Class Expression Learning with Permutation-Invariant Embeddings

Abstract

Class expression learning deals with learning description logic concepts from an RDF knowledge base and input examples. The goal is to learn a concept that covers all positive examples, while not covering any negative examples. Although state-of-the-art models have been successfully applied to tackle this problem, their large-scale applications have been severely hindered due to their impractical runtimes. Arguably, this limitation stems from their needs for exploring numerous concepts. Here, we investigate a remedy for this limitation. We formulate the class expression learning problem in a fashion akin to multi-label classification problem and propose a permutation-invariant neural embedding model (NERO) to reduce the rate of exploration. NERO accurately predicts quality scores for pre-selected concepts for given input example sets. Through ranking concepts in descending order of predicted quality scores, the standard search procedures of state-of-the-art models can start in multiple advantageous regions of the quasi-ordered search space, than starting in the most general concept $\top$. Hence, NERO can be used to initialize a search tree of a state-of-the-art model to accelerate its search process. Our experiments on 5 benchmark datasets with 770 learning problems suggest that using NERO led to significant improvements (p-value $< 1\%$) in the number of explored expressions and the total runtime time.

1 Introduction

It is well established that a lack of transparency and explainability in AI-driven systems reduces the trust in and the verifiability of their decisions [14, 26, 27]. Class Expression Learning (CEL) [21] is a form of explainable AI of increasing importance on the Web, as a learned description logic concepts are interpretable, e.g., through available verbalisation techniques [22, 30]. Improving upon CEL thus has the potential of easing the use of explainable AI in real-life applications at Web scale along with contributing to the corresponding societal advantages tied to explainability [5]. Although state-of-the-art based models have been successfully applied to tackle the CEL problem, their large-scale applications have been severely hindered due to their impractical runtimes.

In this work, we propose to forge state-of-the-art CEL models with a neural embedding model to reduce their impractical runtimes. The formal setting for CEL is as follows: Given an RDF Knowledge Base (KB) $\mathcal{K}$ over a Description Logic (DL), a set of positive examples E^+, and a set of negative examples E^-, a CEL algorithm aims to learn a DL concept $\mathtt{H}$ such that $\forall p \in E^+ \; \mathcal{K} \models \mathtt{H}(p)$ and $\forall n \in E^- \; \mathcal{K} \not\models \mathtt{H}(n)$ [7, 9, 15, 21]. Most CEL algorithms attempt to find a $\mathtt{H}$ by reformulating the CEL problem as a search problem in an infinite quasi-ordered state space $(\mathcal{S}, \preceq)$ [4, 9, 20, 31]. The search for a $\mathtt{H}$ is steered by optimizing a pre-defined heuristic function [19, 25]. A CEL algorithm explores $\mathcal{S}$ to find a $\mathtt{H}$ through iteratively refining states assigned with top heuristic values at a time (see Section 3 for more details). Expectedly, as the size of $\mathcal{K}$ increases, exploring $\mathcal{S}$ becomes a computational bottleneck [28]. To accelerate CEL, we formulate it as a multi-label classification problem and propose NERO (a neural permutation-invariant neural model) that maps two input sets of examples to F_1 quality scores of pre-selected DL concepts. For instance, a given learning problem

Preprint. Under review.

$(\mathcal{K}, E^+, E^-)$ as illustrated in Figure 1, NERO accurately predicts F_1 quality scores over `Person`, `Place`, `Organisation` and Person $\sqcap \exists$hasSibling.$\top$. By this, we can select top ranked concepts and add them in the search tree of state-of-the-art CEL model. State-of-the-art CEL model is therefore allowed to start its search in more advantages stage than $\top$. Our experiments on in 770 CEL problems suggest that using NERO with a state-of-the-art CEL consistently leads better results, i.e., finding accurate expressions, while **significantly** reducing the computational time as well as the number of explored concepts. The results of Wilcoxon signed rank tests confirm that the superior performance of NERO is statistical significant (p-value $< 1\%$).

