Halide10—structures or the ames test of the compound is positive and there are at least five atom bonds which are not of bond type 7>> (cf. [Bühmann *et al.*, 2016], Section 6.1). In our experiments, the test set consists of 100 learning problems on each benchmark dataset; we refer to [Kouagou *et al.*, 2023a] (Table 2, Section 5.1) for complete statistics.

[4]`https://www.semanticbible.com/ntn/ntn-overview.html`

4.2 Hyper-parameter Configuration

Because of its efficiency, we used the random search approach [Bergstra and Bengio, 2012] to find the best hyper-parameter values for our approach, namely the learning rate, the training batch size, the embedding dimension d for the embedding model ConEx, and the number of inducing points in the Set Transformer model. In the case of class expression learning, the authors of [Kouagou *et al.*, 2023a] argue that it is sufficient to search hyper-parameter values on one dataset, e.g. Carcinogenesis, and use the same values on the rest of the datasets; we adopt the same approach in this work. We report hyper-parameter settings in the supplemental material due to space constraints.

4.3 Hardware

We trained the learner f_Θ using Algorithm 1 (that is, ROCES) on a virtual machine equipped with 64 AMD EPYC 9334 32-Core Processors @3.91GHz, and a NVIDIA A100 80GB GPU. Post training, we used a server with 16 Intel Xeon E5-2695 CPUs @2.30GHz and 128GB RAM to conduct experiments on CEL where we compared our approach against the state-of-the-art CELOE, CLIP, EvoLearner, and NCES2. This is because CELOE and EvoLearner do not support GPU computations, and we needed to ensure a fair comparison regarding runtimes.

4.4 Results and Discussion

ROCES was trained for 400 epochs on each dataset. The evaluation metrics used during training are the "hard accuracy" and the "soft accuracy" as defined in [Kouagou *et al.*, 2023b]. We plot the training curves for the hard accuracy in Figure 1, and the rest in the supplemental material due to space constraints. From the figure, we can observe a rapid increase in accuracy in the early epochs and fast convergence on Carcinogenesis and Vicodi, which are the largest datasets. This suggests that on large datasets, ROCES learns better mappings between sets of examples and class expressions that describe them. At inference time, the quality of a solution is computed in terms of the number of positive/negative examples covered/ruled out (given by the F_1 score in Table 2).

To validate the effectiveness of ROCES, we conducted several experiments on the test sets. First, we compare ROCES against the state of the art on learning problems involving small sets of examples: we varied the number of examples between 4 and 264 using powers of 4 (cf. Figures 2, 3, and Table 2). In a second experiment, we compare all approaches on learning problems with full sets of examples (see columns named "Full" in Table 2). Finally, we measure how many times our approach ROCES outperforms EvoLearner without using all the provided examples (Table 3).

From Table 2, we can observe that ROCES significantly outperforms the state-of-the-art (up to +10% F_1 score) on the largest datasets Carcinogenesis and Vicodi when only 16 or 64 input examples are used. Moreover, ROCES appears to be competitive with the strongest baseline CLIP on Carcinogenesis when the complete sets of examples are used, and ranks second on Vicodi behind NCES2. On learning problems with limited input examples, NCES2 lags behind; it ranks last in 6 out of 8 cases when 16 or 64 examples are used. The same

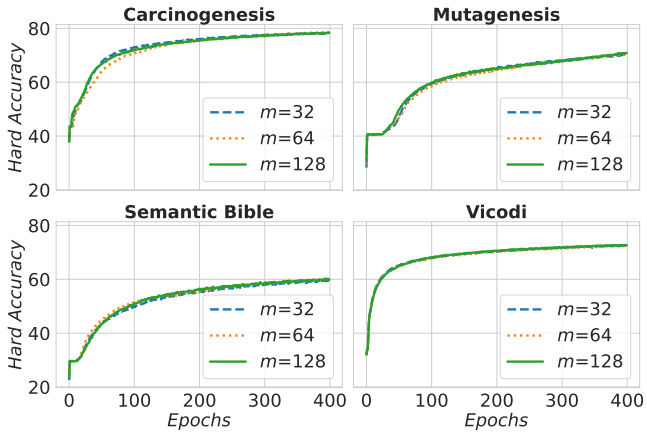

Figure 1: Hard accuracy curves of our proposed approach ROCES during training. m is the number of inducing points in the Set Transformer model. The probability functions p^+, p^- defined in Algorithm 1 are used.

