Is the Proof Length a Good Indicator of Hardness for Reason-able Embeddings?

Jedrzej Potoniec[1]

[1]Institute of Computing Science, Poznan University of Technology, ul. Piotrowo 2, 60-965 Poznan, Poland

Abstract
Reason-able embeddings are recently proposed embeddings for knowledge bases (KBs) in the description logic $\mathcal{ALC}$ capable of casting multiple KBs into a single latent space using a transferable neural reasoner. While they exhibit remarkable performance on real-world KBs, it is so far unknown what are their exact limits. In this paper, we systematically investigate their performance using a set of synthetic KBs in the description logic $\mathcal{EL}$, a subset of $\mathcal{ALC}$. We use the proof length as a measure of reasoning complexity and present a random KB generator taking the proof length into account. We train the reason-able embeddings with and without transfer learning and investigate whether the complexity of the training set and the test set is related to the reasoning performance of the embeddings and their neural reasoner.

Keywords
description logics, reason-able embeddings, transfer learning, neural-symbolic reasoning

1. Introduction

The past 10 years brought numerous solutions for embedding concepts from knowledge bases (KBs) into high-dimensional latent spaces, from a relatively simple TransE to complex architectures such as recurrent transformers [1, 2, 3, 4, 5]. However, with a few notable exceptions such as RDF2Vec [6], OWL2Vec* [7], and TransOWL [8] the focus was on large KBs with shallow semantics. On the other side, there were approaches to replace deductive approaches traditionally used to perform inference from logical KBs, with neural reasoners based on the recent advancements in deep learning, e.g., [9, 10, 11, 12].

Recently proposed reason-able embeddings are a combination of these two trends, offering learnable embeddings for KBs with complex semantics and a transferable, neural reasoner [13]. In experimental evaluation with a set of real-world KBs, they exhibited remarkable performance, yet there remains a suspicion that due to the complexity of the description logic (DL) $\mathcal{ALC}$ they target, they cannot address all the aspects of $\mathcal{ALC}$. In this paper, we aim to tackle this problem, by constructing a set of artificial KBs with measurable complexity and assessing the performance of reason-able embeddings as a function of the complexity of their input. Our goal is to answer the following research questions:

NeSy 2023, 17th International Workshop on Neural-Symbolic Learning and Reasoning, Certosa di Pontignano, Siena, Italy

jedrzej.potoniec@put.poznan.pl (J. Potoniec)
0000-0002-6115-6485 (J. Potoniec)

© 2023 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).
CEUR Workshop Proceedings (CEUR-WS.org)

RQ1 Does the complexity of the training set influence the reasoning performance on the test set derived from the same KBs?

RQ2 Does the complexity of the training set of the embeddings influence the reasoning performance on previously unseen KBs?

RQ3 What is the interplay between the complexity of the training set for the reasoner, the complexity of the training set for the embeddings, and the reasoning performance on previously unseen KBs?

To facilitate reproducibility we publish the experimental code in Python in a GitHub repository at https://github.com/jpotoniec/reasonable-embeddings/tree/main/src/elpp.

The contributions of the paper are as follows:

- A way to measure reasoning complexity by means of the proof length, described in subsection 2.2. It exploits a preexisting reasoning algorithm by Baader et al. [14].
- A generator of synthetic, difficult knowledge bases, described in subsection 2.4.
- A method to split training and test sets according to the reasoning complexity, described in subsection 2.5.
- Experimental evaluation and analysis presented in section 3.

Neither the description logic $\mathcal{EL}$ (presented in subsection 2.1) nor the reason-able embeddings (described in subsection 2.3) are contributions of this work, and they are presented here only to make the paper self-contained.

The rest of the paper is organized as follows: subsection 2.1 introduces the DL $\mathcal{EL}$, subsection 2.2 explains how we measure the complexity, the reason-able embeddings are introduced in subsection 2.3, we describe the KB generator in subsection 2.4, and the details of the employed dataset in subsection 2.5. In subsections of section 3, we report the answers to the posed research questions. We discuss the results and conclude in section 4.

2. Materials and methods

2.1. The description logic $\mathcal{EL}$

Description logics (DLs) are a family of logics developed for their desirable computational properties, to be used in computer systems requiring formal semantics and reasoning capabilities. In this paper, we concentrate on the DL $\mathcal{EL}$, one of the simplest of practical use [14].

Concepts in $\mathcal{EL}$ are constructed inductively from a set of constructors, a set of atomic concepts N_C, and a set of atomic roles N_R. Their semantics is defined in terms of an *interpretation* $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$, where $\Delta^\mathcal{I}$ is a non-empty *domain*, and $\cdot^\mathcal{I}$ is the *interpretation function*, mapping every atomic concept A to $A^\mathcal{I} \subseteq \Delta^\mathcal{I}$, and every atomic role r to $r^\mathcal{I} \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{I}$. Since $\mathcal{EL}$ has formal semantics, it is possible to extend $\cdot^\mathcal{I}$ to arbitrary concepts, summarized in Table 1.

A $\mathcal{EL}$ KB consists of a set of subsumption axioms $C \sqsubseteq D$, and we say that $\mathcal{I}$ is a *model* of the KB if for every such axiom $C^\mathcal{I} \subseteq D^\mathcal{I}$. In the DLs, one of the core problems is *the subsumption problem*, i.e., given two concepts C, D, and a KB KB, deciding whether a subsumption $C \sqsubseteq D$ holds in every model of the KB, denoted $KB \models C \sqsubseteq D$. One of the interesting properties of $\mathcal{EL}$ is that there exists a polynomial algorithm for solving the subsumption problem.

Table 1
Syntax and semantics of $\mathcal{EL}$. C, D denote arbitrary concepts and r an arbitrary role.