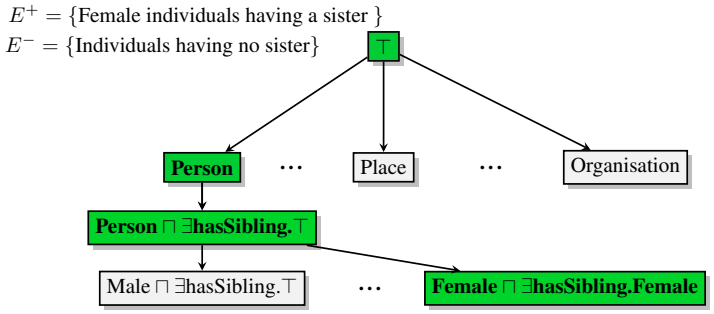

Figure 1: Illustration of traversing a quasi-ordered search space $\mathcal{S}$. Green filled boxes represent a sequence of $\mathcal{ALC}$ expressions terminating in a goal expression.

2 Related Work

A plethora of works have investigated learning in Description Logics [4, 8, 9, 25, 28]. We refer to [2, 1, 12, 16] for an introduction. One of the major differences between most CEL approaches is the design of heuristic functions and the refinement operators [3, 10, 11, 15, 21]. The problem of finding a DL concept `H` satisfying Equation (1) w.r.t. input examples E^+ and E^-. is considered as a search problem in an infinite quasi-ordered state space $(\mathcal{S}, \preceq)$ [4, 9, 20, 31]. The search for a `H` is steered by optimizing a pre-defined heuristic function [19, 25, 3, 19, 21]. Figure 1 visualizes this search process.

The DL-Learner framework including several state-of-the-art CEL algorithms (e.g. OCEL, ELTL, and CELOE) is one of the most commonly used framework to tackle the CEL problem [3, 18, 19, 20]. OCEL computes heuristic values based on the horizontal expansion penalty that introduces a dynamic upper-bound on the length of possible refinements. ELTL is based on a refinement operator for the DL $\mathcal{EL}$ and replaces the horizontal expansion penalty with a penalty for the length of an expression. CELOE builds on OCEL and applies a more sophisticated heuristic function that favors syntactically shorter expressions (see Section 3 for more details). CELOE is currently the best class expression learning algorithm available within DL-Learner and often outperforms many state-of-the-art models including OCEL and ELTL in terms of the quality of learned expression as well as number of explored expressions [17]. This finding is corroborated by our experiments, wherein we also observe that CELOE often found more accurate expressions than ELTL, while exploring less. The aforementioned approaches apply redundancy elimination and expression simplification rules to reduce the number of explored expressions. Although applying redundancy elimination and expression simplification rules often reduce the number of explored expressions, long runtimes still prohibit large scale applications of state-of-the-art models [6, 13].

In this work, we evaluate NERO against OCEL, ELTL, and CELOE models provided in DL-Learner framework for two reasons: (1) the DL-Learner framework is regarded as the most mature and recent system for CEL [28] and (2) most recently developed models are often evaluated w.r.t. the quality of expressions as well as runtimes. Yet, not counting the number of explored expressions does not permit us to quantify whether possible improvements through NERO may stem from our novel idea or our implementation. Consequently, in our experiments, we mainly compare NERO against CELOE in terms of number of explored expressions, quality of learned expressions as well as runtimes.

3 Background & Notation

Description Logics. A DL is a decidable fragment of first-order predicate logic that uses only unary and binary predicates [2]. The set of unary predicates, binary predicates and constants correspond to the set of named concepts N_C, roles N_R, and individuals N_I of the DLs respectively.

Knowledge Bases. A KB $\mathcal{K}$ over DL (say $\mathcal{ALC}$) is a pair $\mathcal{K} = (Tbox, Abox)$. Let $\mathtt{A}, \mathtt{B} \in N_C$ be two named concepts, $r \in N_R$ be a role, and $a, b \in N_I$ be individuals providing in $\mathcal{K}$. $Tbox$ contains terminological axioms between concepts in the form subsumption $\mathtt{A} \sqsubseteq \mathtt{B}$ or equivalence $\mathtt{A} \equiv \mathtt{B}$. $Abox$ contains assertions describing relationships between concepts and individuals in the form membership $\mathtt{A}(a)$ or between individuals in the form $r(a, b)$. Most KBs on the Web provide a large collections of facts in the form of assertions [24]. Yet, they often lack well-structured ontologies [23, 29].