can be observed on Figure 2 where we plot the quality of the computed solutions for various input example set sizes. This is because NCES2 applies the naive approach of training on static input example sets predefined by the data generator. Search-based approaches (CELOE, CLIP, and EvoLearner) show a more stable performance across different input example set sizes as they use classification metrics at each step of the search process. However, they fall short when it comes to runtimes especially on large datasets, see e.g. Figure 3 and the lower part of Table 2. On average, they are over 1000× slower than ROCES and NCES2. While the prediction times of search-based approaches remain volatile across different learning problem sizes, deep learning-based approaches such as ROCES maintain a nearly constant prediction time (see Figure 3). The prediction time of CLIP decreases slightly as the number of examples in learning problems increases. This is probably due to the fact that CLIP uses the refinement operator `ExpressRefinement` [Kouagou *et al.*, 2022] which is also used to generate the learning problems in test sets. Hence, CLIP is able to quickly retrieve solutions when more examples are available. ROCES remains the only approach to achieve both speed and high predictive performance on learning problems with various input example set sizes. Its superiority on large datasets can be attributed to the fact that its deep neural network-based synthesizer performs better when it has enough training data to learn from. Meanwhile, search-based methods struggle to navigate the vast search space induced by large datasets. They perform better on small datasets (e.g., Mutagenesis and Semantic Bible), where ROCES does not generalize well with the available training data.

In Table 3, we report the frequency at which ROCES outperforms EvoLearner—one of the best baselines—without using the complete sets of examples. More precisely, we let ROCES explore 50 random pairs $(\mathcal{E}^+, \mathcal{E}^-)$ of subsets of examples and compute a solution for each. The results in the table suggest that with just 50 trials, ROCES outperforms EvoLearner on at least 63% (the highest being 91%) of the learning problems. The average F_1

Dataset	Carcinogenesis			Mutagenesis			Semantic Bible			Vicodi		
k	16	64	Full	16	64	Full	16	64	Full	16	64	Full
						$\mathbf{F_1}$ (%)						
CELOE	37.33	32.59	29.24	78.50	78.15	74.46	76.21	85.01	88.60	27.16	26.68	22.63
CLIP	69.63	84.37	**96.57***	**88.55***	91.76	**95.65***	**80.75***	**88.24***	**92.24***	48.78	53.06	68.78
EvoLearner	71.52	82.93	89.34	88.27	**92.12***	95.37	77.66	82.47	88.38	60.49	57.39	76.99
NCES2	15.16	37.03	91.29	28.30	21.50	85.12	22.68	33.21	77.00	18.65	42.87	**91.06***
ROCES (ours)	**92.46***	93.45	93.73	62.54	79.11	90.36	67.39	77.15	75.12	**70.39***	**85.31***	84.51
ROCES$_U$ (ours)	80.42	**94.04***	94.54	50.75	79.23	91.75	68.86	75.90	76.43	64.94	81.47	87.81
						Runtime (sec.)						
CELOE	217.08	240.93	268.90	67.45	124.69	165.27	95.38	163.80	172.04	233.18	300.01	300.01
CLIP	201.32	200.29	33.00	197.93	205.42	107.19	188.58	186.06	188.22	300.01	249.64	151.14
EvoLearner	40.96	49.42	89.34	27.65	48.51	70.77	12.36	17.93	18.44	170.78	248.55	236.92
NCES2	0.01	0.01	0.09	0.01	0.01	0.09	0.01	0.01	0.05	0.01	0.01	0.10
ROCES (ours)	**0.01**	**0.01**	**0.07**	**0.01**	**0.01**	0.06	**0.01**	**0.01**	**0.03**	**0.01**	**0.01**	**0.10**
ROCES$_U$ (ours)	0.01	0.01	0.07	0.01	0.01	**0.05**	0.01	0.01	0.03	0.01	0.01	0.10

Table 2: Comparison of different approaches on learning problems with various input example set sizes (k). ROCES$_U$ is our approach ROCES where p^+ and p^- are uniform probability functions. We report the mean across 100 learning problems on each dataset. Bold (resp., underlined) values represent the best (resp., second best) across different approaches. The asterisk represents the Wilcoxon Signed-Rank significance test between ROCES and the best among the other approaches.