Name	Syntax	Semantics
top	$\top$	$\Delta^{\mathcal{I}}$
bottom	$\bot$	$\emptyset$
conjunction	$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
existential restriction	$\exists r.C$	$\{x \in \Delta^{\mathcal{I}} : \exists y \in \Delta^{\mathcal{I}}\, (x, y) \in r^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}$

In this paper, we call a subsumption $C \sqsubseteq D$ an *atomic subsumption* if $C, D \in N_C \cup \{\bot, \top\}$. If $KB \models C \sqsubseteq D$, we say the subsumption *follows* from the KB.

Throughout this work, we are interested in KBs in the *normal form*, where the axioms must be of one of the following forms: $C \sqsubseteq E$, $C \sqcap D \sqsubseteq E$, $C \sqsubseteq \exists r.D$, $\exists r.C \sqsubseteq D$, where $C, D \in N_C \cup \{\top\}$ and $E \in N_C \cup \{\top, \bot\}$. It was shown that any $\mathcal{EL}$ KB can be transformed to a KB in normal form in polynomial time.

2.2. Measuring the proof length

Since we consider only KBs in the normal form, to solve the subsumption problem, we can apply a rule-based algorithm presented in [14] for $\mathcal{EL}^{++}$, a superset of $\mathcal{EL}$. The algorithm performs *classification* of a KB, i.e., it computes all atomic subsumptions following from the KB. We refer to this set of atomic subsumption as *logical consequences* of the KB[1].

We extend the algorithm to keep track, for every atomic subsumption, of the shortest *proof*, i.e, a set of rules and their parameters that must be executed in order to generate this particular conclusion. The algorithm starts by creating a set $S(C) = \{(C, \emptyset), (\top, \emptyset)\}$ for every $C \in N_C \cup \{\top, \bot\}$, and a set $R(r) = \emptyset$ for every $r \in N_R$. Then, rules presented in Table 2 are executed to update the sets $S(C)$ and $R(r)$ until a fixpoint is not reached, i.e., there are changes to $S(C)$ or to $R(r)$. After the algorithm terminates, $S(C)$ represents the subsumption hierarchy: if $(D, P) \in S(C)$, it means that $KB \models C \sqsubseteq D$ and at least $|P|$ rule executions are necessary to prove it. We call $|P|$ the *proof length* of $C \sqsubseteq D$ and use it as a complexity measure of atomic subsumptions throughout the paper. It must be noted that this measure is defined only for subsumptions that follow from a KB and that it measures the length of the shortest proof, not any proof. However, since it is supposed to measure reasoning complexity, we believe the shortest proof is the right choice, as it denotes the minimal number of reasoning steps one must make in order to prove a given subsumption, and all the rules seem to be of comparable complexity.

As an example, consider the following KB: $KB = \{\alpha \sqsubseteq \beta, \alpha \sqsubseteq \gamma, \beta \sqcap \gamma \sqsubseteq \delta, \alpha \sqsubseteq \exists r.\beta, \exists r.\beta \sqsubseteq \delta, \beta \sqsubseteq \bot\}$. Unconventionally, we use Greek letters to clearly distinguish concept names present in the KB from the variable names used in the rules. We will consider a few steps of the algorithm, and we chose one possible order of execution for ease of explanation. First, sets $S(\cdot)$ and $R(\cdot)$ are initialized: $S(\alpha) = \{(\alpha, \emptyset), (\top, \emptyset)\}, S(\beta) = \{(\beta, \emptyset), (\top, \emptyset)\}, S(\gamma) = \{(\gamma, \emptyset), (\top, \emptyset)\}, S(\delta) = \{(\delta, \emptyset), (\top, \emptyset)\}, R(r) = \emptyset$. Next, the rules are executed:

[1]The term usually denotes a much broader set. However, we concentrate only on atomic subsumptions in the paper and thus can hack the term without introducing ambiguity.

Table 2
Reasoning rules used to classify a KB and compute proof lengths for every atomic subsumption following from the KB. Each rule is first presented using formal symbols, and then a corresponding textual description is given. $(A, \cdot) \notin B$ is an abbreviation for $\forall y\, (A, y) \notin B$.

