Improving Knowledge Graph Embeddings with Ontological Reasoning

Paper 259

Abstract. Knowledge graph (KG) embedding models have emerged as powerful means for KG completion. To learn the representation of KGs, entities and relations are projected in a low-dimensional vector space so that not only existing triples in the KG are preserved but also new triples can be predicted. Embedding models might learn a good representation of the input KG, but due to the nature of machine learning approaches, they often lose the semantics of entities and relations, which might lead to nonsensical predictions. To address this issue we propose to improve the accuracy of embeddings using ontological reasoning. More specifically, we present a novel iterative approach *ReasonKGE* that identifies dynamically via symbolic reasoning inconsistent predictions produced by a given embedding model and feeds them as negative samples for retraining this model. In order to address the scalability problem that arises when integrating ontological reasoning into the training process, we propose an advanced technique to generalize the inconsistent predictions to other semantically similar negative samples during retraining. Experimental results demonstrate the improvements in accuracy of facts produced by our method compared to the state-of-the-art.

1 Introduction

Motivation. Knowledge Graphs (KG) describe facts about a certain domain of interest by representing them using entities interconnected via relations. Prominent examples of large KGs are DBPedia [4], Yago [30], and WikiData [33]. KGs are widely used for natural question answering, web search and data analytics. Modern KGs store information about millions of facts, however, since they are typically constructed semi-automatically or using crowd-sourcing methods, KGs are often bound to be incomplete.

To address this issue, knowledge graph embedding methods have been proposed for the *knowledge completion* task, i.e. predicting links between entities. Embedding methods learn the representation of the input KG by projecting entities and relations in a low-dimensional vector space so that not only existing triples in the KG are preserved but also new triples can be predicted (see, e.g., [35] for overview of existing approaches). Typically, the training of KG embedding models aims at discerning between correct (positive) and incorrect (negative) triples. A completion model then associates a score with every input triple. The protocol of evaluating the accuracy of embeddings concerns ranking triples present in the test split, which are considered positive, with respect to their negative counterparts. The goal of the embedding models is to rank every positive triple higher than all its negative alternatives. Therefore, the quality of embedding models is heavily impacted by the way the generation of negative triples is realized. Since KGs store explicitly only positive triples, proper negative triple generation is acknowledged to be a very challenging problem [11,20,39,38].

State-of-the-Art and its limitation. In the majority of existing methods the generation of negative triples is done either completely at random relying on the (local) closed world assumption [26] or by exploiting the KG structure for the generation of likely hard negative samples [1,12,38,39,2].

While powerful, the above methods do not have a systematic way of guaranteeing that the generated negative samples are actually false in reality. In [11] this issue is partially addressed by taking as negative examples only those among corrupted triples that are inconsistent with the KG and its ontology describing the KG schema. However, the generation of all possible such inconsistent triples as negative samples is clearly infeasible in practice, and hence some important inconsistent triples might be missing in the obtained set. Subsequently, the embedding model might predict inconsistent triples, and no controlled way to avoid that exists [36]. This is clearly undesired, and ideally predictions produced by the resulting embedding should comply with the ontology [36].

Approach and Contributions. Thus, to address this shortcoming, in this work we propose to advance the ideas from [11,36] by developing an iterative method that proceeds as follows. We first start with any available negative sampling procedure (e.g., [20,39]) and train the embedding model as usual. Then, among predictions made by the model, we select those that cause inconsistency when being added to the KG, as negative samples for the next iteration of our method. To avoid predicting similar wrong triples, along with the inconsistent facts explicitly inferred by the embedding model, we also generate facts that are semantically similar via the generalization procedure inspired by abstraction [31] originally proposed for dealing with large-scale reasoning tasks. The ontologies that we consider are expressed in the Description Logic (DL) $DL\text{-}Lite$ [3] extended with transitive roles and concept disjunctions (we refer to this DL as $DL\text{-}Lite^{\mathcal{S}\sqcup}$). During the consistency check we exploit the locality property of the target DL, based on which it is sufficient to consider only a small subset of relevant facts when verifying consistency. Our method can support any embedding model, and with the increasing number of iterations it yields better embeddings that make less inconsistent predictions.

The salient contributions of our work can be summarized as follows.

- We introduce the *ReasonKGE* framework for exploiting ontological reasoning to improve existing embedding models by advancing their negative sampling.
- To efficiently filter inconsistent embedding-based predictions, we exploit the locality property of light-weight ontologies. Moreover, in the spirit of [31] we generalize the computed inconsistent facts to a set of other similar ones to be fed back to the embedding model as negative samples.
- The evaluation of the proposed method on a set of state-of-the-art KGs equipped with ontologies, demonstrates that ontological reasoning exploited in the suggested way indeed improves the existing embedding models with respect to the quality of fact prediction.