	Carcino.		Mutag.		Sem. B.		Vicodi	
	Freq.	**F_1**	**Freq.**	**F_1**	**Freq.**	**F_1**	**Freq.**	**F_1**
Min	80	90	42	72	44	63	73	78
Max	91	95	81	93	63	81	85	90
Avg.	82	94	56	88	47	75	79	85

Table 3: Frequency (Freq. in %) at which ROCES—with limited input examples—outperforms EvoLearner, and average quality (F_1 in %) of the solutions computed by ROCES. Min, Max, and Avg. are aggregations accross the 50 trials.

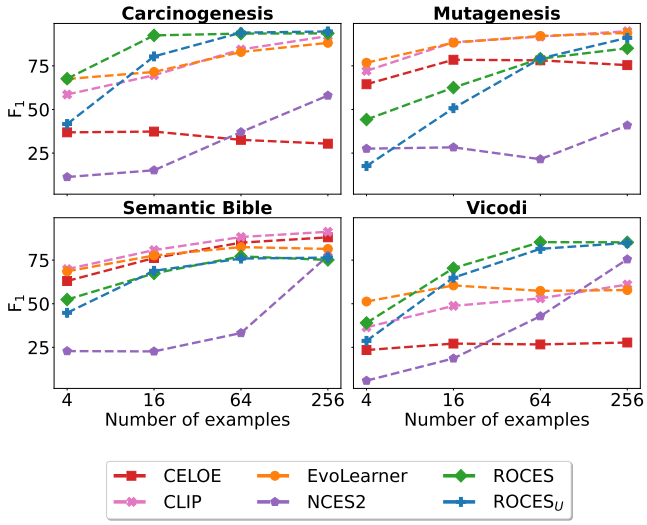

Figure 2: Average F_1 score of the computed solutions for different values of the input examples' set sizes.

score of the computed solutions is 90% and above on three datasets; the lowest performance being 81% and observed on Semantic Bible—see the row Max. Nonetheless, ROCES found the exact solution for 34% of the learning problems on this dataset. For example, it found the exact solution `Mountain` $\sqcup$ (`GeopoliticalArea` $\sqcap$ (`City` $\sqcup$ ($\exists$ `subregionOf`.$\top$))) where EvoLearner computed `City` $\sqcup$ ($\exists$ `location`.$\top$) $\sqcup$ (`Mountain` $\sqcap$ `GeographicArea`) $\sqcup$ ($\exists$ `subregionOf`.($\exists$ `subregionOf`.`GeographicArea`)) with 97.59% F_1 score. The solutions computed by other approaches are reported in the supplemental material.

5 Related Work

5.1 Ontology Learning

Ontology learning (OL) can be defined as the automatic or semi-automatic creation of ontologies using a pre-existing source of information such as natural language text [Wong *et al.*, 2012]. This creation process involves the extraction of domain-specific terms or concepts and the links between them. Links between concepts are represented using hierarchical data structures, e.g., class inclusions (CIs) in description logics. As an example, the statement "penguins are birds that can swim or animals that have wings and cannot fly" would be translated into `Penguin` $\sqsubseteq$ (`Bird` $\sqcap$ `CanSwim`) $\sqcup$ (`Animal` $\sqcap$ `HasWings` $\sqcap$ $\neg$`CanFly`). As a result, CI learning is a fundamental component in OL. CI learning approaches are tasked to build a hierarchy between concepts in the provided source of information. Neural network-based approaches [Petrucci *et al.*, 2016; Ma and Distel, 2013] leverage NLP techniques [Cambria and White, 2014] to translate text into description logic class hierarchy axioms. Inductive logic programming (ILP) approaches [Muggleton, 1991] learn CIs from positive and neg-