CR1	If $(C', P_{C'}) \in S(C)$, $C' \sqsubseteq D \in KB$, $(D, \cdot) \notin S(C)$ or $(D, P_D) \in S(C)$ and $\lvert NP\rvert < \lvert P_D\rvert$, then $S(C) \leftarrow S(C) \backslash \{(D, P_D)\} \cup \{(D, NP)\}$, where $NP = P_{C'} \cup \{(\text{CR1}, C, C', D)\}$ If C' is known to be a superclass of C, it is asserted in the KB that $C' \sqsubseteq D$, and either D is not known to be a superclass of C or it is known, but the proof is longer than can be achieved by using this rule, then add/replace the existing proof with the new proof NP.
CR2	If $(C_1, P_{C_1}), (C_2, P_{C_2}) \in S(C)$, $C_1 \sqcap C_2 \sqsubseteq D \in KB$, $(D, \cdot) \notin S(C)$ or $(D, P_D) \in S(C)$ and $\lvert NP\rvert < \lvert P_D\rvert$, then $S(C) \leftarrow S(C) \backslash \{(D, P_D)\} \cup \{(D, NP)\}$, where $NP = P_{C_1} \cup P_{C_2} \cup \{(\text{CR2}, C, C_1, C_2, D)\}$ If it is known that C_1 and C_2 are superclasses of C, it is asserted in the KB that $C_1 \sqcap C_2 \sqsubseteq D$, and either D is not known to be a superclass of C or it is known, but the proof is longer than can be achieved by using this rule, then add/replace the existing proof with the new proof NP.
CR3	If $(C', P_{C'}) \in S(C)$, $C' \sqsubseteq \exists r.D \in KB$, $((C, D), \cdot) \notin R(r)$ or $((C, D), P_{CD}) \in R(r)$ and $\lvert NP\rvert < \lvert P_{CD}\rvert$, then $R(r) \leftarrow R(r) \backslash \{((C, D), P_{CD})\} \cup \{((C, D), NP)\}$, where $NP = P_{C'} \cup \{(CR3, C, C', r, D)\}$ If C' is known to be a superclass of C, it is asserted in the KB that $C' \sqsubseteq \exists r.D$, and either C is not known to be a subclass of $\exists r.D$ or it is known, but the proof is longer than can be achieved by using this rule, then add/replace the existing proof with the new proof NP.
CR4	If $((C, D), P_{CD}) \in R(r)$, $(D', P_{D'}) \in S(D)$, $\exists r.D' \sqsubseteq E \in KB$, $(E, \cdot) \notin S(C)$ or $(E, P_E) \in S(C)$ and $\lvert NP\rvert < \lvert P_E\rvert$, then $S(C) \leftarrow S(C) \backslash \{(E, P_E)\} \cup \{(E, NP)\}$, where $NP = P_{CD} \cup P_{D'} \cup \{(\text{CR4}, C, D, r, D', E)\}$ If C is known to be a subclass of $\exists r.D$, D' is known to be a superclass of D, it is asserted in the KB that $\exists r.D' \sqsubseteq E$, and either E is not known to be a superclass of C or it is known, but the proof is longer than can be achieved by using this rule, then add/replace the existing proof with the new proof NP.
CR5	If $((C, D), P_{CD}) \in R(r)$, $(\bot, P_D) \in S(D)$, $(\bot, \cdot) \notin S(C)$ or $(\bot, P_C) \in S(C)$ and $\lvert NP\rvert < \lvert P_C\rvert$, then $S(C) \leftarrow S(C) \backslash \{(\bot, P_C)\} \cup \{(\bot, NP)\}$, where $NP = P_{CD} \cup P_D \cup \{(\text{CR5}, C, D, r)\}$ If C is known to be a subclass of $\exists r.D$, D is known to be an unsatisfiable concept, and either C is not known to be an unsatisfiable concept or it is known, but the proof is longer than can be achieved by using this rule, then add/replace the existing proof with the new proof NP.

1. CR1 for $C = C' = \alpha$, $D = \beta$ and $P_{C'} = \emptyset$. The conditions are satisfied: $(\alpha, \emptyset) \in S(\alpha)$, $\alpha \sqsubseteq \beta \in KB$ and $(\beta, \cdot) \notin S(\alpha)$. The new proof can thus be computed as $NP = \emptyset \cup \{(\text{CR1}, \alpha, \alpha, \beta)\}$ and included in the set $S(\alpha)$: $S(\alpha) = \{(\alpha, \emptyset), (\top, \emptyset), (\beta, \{(\text{CR1}, \alpha, \alpha, \beta)\})\}$
2. CR1 for $C = C' = \alpha$, $D = \gamma$ and $P_{C'} = \emptyset$. The conditions are satisfied: $(\alpha, \emptyset) \in S(\alpha)$, $\alpha \sqsubseteq \gamma \in KB$ and $(\gamma, \cdot) \notin S(\alpha)$. The new proof can thus be computed as $NP = \emptyset \cup \{(\text{CR1}, \alpha, \alpha, \gamma)\}$ and thus: $S(\alpha) = \{(\alpha, \emptyset), (\top, \emptyset), (\beta, \{(\text{CR1}, \alpha, \alpha, \beta)\}), (\gamma, \{(\text{CR1}, \alpha, \alpha, \gamma)\})\}$
3. CR2 for $C = \alpha, C_1 = \beta, C_2 = \gamma, D = \delta, P_{C_1} = \{(\text{CR1}, \alpha, \alpha, \beta)\}, P_{C_2} = \{(\text{CR1}, \alpha, \alpha, \gamma)\}$. The conditions are satisfied: $(\beta, P_{C_1}) \in S(\alpha))$, $(\gamma, P_{C_2}) \in S(\alpha)$, $\beta \sqcap \gamma \sqsubseteq \delta \in KB$, $(\delta, \cdot) \notin S(A)$. The new proof can thus

be computed as $NP = P_{C_1} \cup P_{C_2} \cup \{(\text{CR2}, \alpha, \beta, \gamma, \delta)\}$, and we arrive at: $S(\alpha) = \{(\alpha, \emptyset), (\top, \emptyset), (\beta, \{(\text{CR1}, \alpha, \alpha, \beta)\}), (\gamma, \{(\text{CR1}, \alpha, \alpha, \gamma)\}), (\delta, \{(\text{CR1}, \alpha, \alpha, \beta), (\text{CR1}, \alpha, \alpha, \gamma), (\text{CR2}, \alpha, \beta, \gamma, \delta)\})\}$