Class Expression Learning and Heuristics. Let $\mathcal{K}$ over $\mathcal{ALC}$, the set E^+ of positive examples, and the set E^- of negative examples be given, where $E^+, E^- \subset N_I$ and $E^+ \cap E^- = \emptyset$. The goal of CEL is to find an $\mathcal{ALC}$ concept $\mathtt{H}$ s.t.

$$\forall p \in E^+, \forall n \in E^- \big(\mathcal{K} \models \mathtt{H}(p)) \wedge (\mathcal{K} \not\models \mathtt{H}(n)\big). \tag{1}$$

The CEL problem is transformed into a search problem within a quasi-ordered state space $(\mathcal{S}, \preceq)$ [4, 9, 11, 21, 20, 31], where each state is an $\mathcal{ALC}$ concept. Traversing in $\mathcal{S}$ is commonly conducted via top-down (also called downward) refinement operators, which are defined as $\rho : \mathcal{S} \rightarrow 2^{\mathcal{S}}$ with

$$\forall \mathtt{A} \in \mathcal{S} : \rho(\mathtt{A}) \subseteq \{\mathtt{B} \in \mathcal{S} \mid \mathtt{B} \preceq \mathtt{A}\}. \tag{2}$$

State-of-the-art CEL models begin their search towards a $\mathtt{H}$ after a search tree is initialized with the most general expression ($\top$) as a root node. This search tree is iteratively built by selecting a node containing a quasi-ordered expression with the highest heuristic value and adding its qualifying refinements as its children into a search tree [20]. The key to an efficient search in $\mathcal{S}$ is a heuristic function steering the search towards a $\mathtt{H}$ [21]. To introduce an example of a state-of-the-art heuristic function as well as a quality function, we must first define a *retrieval function*. Let $\mathcal{C}$, N_I be all valid $\mathcal{ALC}$ class expressions and the set of individuals occurring in $\mathcal{K}$, respectively. We define the retrieval function $\mathcal{R}_{\mathcal{K}} : \mathcal{C} \rightarrow 2^{N_I}$. $\mathcal{R}$ maps a class expression $\mathtt{A}$ to the set of its individuals under Open World Assumption (OWA) or Close World Assumption (CWA). Given $\mathtt{A} \in \mathcal{S}$ and one of its downward refinements $\mathtt{B} \in \rho(\mathtt{A})$, CELOE computes heuristic values as

$$\phi_{\text{CELOE}}(\mathtt{A}, \mathtt{B}) = \mathrm{Q}(\mathtt{B}) + \lambda \cdot \big[\mathrm{Q}(\mathtt{B}) - \mathrm{Q}(\mathtt{A})\big] - \beta \cdot |\mathtt{B}|, \tag{3}$$

where $\beta > \lambda \geq 0$ and $\mathrm{Q}(\cdot)$ denotes a quality function (e.g. F_1 score, accuracy). Through $\mathrm{Q}(\cdot)$ and $|\cdot|$, the search is steered towards more accurate and syntactically shorter concepts. F_1 score of an concept $\mathtt{A}$ is computed as

$$F_1(\mathtt{A}) = \frac{|E^+ \cap \mathcal{R}_k b(\mathtt{A})|}{|E^+ \cap \mathcal{R}_k b(\mathtt{A})| + 0.5(|\ E^- \cap \mathcal{R}_k b(\mathtt{A})| + |\ E^+ \setminus \mathcal{R}_k b(\mathtt{A})|)}. \tag{4}$$

Note that maximizing Equation (4) is more restrictive than satisfying Equation (1).

4 Methodology

Motivation. The goal in the CEL problem is to find a DL concept $\mathtt{A} \in \mathcal{C}$ satisfying Equation (1) and ideally maximizing Equation (4). Here, our goal is to find a $\mathtt{A}$ without excessive exploration.

NERO. Equation (4) indicates that $F_1(\cdot)$ is invariant to the order of individuals in E^+, E^-, and $\mathcal{R}(\cdot)$. Previously, Zaheer et al. [32] have proven that all functions being invariant to the order in inputs can be decomposed into

$$f(\mathbf{x}) = \phi\Big(\sum_{x \in \mathbf{x}} \psi(x)\Big), \tag{5}$$

where $\mathbf{x} = \{x_1, \ldots, x_m\} \in 2^{\mathcal{X}}$ and $\phi(\cdot)$ and $\psi(\cdot)$ denote a set of input and two parametrized continuous functions, respectively. This implies that a neural network defined in Equation (5) can be effectively used to predict F_1 score of an $\mathcal{ALC}$ concept w.r.t. E^+ and E^-. Through accurately