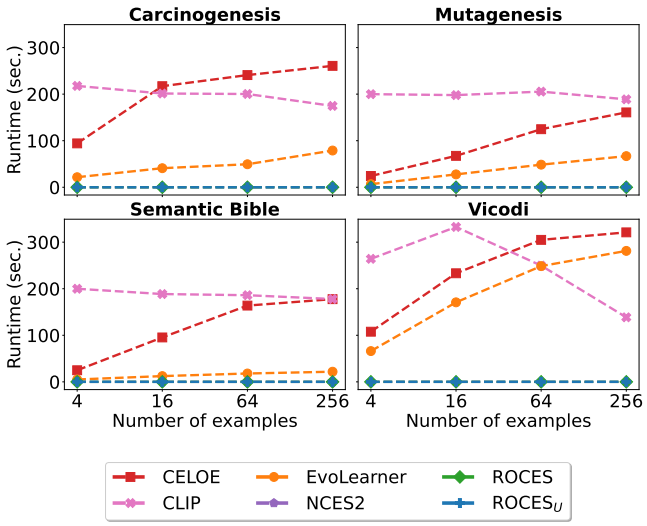

Figure 3: Average runtime per approach for different values of the input examples' set sizes.

ative examples. These examples are provided by an oracle, e.g., a domain expert or even a neural network [Weiss *et al.*, 2017]. Association rule mining (ARM) approaches [Agrawal *et al.*, 1993; Galárraga *et al.*, 2015; Meilicke *et al.*, 2019] rely on support and confidence score functions. In this work, we do not aim to learn CIs but rather class expressions from given positive and negative examples. CEL can be regarded as a step beyond OL as it can be applied to cure or enrich the created ontology [Lehmann *et al.*, 2011].

5.2 Class Expression Learning Approaches

Several approaches have been developed to tackle CEL problems. These approaches can be classified into three main groups: 1.) (pure) search-based approaches [Heindorf *et al.*, 2022; Lehmann *et al.*, 2011; Sarker and Hitzler, 2019], 2.) neural symbolic-based approaches [Demir and Ngonga Ngomo, 2023; Kouagou *et al.*, 2022], and 3.) neural network-based (a.k.a synthesis-based) approaches [Kouagou *et al.*, 2023b; Kouagou *et al.*, 2023a]. Our approach ROCES belongs to the third category. EvoLearner [Heindorf *et al.*, 2022] employs evolutionary algorithms, and initializes its population (class expressions) by random walks on the input knowledge base. It then subsequently applies mutations, cross-over operations, and a heuristic function to construct the next generations of the initial population until a solution is found. CELOE [Lehmann *et al.*, 2011] is a CEL algorithm developed specifically for ontology engineering. It is implemented in DL-Learner [Lehmann, 2009] and regarded as the best algorithm in the framework. ECII [Sarker and Hitzler, 2019] is a search-based algorithm that constructs its search space by using disjunctions, conjunctions and negations of pre-selected complex class expressions. It does not use a refinement operator and invokes a reasoner only once for each learning problem. CLIP [Kouagou *et al.*, 2022] steers the search space by using concept length predictors which are initially trained in an unsupervised manner. DRILL learns to optimally traverse the search space via deep Q-learning. NCES [Kouagou *et al.*, 2023b] and NCES2 [Kouagou *et al.*, 2023a] use deep neural networks to learn mappings between sets of examples and class expressions without a search process. NCES2 runs in the description logic $\mathcal{ALCHIQ}(\mathcal{D})$ and is regarded as the most mature synthesis-based approach for CEL to date. Synthesis-based approaches are known to be highly scalable and well suited for web-scale applications of CEL.

Despite their effectiveness, current synthesis-based approaches are not robust to variations in learning problem sizes. Their performance decreases drastically on learning problems whose example sets are larger or smaller than those encountered during training. Our approach ROCES addresses this limitation by repeatedly constructing multiple, different-size subsets of examples during training via a probability distribution on possible example set sizes, see Algorithm 1. We show empirically that ROCES is robust to changes in the number of input examples while remaining highly scalable.