4. CR3 for $C = C' = \alpha, D = \beta, P_{C'} = \emptyset$. The conditions are satisfied: $(\alpha, \emptyset) \in S(\alpha)$, $\alpha \sqsubseteq \exists r.\beta \in KB$, $((\alpha, \beta), \cdot) \notin R(r)$. The new proof can thus be computed as $NP = \{(CR3, \alpha, \alpha, r, \beta)\}$ and $R(r)$ can be updated: $R(r) = \{((\alpha, \beta), \{(CR3, \alpha, \alpha, r, \beta)\})\}$.
5. CR4 for $C = \alpha, D = \beta, D' = \beta, E = \delta, P_{CD} = \{(CR3, \alpha, \alpha, r, \beta)\}, P_{D'} = \emptyset, P_E = \{(\text{CR1}, \alpha, \alpha, \beta), (\text{CR1}, \alpha, \alpha, \gamma), (\text{CR2}, \alpha, \beta, \gamma, \delta)\}$. The conditions are satisfied: $((\alpha, \beta), P_{CD}) \in R(r)$, $(\beta, \emptyset) \in S(\beta)$, $\exists r.\beta \sqsubseteq \delta \in KB$, and $(\delta, P_E) \in S(\alpha)$. This application of a rule is different from previous ones: there already is a proof (P_E), and it must be determined whether it can be replaced. The new proof is thus computed as $NP = \{(CR3, \alpha, \alpha, r, \beta)\} \cup \emptyset \cup \{(CR4, \alpha, \beta, r, \beta, \delta)\}$, and we conclude it is shorter than the pre-existing proof, since $|P_E| = 3 > |NP| = 2$. We thus proceed with updating $S(\alpha)$, which now becomes: $S(\alpha) = \{(\alpha, \emptyset), (\top, \emptyset), (\beta, \{(\text{CR1}, \alpha, \alpha, \beta)\}), (\gamma, \{(\text{CR1}, \alpha, \alpha, \gamma)\}), (\delta, \{(CR3, \alpha, \alpha, r, \beta), (CR4, \alpha, \beta, r, \beta, \delta)\})\}$
6. CR1 for $C = C' = \beta, D = \bot, P_{C'} = \emptyset$. The conditions are satisfied: $(\beta, \emptyset) \in S(\beta)$, $\beta \sqsubseteq \bot \in KB$ and $(\bot, \cdot) \notin S(\beta)$. The new proof can thus be computed as $NP = \emptyset \cup \{(\text{CR1}, \beta, \beta, \bot)\}$ and included in the set $S(\beta)$: $S(\beta) = \{(\beta, \emptyset), (\top, \emptyset), (\bot, \{(\text{CR1}, \beta, \beta, \bot)\})\}$.
7. CR5 for $C = \alpha, D = \beta, P_{CD} = \{(CR3, \alpha, \alpha, r, \beta)\}, P_D = \{(\text{CR1}, \beta, \beta, \bot)\}$. The conditions are satisfied: $((\alpha, \beta), P_{CD}) \in R(r)$ and $(\bot, P_D) \in S(\beta)$. The new proof can thus be computed as $NP = \{(CR3, \alpha, \alpha, r, \beta)\} \cup \{(\text{CR1}, \beta, \beta, \bot)\} \cup \{(CR5, \alpha, \beta, r)\}$ and $S(\alpha)$ can be extended with $(\bot, NP)$.

The presented example covers all the proposed rules and the crucial part, namely, replacing longer proofs with shorter ones. While it does not present the entirety of the execution for the given KB, nor the termination condition by reaching a fixpoint, we can still read proof lengths (albeit - since the algorithm did not terminate - not necessarily the shortest), as seen in item 5.

A similar idea is present in the research on DLs under the term *justifications*, which concentrates on computing a minimal subset of KB for a subsumption to follow from it [15]. The proofs we compute are more complex, e.g., because the rule CR5 does not reference any KB axiom, yet is included in proofs. The purpose is also different: justifications are meant as means of explanations for humans, whereas here proofs are an internal artifact to assess complexity.

Exploiting the properties of rule-based reasoning for lightweight DLs was also used in the context of finding unexplainable triples in RDF graphs [16] or in the work on emulating deductive reasoning with recurrent neural networks [17].

2.3. Reason-able embeddings

Reason-able embeddings are a recently proposed framework to compute embedding vectors for concepts in the $\mathcal{ALC}$ DL [13]. They have several interesting properties that differentiate them from pre-existing embeddings. First, they are tailored to an expressive DL, instead of to a simple semantics of a knowledge graph. Second, the framework consists of two major components: the embeddings themselves which must be trained for each KB separately, and a neural reasoner, which is shared between multiple KBs, and once trained can be reused in a transfer learning manner for reasoning on KBs not available during its training. Third, they are capable of computing an embedding of a complex expression given the embeddings of its parts.

The neural reasoner consists of two parts: a reasoner head, which is a binary classifier that, given two concept embeddings from the same KB, answers whether one of the concepts is a subclass of the other; and a set of neural operators, which correspond to the logical operators of $\mathcal{ALC}$ and can, e.g., compute the embedding of an intersection of two concepts given the embeddings of those concepts.

In [13], a two-step training procedure was proposed: first, a neural reasoner and embeddings are trained jointly on a set of KBs. Multiple KBs are employed during this step to ensure the reasoner is not tailored to any particular KB but rather is capable of shaping the latent space of embeddings such that it can accommodate multiple KBs. In the second step, the reasoner is frozen and the embeddings for the KB(s) of interest are trained by backpropagation through the frozen reasoner.

Both training and testing a neural reasoner require subsumptions along with a binary label denoting whether a subsumption follows from the corresponding KB or not. We call such a pair an *example*. Since the label is contextual w.r.t. a KB, it is necessary to keep track of which example is related to which KB. However, this information is not a part of an example and is not presented to the neural reasoner during training. The subsumptions themselves may be synthetic, as long as the label associated with them is correct.

A two-step procedure requires two-level separation into training and test set: first, there is a training set of KBs used to jointly train the reasoner and the embeddings, and there is a test set of KBs used to assess the quality of the trained, frozen reasoner. However, embeddings for the KBs in the test must somehow be trained. Thus, for each KB, be it from the training set or from the test set, a training set and a test set of examples must be constructed. The training set is used to train embeddings for the KB, while the test set is used to evaluate them.

There are numerous hyperparameters that could be tuned in the framework. In this paper, we follow the setup from [13]: the latent space of embeddings has a dimension of 10, there are 16 neurons in the hidden layer of the reasoner head, and the training takes 15 epochs.