Organization. The rest of the paper is structured as follows. In Sec. 2 we present necessary background on KGs, ontologies and embedding models. In Sec. 3 our approach is described in details, and then in Sec. 4 the results of our empirical evaluation are discussed. Finally, in Sec. 5 we present the related work, and conclude in Sec. 6. An ex-

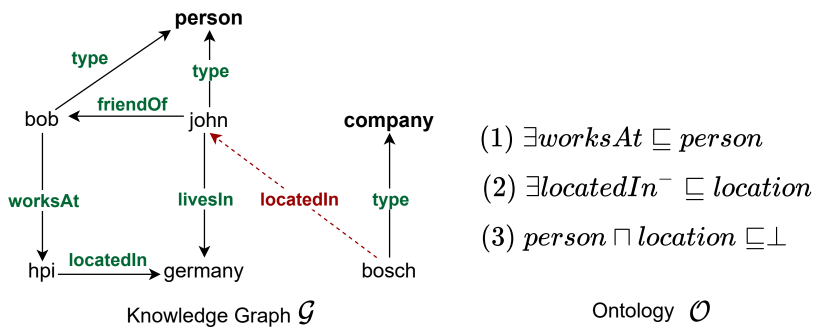

Fig. 1: Example knowledge graph with its ontology, where solid links correspond to the true facts, while the dashed one to a spurious predicted fact.

tended version of this work with additional experimental details can be accessed using the external repository[1].

2 Preliminaries

We assume countable pairwise disjoint sets $\mathsf{N_C}$, $\mathsf{N_P}$ and $\mathsf{N_I}$ of class names (*a.k.a.* types), property names (*a.k.a.* relations), and individuals (*a.k.a.* entities). We also assume the standard relation $rdf{:}type$ (abbreviated as $type$) to be included in $\mathsf{N_P}$. A *knowledge graph* (KG) $\mathcal{G}$ is a finite set of *triples* of the form $\langle s, p, o\rangle$, where $s \in \mathsf{N_I}, p \in \mathsf{N_P}, o \in \mathsf{N_I}$, if $p \neq type$, and $o \in \mathsf{N_C}$ otherwise. KGs typically follow Open World Assumption (OWA), meaning that they store only a fraction of positive facts. For instance, given the KG from Fig. 1 $\langle john, type, person\rangle$ and $\langle john, livesIn, germany\rangle$ are true KG facts; however, whether $\langle john, worksAt, bosch\rangle$ holds or not is unknown. Given a triple α, we denote by $\mathsf{Ent}(\alpha)$ a set of all entities occurring in α and extend this notation to a set of triples as $\mathsf{Ent}(\mathcal{G}) = \bigcup_{\alpha\in\mathcal{G}} \mathsf{Ent}(\alpha)$.

An ontology $\mathcal{O}$ (*a.k.a.* TBox) is a set of axioms expressed in a certain Description Logic (DL) [5]. In this work we focus on $\mathit{DL\text{-}Lite}^{\mathcal{S}\sqcup}$, i.e., extension of $\mathit{DL\text{-}Lite}$ [3] with transitive roles and concept disjunctions. Classes C denoting sets of entities, and roles R denoting binary relations between entities, obey the following syntax:

$$C ::= A \mid \exists R \mid A \sqcup B \mid A \sqcap B \mid \neg C$$
$$R ::= P \mid P^{-}$$

Here, $A, B \in \mathsf{N_C}$ are atomic classes and $P \in \mathsf{N_P}$ is an atomic property (i.e., binary relation). An ontology $\mathcal{O}$ is a finite set of axioms of the form $C_1 \sqsubseteq C_2$, $R_1 \sqsubseteq R_2$, $R \circ R \sqsubseteq R$, reflecting the transitivity of the relation R. The summary of the DL syntax in $\mathit{DL\text{-}Lite}^{\mathcal{S}\sqcup}$ and its translation to OWL 2[2] is presented in Table 1. In the rest of the paper, we assume that all ontologies in this work are expressed in $\mathit{DL\text{-}Lite}^{\mathcal{S}\sqcup}$.

[1] https://tiny.one/tp4vark
[2] https://www.w3.org/TR/owl2-overview/

DL Syntax	OWL Syntax	Semantics
R	R	$R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
R^{-}	`ObjectInverseOf`(R)	$\{\langle e, d\rangle \mid \langle d, e\rangle \in R^{\mathcal{I}}\}$
A	A	$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
$\top$	`owl:Thing`	$\Delta^{\mathcal{I}}$
$\bot$	`owl:NoThing`	$\emptyset$
$\neg C$	`ObjectComplementOf`(C)	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
$C \sqcap D$	`ObjectIntersectionOf`(C, D)	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
$C \sqcup D$	`ObjectUnionOf`(C, D)	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
$\exists P$	`ObjectSomeValuesFrom`$(P,$ `owl:Thing`$)$	$\{d \mid \exists e \in \Delta^{\mathcal{I}}: \langle d, e\rangle \in P^{\mathcal{I}}\}$
$C \sqsubseteq D$	`SubClassOf`(C, D)	$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
$P \sqsubseteq S$	`SubObjectPropertyOf`(P, S)	$P^{\mathcal{I}} \subseteq S^{\mathcal{I}}$
$P \circ P \sqsubseteq P$	`TransitiveObjectProperty`(P)	$P^{\mathcal{I}} \circ P^{\mathcal{I}} \subseteq P^{\mathcal{I}}$
$\langle a, type, c\rangle$	`ClassAssertion`(C, a)	$a^{\mathcal{I}} \in C^{\mathcal{I}}$
$\langle a, p, b\rangle$	`ObjectPropertyAssertion`(P, a, b)	$\langle a^{\mathcal{I}}, b^{\mathcal{I}}\rangle \in P^{\mathcal{I}}$

Table 1: Syntax and semantics of the ontology language considered in this paper where A, R are a class name and property name, respectively; C and D are class expressions, P, S are property expressions, and a, b are entities.