6 Conclusion and Outlook

We proposed a generalization of the class expression learning problem dubbed $\mathcal{GLP}$ and established the connections between the solutions of the classical and generalized problem. In particular, we showed how solutions to $\mathcal{GLP}$ can be sought starting from an arbitrary solution to the classical problem. We also proposed ROCES, a neural network-based approach to solve $\mathcal{GLP}$, which uses predefined probability functions to construct subsets of examples with various sizes during training. ROCES consistently outperformed existing synthesis-based approaches on learning problems with limited input examples while remaining highly scalable and competitive on full-size learning problems.

The probability functions p^+ and p^- (skewed towards smaller values) in Algorithm 1 are better than uniform probability functions when only a few examples are used (cf. ROCES vs. ROCES$_U$ in Table 2). However, as the number of examples increases, uniform probability functions catch up rapidly. The search for the best probability functions in this context is an interesting topic which we plan to investigate further.

Acknowledgements

This work has received funding from the European Union's Horizon 2020 research and innovation programme within the project KnowGraphs under the Marie Skłodowska-Curie grant No 860801, the European Union's Horizon Europe research and innovation programme within the project ENEXA under the grant No 101070305, and the European Union's Horizon Europe research and innovation programme within the project LEMUR under the Marie Skłodowska-Curie grant agreement No 101073307. This work has also been supported by the Ministry of Culture and Science of North Rhine-Westphalia (MKW NRW) within the project SAIL under the grant No NW21-059D and by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation): TRR 318/1 2021 – 438445824.

References

[Agrawal *et al.*, 1993] Rakesh Agrawal, Tomasz Imielinski, and Arun N. Swami. Mining association rules between sets of items in large databases. In *SIGMOD Conference*, pages 207–216. ACM Press, 1993.

[Baci and Heindorf, 2023] Alkid Baci and Stefan Heindorf. Accelerating concept learning via sampling. In *CIKM*, pages 3733–3737. ACM, 2023.

[Badea and Nienhuys-Cheng, 2000] Liviu Badea and Shan-Hwei Nienhuys-Cheng. A refinement operator for description logics. In *ILP*, volume 1866 of *Lecture Notes in Computer Science*, pages 40–59. Springer, 2000.

[Barwise, 1977] Jon Barwise. An introduction to first-order logic. In *Studies in Logic and the Foundations of Mathematics*, volume 90, pages 5–46. Elsevier, 1977.

[Bergstra and Bengio, 2012] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *J. Mach. Learn. Res.*, 13:281–305, 2012.

[Boeker *et al.*, 2016] Martin Boeker, Fábio França, Peter Bronsert, and Stefan Schulz. Tnm-o: ontology support for staging of malignant tumours. *Journal of biomedical semantics*, 7:1–11, 2016.

[Bordes *et al.*, 2013] Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *NIPS*, pages 2787–2795, 2013.

[Bordes *et al.*, 2014] Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. A semantic matching energy function for learning with multi-relational data - application to word-sense disambiguation. *Mach. Learn.*, 94(2):233–259, 2014.

[Bühmann *et al.*, 2016] Lorenz Bühmann, Jens Lehmann, and Patrick Westphal. Dl-learner—a framework for inductive learning on the semantic web. *Journal of Web Semantics*, 39:15–24, 2016.

[Büning and Lettmann, 1999] Hans Kleine Büning and Theodor Lettmann. *Propositional logic: deduction and algorithms*, volume 48. Cambridge University Press, 1999.

[Cambria and White, 2014] Erik Cambria and Bebo White. Jumping NLP curves: A review of natural language processing research [review article]. *IEEE Comput. Intell. Mag.*, 9(2):48–57, 2014.

[Chen and Zaniolo, 2017] Muhao Chen and Carlo Zaniolo. Learning multi-faceted knowledge graph embeddings for natural language processing. In *IJCAI*, pages 5169–5170, 2017.

[Demir and Ngonga Ngomo, 2021] Caglar Demir and Axel-Cyrille Ngonga Ngomo. Convolutional complex knowledge graph embeddings. In *ESWC*, volume 12731 of *Lecture Notes in Computer Science*, pages 409–424. Springer, 2021.

[Demir and Ngonga Ngomo, 2023] Caglar Demir and Axel-Cyrille Ngonga Ngomo. Neuro-symbolic class expression learning. In *IJCAI*, pages 3624–3632, 2023.