2.4. Synthetic knowledge bases generator

To generate a random KB, we start by defining an approach to generate a random axiom in the context of a KB KB. First, we select with uniform probability one of the following forms for the axiom: $A \sqsubseteq C$, $A \sqcap B \sqsubseteq C$, $D \sqsubseteq \exists r.E$, $\exists r.D \sqsubseteq E$. Then, we populate the form with concrete values by selecting uniformly at random: A, B, C from N_C, r from N_R, D, E from $N_C \cup \{\bot, \top\}$.

To construct difficult KBs, i.e., KBs such that some of the atomic subsumptions following from them have long proofs, we have devised the following algorithm. First, an empty KB is initialized with n_c concepts and n_r roles. Then, we repeat at most n_{rep} times: an axiom not yet present in the KB is randomly generated and added to the KB, then the logical consequences with the proof length of at least l_{min} are counted, and if there are at least n_{min}, we terminate and return the KB. If we exceed n_{rep} without reaching n_{min}, the process is restarted from an empty KB.

Table 3
Statistics on the generated synthetic KBs. Two dimensions are considered: dataset (training/test set), and minimal proof length (≥ 0 denotes the set of all logical consequences, incl. those explicitly asserted in the KB; ≥ 2 denotes the set of non-asserted logical consequences, i.e., those with the proof length at least 2; ≥ 1 - not present in the table - is essentially equal to ≥ 0, except for those consequences that are starting points for the reasoning algorithm, i.e., $C \sqsubseteq C$ and $C \sqsubseteq \top$ for any $C \in N_C \cup \{\top, \bot\}$). Two measures are computed: the number of logical consequences, denoted *count*, and their proof lengths, denoted *proof length* and reported as mean $\pm$ standard deviation[min, max]. The measures are averaged in two ways: *macro-average* meaning we first compute the average for each KB separately and report the average of averages; and *micro-average* meaning we directly average over the logical consequences.

	macro-average		micro-average	
	count	proof length	count	proof length
training ≥ 0	$1124 \pm 1430[498, 7930]$	$9.32 \pm 9.14[2.56, 37.22]$	$44,951$	$17.78 \pm 17.97[0, 87]$
training ≥ 2	$971 \pm 1427[246, 7670]$	$12.71 \pm 9.54[4.17, 42.04]$	$34,821$	$22.89 \pm 17.34[2, 87]$
test ≥ 0	$1423 \pm 1861[520, 7461]$	$9.99 \pm 9.76[3.45, 37.44]$	$28,458$	$18.69 \pm 16.94[0, 83]$
test ≥ 2	$1167 \pm 1860[261, 7196]$	$13.50 \pm 9.80[6.07, 43.24]$	$23,348$	$22.73 \pm 16.09[2, 83]$

In the paper we fixed $n_c = 100$, $n_r = 4$, $n_{rep} = 300$, $l_{min} = 4$ and $n_{min} = 200$. The choice of n_c and n_r was guided by the original work on reason-able embeddings [13], the others were chosen arbitrarily, as they seemed to represent a good trade-off between the generation time and the complexity of the final KBs. We use two sets of synthetic KBs: a training set of 40 KBs $Tr_1, \dots, Tr_{40}$, and a test set of 20 KBs $Te_1, \dots, Te_{20}$. We report detailed statistics on them in Table 3.

2.5. Training and test set split

For each KB, we generate several training – test splits using the following procedure. Let $I^{\leq j}$ be the set of all atomic subsumptions following from a KB KB such that their proofs are no longer than j. Conversely, let $I^{>j}$ be the set of all atomic subsumptions following from the KB with the proof length greater than j. Let $\tilde{I}$ be the set of all atomic subsumptions $A \sqsubseteq B$ that uses the vocabulary of the KB, do not follow from the KB nor the atomic subsumption $B \sqsubseteq A$ follow from the KB:

$$\tilde{I} = \{A \sqsubseteq B \colon A, B \in N_C \cup \{\bot, \top\} \wedge KB \not\models A \sqsubseteq B \wedge KB \not\models B \sqsubseteq A\}$$

This final requirement on $\tilde{I}$ is introduced to ensure that knowledge from $\tilde{I}$ cannot be trivially exploited for information about $I^{>j}$. Since both I and $\tilde{I}$ consist of atomic subsumptions, they cannot be separated syntactically and some semantic information from the labels must be exploited by the neural reasoner.

The *j-th training set* for KB consists of the whole set $I^{\leq j}$, labeled with the positive label, and a random subset of $\tilde{I}$ counting $\left|I^{\leq j}\right|$ to achieve a balanced training set and labeled with the negative label. The *j-th test set* for KB is equal to $I^{>j}$ labeled with the positive label. There are no negative examples in the test set, as we have no complexity measure for subsumptions that do not follow from a KB, and thus the answers of a neural reasoner would offer no insight into the RQs. We thus use *recall*, i.e., the number of examples labeled as positive by the neural reasoner to the number of positive examples in the test set, as the measure to assess the performance.

Since we have two levels of training-test splits, there is a tremendous possibility for confusion. We thus introduce the following symbols and names: $Tr_i^{\leq j}$ (resp. $Te_i^{\leq j}$) denotes the j-th training set for the KB Tr_i (resp. Te_i), and $Tr_i^{>j}$ (resp. $Te_i^{>j}$) denotes the j-th test set for the KB Tr_i (resp. Te_i). The *j-th joint training set* is $Tr^{\leq j} = \bigcup_{i=1}^{40} Tr_i^{\leq j}$, and the *$j$-th embeddings training set* is $Te^{\leq j} = \bigcup_{i=1}^{20} Te_i^{\leq j}$. We use the name *joint* to underline that both embeddings and the neural reasoner are trained jointly using these training sets; conversely, the embeddings training sets are used to train embeddings while the neural reasoner is already trained and frozen. The *j-th internal test set* is $Tr^{>j} = \bigcup_{i=1}^{40} Tr_i^{>j}$, and the *$j$-th external test set* $Te^{>j} = \bigcup_{i=1}^{20} Te_i^{>j}$. Here, *internal* underlines that the neural reasoner was trained with these KBs, whereas *external* denotes KBs that did not participate in the training of the neural reasoner.