predicting quality of $\mathcal{ALC}$ concepts without using F_1 and $\mathcal{R}_k b$, we can accelerate the process of finding a H, i.e., alleviate the need of excessive exploration. Therefore, we define NERO as follows

$$\text{NERO}(E^+, E^-) = \sigma\Big(\phi\Big(\sum_{x \in E^+} \psi(x)\Big) - \phi\Big(\sum_{x \in E^-} \psi(x)\Big)\Big), \tag{6}$$

where $\psi(\cdot) : N_I \to \mathbb{R}^m$ and $\phi : \mathbb{R}^m \to [0,1]^{|\mathcal{T}|}$ denote the encoder and the decoder functions, respectively. $\mathcal{T}$ represents the pre-selected target concepts. The result of the translation operation denoted with $\mathbf{z} \in \mathbb{R}^m$ is normalized via the logistic sigmoid function $\sigma(\mathbf{z}) = \frac{1}{1+\exp(-\mathbf{z})}$.

Construction of Target Labels & Training Procedure. Target concepts are obtained as $\mathcal{T} := \{C \mid C \in \rho(\top) \wedge |C| \leq maxlength \wedge 0 < |\mathcal{R}(C)|\}$, where $maxlength \in \mathbb{N}$ and $|\mathcal{T}| = d$. $maxlength$ and d can be used to define the size of $\mathcal{T}$. Next, we define the training procedure used in our experiments. Let $\mathcal{D} = \{(E_i^+, E_i^-, \mathbf{y}_i)\}_{i=1}^N$ represent a training dataset, where a training data point $(E^+, E^-, \mathbf{y})$ is obtained in four consecutive steps: (i) Sample C from $\mathcal{T}$ uniformly at random, (ii) Sample k individuals $E^- \subset \mathcal{R}(C)$ uniformly at random, (iii) Sample k individuals $E^- \subset N_I \setminus E^+$ uniformly at random, and (iv) Compute F_1 scores $\mathbf{y}$ via Equation (4) w.r.t. E^+, E^-, for all $C \in \mathcal{T}$. For a given data point $E^+, E^-, \mathbf{y}$ and predictions $\hat{\mathbf{y}} := \sigma\big(\text{NERO}(E^+, E^-)\big)$, an incurred binary cross entropy loss is computed as

$$\ell(\mathbf{y}, \hat{\mathbf{y}}) = -\frac{1}{|\mathcal{T}|} \sum_{i=1}^{|\mathcal{T}|} \mathbf{y}^{(i)} \log(\hat{\mathbf{y}}^{(i)}) + (1 - \mathbf{y}^{(i)}) \log(1 - \hat{\mathbf{y}}^{(i)}). \tag{7}$$

During training, NERO learns permutation-invariant representations for E^+ and E^- tailored towards predicting F_1 score of $\mathcal{T}$.

5 Experiments

We followed the commonly used experimental setup providing in DL-Learner [21, 3]. Hence, we used five benchmark datasets and the respective learning problems provided therein. To perform a more extensive comparison between models, we also generated additional random learning problems. In our experiments, we evaluate all models in $\mathcal{ALC}$ for CEL on the same hardware. To ensure the reproducibility of our experiments, we provide all necessary details pertaining to the construction of $\mathcal{T}$, the training procedure, the hardware setup, hyperparameter selection in the supplemental material.[1]

We evaluated models via the F_1 score, the runtime and number of tested concepts. The F_1 score is used to measure the quality of the concepts found w.r.t. positive and negative examples, while the runtime and the number of tested concepts are to measure the efficiency. We measured the full computation time including the time spent prepossessing time of the input data and tackling the learning problem. Moreover, we used two standard stopping criteria for state-of-the-art models. (i) We set the maximum runtime to 10 seconds although models often reach good solutions within 1.5 seconds [20]. (ii) The models are configured to terminate as soon as they found a goal state (i.e., a state with F_1 score = 1.0).