[Demir *et al.*, 2022] Caglar Demir, Anna Himmelhuber, Yushan Liu, Alexander Bigerl, Diego Moussallem, and Axel-Cyrille Ngonga Ngomo. Rapid explainability for skill description learning. In Anastasia Dimou, Armin Haller, Anna Lisa Gentile, and Petar Ristoski, editors, *Proceedings of the ISWC 2022 Posters, Demos and Industry Tracks: From Novel Ideas to Industrial Practice co-located with 21st International Semantic Web Conference (ISWC 2022), Virtual Conference, Hangzhou, China, October 23-27, 2022*, volume 3254 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2022.

[Galárraga *et al.*, 2015] Luis Galárraga, Christina Teflioudi, Katja Hose, and Fabian M Suchanek. Fast rule mining in ontological knowledge bases with amie+. *The VLDB Journal*, 24(6):707–730, 2015.

[Hartel *et al.*, 2005] Frank W Hartel, Sherri de Coronado, Robert Dionne, Gilberto Fragoso, and Jennifer Golbeck. Modeling a description logic vocabulary for cancer research. *Journal of biomedical informatics*, 38(2):114–129, 2005.

[Heindorf *et al.*, 2022] Stefan Heindorf, Lukas Blübaum, Nick Düsterhus, Till Werner, Varun Nandkumar Golani, Caglar Demir, and Axel-Cyrille Ngonga Ngomo. Evolearner: Learning description logics with evolutionary algorithms. In *WWW*, pages 818–828. ACM, 2022.

[Hogan *et al.*, 2022] Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard de Melo, Claudio Gutierrez, Sabrina Kirrane, José Emilio Labra Gayo, Roberto Navigli, Sebastian Neumaier, Axel-Cyrille Ngonga Ngomo, Axel Polleres, Sabbir M. Rashid, Anisa Rula, Lukas Schmelzeisen, Juan F. Sequeda, Steffen Staab, and Antoine Zimmermann. Knowledge graphs. *ACM Comput. Surv.*, 54(4):71:1–71:37, 2022.

[Kouagou *et al.*, 2022] N'Dah Jean Kouagou, Stefan Heindorf, Caglar Demir, and Axel-Cyrille Ngonga Ngomo. Learning concept lengths accelerates concept learning in ALC. In *ESWC*, volume 13261 of *Lecture Notes in Computer Science*, pages 236–252. Springer, 2022.

[Kouagou *et al.*, 2023a] N'Dah Jean Kouagou, Stefan Heindorf, Caglar Demir, and Axel-Cyrille Ngonga Ngomo. Neural class expression synthesis in *ALCHIQ(D)*. In *ECML/PKDD (4)*, volume 14172 of *Lecture Notes in Computer Science*, pages 196–212. Springer, 2023.

[Kouagou *et al.*, 2023b] N'Dah Jean Kouagou, Stefan Heindorf, Caglar Demir, and Axel-Cyrille Ngonga Ngomo. Neural class expression synthesis. In *European Semantic Web Conference*, pages 209–226. Springer, 2023.

[Kouagou *et al.*, 2024] N'Dah Jean Kouagou, Caglar Demir, Hamada M. Zahera, Adrian Wilke, Stefan Heindorf, Jiayi Li, and Axel-Cyrille Ngonga Ngomo. Universal knowledge graph embeddings. In *Companion Proceedings of the ACM on Web Conference 2024*, WWW '24, page 1793–1797, New York, NY, USA, 2024. Association for Computing Machinery.

[Lee *et al.*, 2019] Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh.

Set transformer: A framework for attention-based permutation-invariant neural networks. In *International conference on machine learning*, pages 3744–3753. PMLR, 2019.

[Lehmann and Hitzler, 2007] Jens Lehmann and Pascal Hitzler. Foundations of refinement operators for description logics. In *ILP*, volume 4894 of *Lecture Notes in Computer Science*, pages 161–174. Springer, 2007.

[Lehmann and Hitzler, 2010] Jens Lehmann and Pascal Hitzler. Concept learning in description logics using refinement operators. *Machine Learning*, 78, 2010.