3. Experimental results

3.1. The influence of complexity of the joint training set on the performance on the internal test set

To answer RQ1, we employed the joint training sets $Tr^{\leq j}$ and their respective internal test sets $Tr^{>j}$ for $j \in \{2, 3, \ldots, 20\}$. For each $Tr^{\leq j}$, we trained a neural reasoner using the training procedure described earlier, and tested it using $Tr^{>j}$, arriving at the average recall $0.701 \pm 0.044\,[0.587, 0.764]$. We then computed the Pearson correlation coefficient ρ between j and the recall, arriving at $\rho = 0.931$ and the p-value $< 10^{-5}$ ($H_0\colon \rho = 0$, using the exact probability distribution as computed by SciPy[2]).

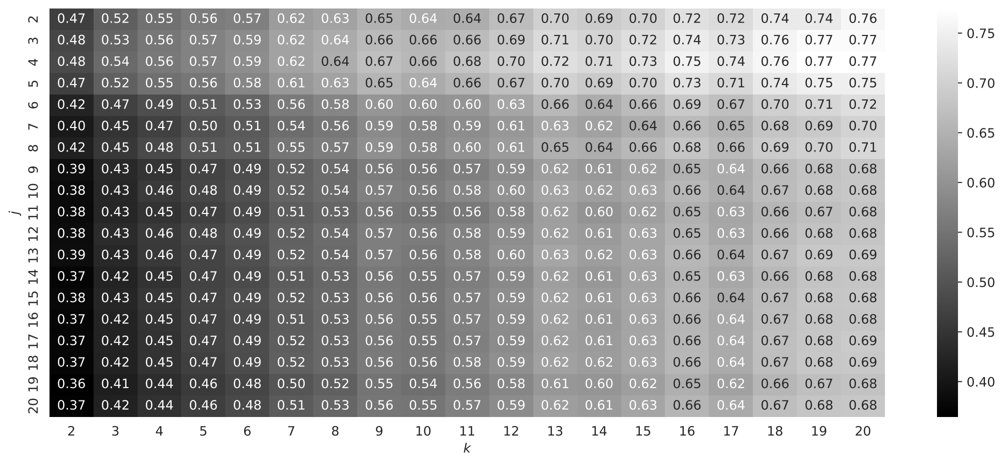
To ensure this is not due to using different test sets for each neural reasoner (since $Tr^{>j} \supseteq Tr^{>j+1}$), we also tested all the neural reasoners on the common test set $Tr^{>20}$, arriving at $\rho = 0.920$ and the p-value $< 10^{-5}$.

We have decided not to use more complicated approaches for measuring the influence since the Pearson correlation coefficients are high enough (and the p-values low enough) to give us reasonable confidence that the answer to RQ1 is positive and the complexity of the training set is an indicator of the performance on the test set even when transfer learning is not used.

3.2. The influence of complexity of the embeddings training set on the performance on the external test set

To answer RQ2, we used the neural reasoner trained with $Tr^{\leq 20}$, i.e., presumably the best one. We froze it and used it to train embeddings using the embeddings test sets $Te^{\leq k}$ for $k \in \{2, 3, \ldots, 20\}$. Then, we tested the neural reasoner and the newly trained embeddings on the respective external test sets $Te^{>k}$. Averaging the recall, we arrived at $0.610 \pm 0.094\,[0.387, 0.699]$. The Pearson correlation coefficient between k and recall is $\rho = 0.913$, with the p-value $< 10^{-5}$ (the same test as for RQ1). Similarly to the previous experiment, we also tested all the embeddings on the common subset $Te^{>20}$, arriving at $\rho = 0.982$ with the p-value $< 10^{-5}$.