6 Results

Results on benchmark learning problems. Table 1 suggests that NERO outperforms CELOE and ELTL in 8 out of 9 and in 5 out of 6 metrics, respectively. On the Family benchmark dataset, NERO explores on average $\mathbf{67.6}\times$ fewer concepts to find more accurate concepts. This rate of efficiency in terms of exploration results in the fact that NERO consistently requires less computational time. CELOE and ELTL require at least $\mathbf{14.7}\times$ more time than NERO. Similarly, results on Mutagenesis and Carcinogenesis benchmark dataset indicate that NERO requires **at least 38% less time** than CELOE and ELTL, while requiring to explore at least $\mathbf{2.3}\times$ fewer concepts. Note that we did not use any parallel computation in NERO, i.e., the input knowledge base and parameters of NERO are

[1]The URL of the supplemental material:`https://drive.google.com/drive/folders/1_Mdn6cuVNFIiaveU3Y_spJszixcoRteX?usp=sharing`

reloaded for each learning problem. Table 1 confirms that our hypothesis (see Section 4): through learning permutation-invariant representation, NERO can effectively predicts quality scores of $\mathcal{ALC}$ concepts.

Table 1: Results on benchmark learning problems provided within DL-Learner. F_1, T, and Exp. denote F_1 score, total runtime in seconds, and the number of explored expressions, respectively. ELTL does not report the Exp. Predictions of NERO are obtained by exploring only at most the top-100 ranked expressions. Bold entries denote best results.

Dataset	**NERO**			**CELOE**			**ELTL**	
	F_1	T	Exp.	F_1	T	Exp.	F_1	T
Family	$\mathbf{0.984}$	$\mathbf{0.68}$	$\mathbf{21}$	0.980	4.65	1429	0.964	4.12
Mutagenesis	$\mathbf{0.704}$	$\mathbf{13.18}$	$\mathbf{100}$	$\mathbf{0.704}$	23.05	516	$\mathbf{0.704}$	21.04
Carcinogenesis	$\mathbf{0.720}$	$\mathbf{26.26}$	$\mathbf{100}$	0.714	37.18	230	0.719	36.29

Results on random learning problems & Hypothesis Testing. The full respective results on random learning problems are reported in the supplemental material. Overall, these results suggest that NERO consistently outperforms CELOE and ELTL in all metrics. On all benchmark datasets, CELOE requires to explore at least $\mathbf{3.19}\times$ more concepts than NERO. This resulted in reducing the total computation time of NERO by $3-6\times$ on all benchmark datasets. We conducted the Wilcoxon signed-rank tests (one and two-sided) on F_1 scores, runtimes, and the number of explored expression. We were able to reject the null hypothesis with a p-value $< 1\%$ across all the datasets. Therefore, the superior performance of NERO is statistically significant.

7 Discussion

Our results uphold our hypothesis: implicit knowledge encoded in (E^+, E^-) about many $\mathcal{ALC}$ expression can be leveraged to mitigate the need of excessive exploration. Through learning permutation-invariant representations for a set of individuals, we can effectively predict quality scores for pre-selected $\mathcal{ALC}$ concepts. Throughout our experiments, NERO consistently outperforms state-of-the-art models w.r.t. the F_1 score, the number of expression retrievals and the total computational time. Our experiments also indicate that the superior performance of NERO is statistically significant.

It is important to note that it has been proven that CELOE is complete in the CEL problem, i.e., for a given learning problem, CELOE finds a goal expression if it exists provided that there are no upper-bounds on the time and memory requirements [21]. Although these requirements are simply not practical, being not complete can be considered as a drawback of Hence, we argue that NERO can be used initialize the search tree of CELOE and using CELOE to conduct the search may be necessary to fulfill the completeness criterion.

8 Conclusion

We introduced NERO, a novel neural CEL approach to accelerate the problem of learning $\mathcal{ALC}$ concepts. NERO effectively detects adequate $\mathcal{ALC}$ concepts without excessive exploration. Our experiments show that NERO outperforms state-of the art models in 770 CEL problems on 5 benchmark datasets w.r.t. the quality of found concepts, number of retrievals and the total computational time. The results of our statistical tests (one- and two-sided Wilcoxon signed rank tests) confirm the superior performance of NERO.

We strongly believe that incorporating neural models in class expression learning problems is worth pursuing further. Here, we focused mainly on learning permutation-embeddings tailored towards predicting quality of predefined $\mathcal{ALC}$ expressions. Yet, learning embeddings tailored towards more expressive DLs expressions may lead interesting results.

9 Submission of papers to NeurIPS 2022

Please read the instructions below carefully and follow them faithfully.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See the ac **??**.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] see Section 7
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] see Section 7
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes] see section 1
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] see the supplemental material
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] see the supplemental material
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] see the supplemental material
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] we included our pretrained models
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]