[Lehmann *et al.*, 2011] Jens Lehmann, Sören Auer, Lorenz Bühmann, and Sebastian Tramp. Class expression learning for ontology engineering. *Journal of Web Semantics*, 2011.

[Lehmann, 2009] Jens Lehmann. Dl-learner: learning concepts in description logics. *The Journal of Machine Learning Research*, 2009.

[Lehmann, 2010] Jens Lehmann. *Learning OWL class expressions*, volume 22. IOS Press, 2010.

[Lin *et al.*, 2015] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. Learning entity and relation embeddings for knowledge graph completion. In *AAAI*, pages 2181–2187. AAAI Press, 2015.

[Ma and Distel, 2013] Yue Ma and Felix Distel. Learning formal definitions for snomed CT from text. In *AIME*, volume 7885 of *Lecture Notes in Computer Science*, pages 73–77. Springer, 2013.

[Meilicke *et al.*, 2019] Christian Meilicke, Melisachew Wudage Chekol, Daniel Ruffinelli, and Heiner Stuckenschmidt. Anytime bottom-up rule learning for knowledge graph completion. In *IJCAI*, pages 3137–3143, 2019.

[Mnih *et al.*, 2013] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. In *NIPS*, 2013.

[Muggleton, 1991] Stephen Muggleton. Inductive logic programming. *New generation computing*, 1991.

[Nagypál, 2005] Gábor Nagypál. History ontology building: The technical view. *Humanities, Computers and Cultural Heritage*, page 207, 2005.

[Nardi *et al.*, 2003] Daniele Nardi, Ronald Jay Brachman, et al. An introduction to description logics. *Description logic handbook*, 1, 2003.

[Nickel *et al.*, 2012] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. Factorizing YAGO: scalable machine learning for linked data. In *WWW*, pages 271–280. ACM, 2012.

[Petrucci *et al.*, 2016] Giulio Petrucci, Chiara Ghidini, and Marco Rospocher. Ontology learning in the deep. In *EKAW*, volume 10024 of *Lecture Notes in Computer Science*, pages 480–495, 2016.

[Rizzo *et al.*, 2020] Giuseppe Rizzo, Nicola Fanizzi, and Claudia d'Amato. Class expression induction as concept space exploration: From dl-foil to dl-focl. *Future Generation Computer Systems*, 2020.

[Sarker and Hitzler, 2019] Md. Kamruzzaman Sarker and Pascal Hitzler. Efficient concept induction for description logics. In *AAAI*, pages 3036–3043. AAAI Press, 2019.

[Schulz *et al.*, 2008] Stefan Schulz, Kornél Markó, and Boontawee Suntisrivaraporn. Formal representation of complex snomed ct expressions. In *BMC medical informatics and decision making*, volume 8, pages 1–6. Springer, 2008.

[Wang and Li, 2016] Zhigang Wang and Juanzi Li. Text-enhanced representation learning for knowledge graph. In *Proc. of IJCAI*, 2016.

[Weiss *et al.*, 2017] Gail Weiss, Yoav Goldberg, and Eran Yahav. Extracting automata from recurrent neural networks using queries and counterexamples. *ArXiv*, abs/1711.09576, 2017.

[Westphal *et al.*, 2019] Patrick Westphal, Lorenz Bühmann, Simon Bin, Hajira Jabeen, and Jens Lehmann. Sml-bench–a benchmarking framework for structured machine learning. *Semantic Web*, 10(2):231–245, 2019.

[Wong *et al.*, 2012] Wilson Wong, Wei Liu, and Mohammed Bennamoun. Ontology learning from text: A look back and into the future. *ACM computing surveys (CSUR)*, 44(4):1–36, 2012.

[Xie *et al.*, 2016] Ruobing Xie, Zhiyuan Liu, Jia Jia, Huanbo Luan, and Maosong Sun. Representation learning of knowledge graphs with entity descriptions. In *AAAI*, pages 2659–2665. AAAI Press, 2016.

[Zhang *et al.*, 2016] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. Collaborative knowledge base embedding for recommender systems. In *KDD*, pages 353–362. ACM, 2016.