[2]https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.pearsonr.html



j \ k	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
2	0.47	0.52	0.55	0.56	0.57	0.62	0.63	0.65	0.64	0.64	0.67	0.70	0.69	0.70	0.72	0.72	0.74	0.74	0.76
3	0.48	0.53	0.56	0.57	0.59	0.62	0.64	0.66	0.66	0.66	0.69	0.71	0.70	0.72	0.74	0.73	0.76	0.77	0.77
4	0.48	0.54	0.56	0.57	0.59	0.62	0.64	0.67	0.66	0.68	0.70	0.72	0.71	0.73	0.75	0.74	0.76	0.77	0.77
5	0.47	0.52	0.55	0.56	0.58	0.61	0.63	0.65	0.64	0.66	0.67	0.70	0.69	0.70	0.73	0.71	0.74	0.75	0.75
6	0.42	0.47	0.49	0.51	0.53	0.56	0.58	0.60	0.60	0.60	0.63	0.66	0.64	0.66	0.69	0.67	0.70	0.71	0.72
7	0.40	0.45	0.47	0.50	0.51	0.54	0.56	0.59	0.58	0.59	0.61	0.63	0.62	0.64	0.66	0.65	0.68	0.69	0.70
8	0.42	0.45	0.48	0.51	0.51	0.55	0.57	0.59	0.58	0.60	0.61	0.65	0.64	0.66	0.68	0.66	0.69	0.70	0.71
9	0.39	0.43	0.45	0.47	0.49	0.52	0.54	0.56	0.56	0.57	0.59	0.62	0.61	0.62	0.65	0.64	0.66	0.68	0.68
10	0.38	0.43	0.46	0.48	0.49	0.52	0.54	0.57	0.56	0.58	0.60	0.63	0.62	0.63	0.66	0.64	0.67	0.68	0.68
11	0.38	0.43	0.45	0.47	0.49	0.51	0.53	0.56	0.55	0.56	0.58	0.62	0.60	0.62	0.65	0.63	0.66	0.67	0.68
12	0.38	0.43	0.46	0.48	0.49	0.52	0.54	0.57	0.56	0.58	0.59	0.62	0.61	0.63	0.65	0.63	0.66	0.68	0.68
13	0.39	0.43	0.46	0.47	0.49	0.52	0.54	0.57	0.56	0.58	0.60	0.63	0.62	0.63	0.66	0.64	0.67	0.69	0.69
14	0.37	0.42	0.45	0.47	0.49	0.51	0.53	0.56	0.55	0.57	0.59	0.62	0.61	0.63	0.65	0.63	0.66	0.68	0.68
15	0.38	0.43	0.45	0.47	0.49	0.52	0.53	0.56	0.56	0.57	0.59	0.62	0.61	0.63	0.66	0.64	0.67	0.68	0.68
16	0.37	0.42	0.45	0.47	0.49	0.51	0.53	0.56	0.55	0.57	0.59	0.62	0.61	0.63	0.66	0.64	0.67	0.68	0.68
17	0.37	0.42	0.45	0.47	0.49	0.52	0.53	0.56	0.55	0.57	0.59	0.62	0.61	0.63	0.66	0.64	0.67	0.68	0.69
18	0.37	0.42	0.45	0.47	0.49	0.52	0.53	0.56	0.56	0.58	0.59	0.62	0.62	0.63	0.66	0.64	0.67	0.68	0.69
19	0.36	0.41	0.44	0.46	0.48	0.50	0.52	0.55	0.54	0.56	0.58	0.61	0.60	0.62	0.65	0.62	0.66	0.67	0.68
20	0.37	0.42	0.44	0.46	0.48	0.51	0.53	0.56	0.55	0.57	0.59	0.62	0.61	0.63	0.66	0.64	0.67	0.68	0.68

Figure 1: Recall for different combinations of the complexity of the joint training set j and the complexity of the embeddings training set k.

These results strongly indicate that the answer to RQ2 is positive, and the complexity of the embeddings training set is a good indicator of the reasoning performance on the test set when transfer learning is employed.

3.3. The interplay between the complexity of the joint training set, the complexity of the embeddings training set, and the performance on the external test set

After seeing RQ1 and RQ2, a natural question about the interplay of the complexity j of the joint training set and the complexity k of the embeddings training set arises. To answer RQ3, we have computed recall for all combinations of $j \in \{2, 3, \ldots, 20\}$ and $k \in \{2, 3, \ldots, 20\}$, by first training the neural reasoner using $Tr^{\leq j}$, then training embeddings on $Te^{\leq k}$, and finally testing them using $Te^{>k}$. We report the results in Figure 1 and observe that they are somewhat counterintuitive. It turns out that for fixed k, the higher j the worse the recall. Conversely, for fixed j, the higher k the better the recall.

We hypothesize that, while using only atomic subsumptions, the reasoning capabilities of the neural reasoner are not necessary and the most important part is the interplay of different dimensions in the latent space of embeddings. Thus, on the one hand, higher values of k cause the embeddings to be more attuned to the properties of the latent space, whereas the lower values of j cause the reasoner to shape the latent space less strictly.

We note in passing that, following [18], we have conducted the Friedman test assuming every KB is a separate dataset, and every pair (j, k) constitutes a separate classifier. The test indicated there are differences between classifiers (the p-value $< 10^{-5}$). However, deciding which pair-wise differences are significant would require making $\binom{19^2}{2} = 64,980$ comparisons and thus it would be extremely hard to achieve reliable results.

4. Discussion and conclusions

In the paper, we have presented an approach to generate non-trivial KBs in the $\mathcal{EL}$ DL and to measure the complexity of their logical conclusions. We then used these results to evaluate the influence of the complexity on the performance of reason-able embeddings, a recently proposed framework offering a transferable neural reasoner for deciding the subsumption problem in the $\mathcal{ALC}$ DL.

We have shown that within the assumed framework the complexity of the training set is a good performance indicator in scenarios both employing and not employing transfer learning. Moreover, the experimental results indicate that the complexity of the embeddings training set is much more important than the complexity of the joint training set, and the latter can even influence the performance negatively, possibly because it constrains the latent space too tightly.

The study has numerous limitations: we have concentrated on a very simple DL $\mathcal{EL}$, we considered only atomic subsumptions in the training and test sets, and measured the complexity of only subsumptions following from the KBs. The first assumption limits the generalizability of this study, but since $\mathcal{EL}$ is a proper subset of $\mathcal{ALC}$, one can suspect that the results on $\mathcal{ALC}$ KBs will be no better than on $\mathcal{EL}$ KBs. Concentrating on atomic subsumptions only effectively means we have ignored neural operators, one of the most interesting parts of the reason-able embeddings. The third assumption means we were only able to assess the incompleteness of the neural reasoner, i.e., its tendency to reject subsumptions that in fact follow from the KB. Due to these assumptions, this is only the first step to properly evaluating the scope of usefulness and applicability of the reason-able embeddings.

Acknowledgments

This research was partially supported by 0311/SBAD/0726 and by TAILOR, a project funded by EU Horizon 2020 research and innovation programme under GA No 952215.

References

[1] S. Choudhary, T. Luthra, A. Mittal, R. Singh, A survey of knowledge graph embedding and their applications, CoRR abs/2107.07842 (2021). URL: https://arxiv.org/abs/2107.07842. arXiv:2107.07842.

[2] Q. Wang, Z. Mao, B. Wang, L. Guo, Knowledge graph embedding: A survey of approaches and applications, IEEE Trans. Knowl. Data Eng. 29 (2017) 2724–2743. URL: https://doi.org/10.1109/TKDE.2017.2754499. doi:10.1109/TKDE.2017.2754499.

[3] Y. Dai, S. Wang, N. N. Xiong, W. Guo, A survey on knowledge graph embedding: Approaches, applications and benchmarks, Electronics 9 (2020) 750. URL: https://doi.org/10.3390/electronics9050750. doi:10.3390/electronics9050750.

[4] H. Cai, V. W. Zheng, K. C. Chang, A comprehensive survey of graph embedding: Problems, techniques, and applications, IEEE Trans. Knowl. Data Eng. 30 (2018) 1616–1637. URL: https://doi.org/10.1109/TKDE.2018.2807452. doi:10.1109/TKDE.2018.2807452.

[5] M. Wang, L. Qiu, X. Wang, A survey on knowledge graph embeddings for link prediction, Symmetry 13 (2021). URL: https://www.mdpi.com/2073-8994/13/3/485. doi:10.3390/sym13030485.

[6] P. Ristoski, J. Rosati, T. D. Noia, R. D. Leone, H. Paulheim, Rdf2vec: RDF graph embeddings and their applications, Semantic Web 10 (2019) 721–752. URL: https://doi.org/10.3233/SW-180317. doi:10.3233/SW-180317.

[7] J. Chen, P. Hu, E. Jiménez-Ruiz, O. M. Holter, D. Antonyrajah, I. Horrocks, Owl2vec*: embedding of OWL ontologies, Mach. Learn. 110 (2021) 1813–1845. URL: https://doi.org/10.1007/s10994-021-05997-6. doi:10.1007/s10994-021-05997-6.

[8] C. d'Amato, N. F. Quatraro, N. Fanizzi, Injecting background knowledge into embedding models for predictive tasks on knowledge graphs, in: R. Verborgh, et al. (Eds.), ESWC 2021, volume 12731 of *Lecture Notes in Computer Science*, Springer, 2021, pp. 441–457. URL: https://doi.org/10.1007/978-3-030-77385-4_26. doi:10.1007/978-3-030-77385-4_26.

[9] B. Makni, J. A. Hendler, Deep learning for noise-tolerant RDFS reasoning, Semantic Web 10 (2019) 823–862. URL: https://doi.org/10.3233/SW-190363. doi:10.3233/SW-190363.

[10] P. Hohenecker, T. Lukasiewicz, Ontology reasoning with deep neural networks, J. Artif. Intell. Res. 68 (2020) 503–540. URL: https://doi.org/10.1613/jair.1.11661. doi:10.1613/jair.1.11661.

[11] F. Bianchi, P. Hitzler, On the capabilities of logic tensor networks for deductive reasoning, in: A. Martin, et al. (Eds.), AAAI-MAKE 2019, volume 2350 of *CEUR Workshop Proc.*, 2019. URL: http://ceur-ws.org/Vol-2350/paper22.pdf.

[12] M. Ebrahimi, M. K. Sarker, F. Bianchi, N. Xie, A. Eberhart, D. Doran, H. Kim, P. Hitzler, Neuro-symbolic deductive reasoning for cross-knowledge graph entailment, in: A. Martin, et al. (Eds.), AAAI-MAKE 2021, volume 2846 of *CEUR Workshop Proc.*, 2021. URL: http://ceur-ws.org/Vol-2846/paper8.pdf.

[13] D. M. Adamski, J. Potoniec, Reason-able embeddings: Learning concept embeddings with a transferable neural reasoner, Semantic Web (2023). URL: https://www.semantic-web-journal.net/content/reason-able-embeddings-learning-concept-embeddings-transferable-neural-reasoner, under review.

[14] F. Baader, S. Brandt, C. Lutz, Pushing the EL envelope, in: L. P. Kaelbling, A. Saffiotti (Eds.), IJCAI-05, Professional Book Center, 2005, pp. 364–369. URL: http://ijcai.org/Proceedings/05/Papers/0372.pdf.

[15] M. Horridge, B. Parsia, From justifications towards proofs for ontology engineering, in: F. Lin, U. Sattler, M. Truszczynski (Eds.), Principles of Knowledge Representation and Reasoning: Proceedings of the Twelfth International Conference, KR 2010, Toronto, Ontario, Canada, May 9-13, 2010, AAAI Press, 2010. URL: http://aaai.org/ocs/index.php/KR/KR2010/paper/view/1263.

[16] J. Potoniec, Finding unexplainable triples in an RDF graph, in: A. Gangemi, A. L. Gentile, A. G. Nuzzolese, S. Rudolph, M. Maleshkova, H. Paulheim, J. Z. Pan, M. Alam (Eds.), The Semantic Web: ESWC 2018 Satellite Events - ESWC 2018 Satellite Events, Heraklion, Crete, Greece, June 3-7, 2018, Revised Selected Papers, volume 11155 of *Lecture Notes in Computer Science*, Springer, 2018, pp. 3–7. URL: https://doi.org/10.1007/978-3-319-98192-5_1. doi:10.1007/978-3-319-98192-5_1.

[17] M. Ebrahimi, A. Eberhart, F. Bianchi, P. Hitzler, Towards bridging the neuro-symbolic gap: deep deductive reasoners, Appl. Intell. 51 (2021) 6326–6348. URL: https://doi.org/10.1007/s10489-020-02165-6. doi:10.1007/s10489-020-02165-6.

[18] J. Demsar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. 7 (2006) 1–30. URL: http://jmlr.org/papers/v7/demsar06a.html.