Our running example of a KG with an ontology given in Figure 1 reflects the domain knowledge about people and their working places. The ontology states that (1) the domain of $worksAt$ relation is $person$, (2) the range of $locatedIn$ is $location$, and (3) $person$ is disjoint with $location$.

Inconsistency and Explanations. The semantics of knowledge graphs and ontologies is defined using the direct model-theoretic semantics via interpretations [25]. An *interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ consists of a non-empty set $\Delta^{\mathcal{I}}$, the *domain* of $\mathcal{I}$, and an *interpretation function* $\cdot^{\mathcal{I}}$, that assigns to each $A \in \mathsf{N_C}$ a subset $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, to each $R \in \mathsf{N_R}$ a binary relation $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, and to each $a \in \mathsf{N_I}$ an element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$. This assignment is extended to (complex) classes and roles as shown in Table 1.

An interpretation $\mathcal{I}$ *satisfies* an axiom α (written $\mathcal{I} \models \alpha$) if the corresponding condition in Table 1 holds. Given a KG $\mathcal{G}$ and an ontology $\mathcal{O}$, $\mathcal{I}$ is a *model* of $\mathcal{G} \cup \mathcal{O}$ (written $\mathcal{I} \models \mathcal{G} \cup \mathcal{O}$) if $\mathcal{I} \models \alpha$ for all axioms $\alpha \in \mathcal{G} \cup \mathcal{O}$. We say that $\mathcal{G} \cup \mathcal{O}$ *entails* an axiom α (written $\mathcal{G} \cup \mathcal{O} \models \alpha$), if every model of $\mathcal{G} \cup \mathcal{O}$ satisfies α. A KG $\mathcal{G}$ is *inconsistent* w.r.t. an ontology $\mathcal{O}$ if no model for $\mathcal{G} \cup \mathcal{O}$ exists. In this case, $\mathcal{G} \cup \mathcal{O}$ is inconsistent. Intuitively, $\mathcal{G} \cup \mathcal{O}$ is inconsistent when some facts of $\mathcal{G}$ contradict some axioms of $\mathcal{O}$.

Under the considered ontology language, KG inconsistency has a locality property, i.e., the problem of checking inconsistency for a KG (w.r.t. an ontology $\mathcal{O}$) can be reduced to checking inconsistency for separated KG *modules* (w.r.t. $\mathcal{O}$) [31].

Definition 1 (Modules). *Given a KG $\mathcal{G}$ and an entity $e \in \mathsf{Ent}(\mathcal{G})$, the* module *of e w.r.t. $\mathcal{G}$ is defined as $\mathcal{M}(e, \mathcal{G}) = \{\alpha \mid \alpha \in \mathcal{G}$ and e occurs in $\alpha\}$. We denote the set of all modules for individuals occurring in $\mathcal{G}$ as $\mathcal{M}_{\mathcal{G}} = \{\mathcal{M}(e, \mathcal{G}) \mid e \in \mathsf{Ent}(\mathcal{G})\}$.*

Lemma 1 (Consistency Local Property). *Let $\mathcal{G}$ be a KG and $\mathcal{O}$ an ontology. Then $\mathcal{G} \cup \mathcal{O}$ is consistent iff $\mathcal{M}(a, \mathcal{G}) \cup \mathcal{O}$ is consistent for every $a \in \mathsf{Ent}(\mathcal{G})$.*

An *explanation* for inconsistency of $\mathcal{G} \cup \mathcal{O}$ [19], denoted by $\mathcal{E} = \mathcal{E}_\mathcal{G} \cup \mathcal{E}_\mathcal{O}$ with $\mathcal{E}_\mathcal{G} \subseteq \mathcal{G}$ and $\mathcal{E}_\mathcal{O} \subseteq \mathcal{O}$, is a (subset-inclusion) smallest inconsistent subset of $\mathcal{G} \cup \mathcal{O}$.

Example 1. The KG from Fig. 1 with all facts including the dashed red one is inconsistent with the ontology $\mathcal{O}$, and a possible explanation for that is $\mathcal{E} = \mathcal{E}_\mathcal{G} \cup \mathcal{E}_\mathcal{O}$ with $\mathcal{E}_\mathcal{G} = \{\langle \mathit{bosch}, \mathit{locatedIn}, \mathit{john}\rangle, \langle \mathit{john}, \mathit{type}, \mathit{person}\rangle\}$ and $\mathcal{E}_\mathcal{O} = \{\exists \mathit{locatedIn}^- \sqsubseteq \mathit{location}$, $\mathit{person} \sqcap \mathit{location} \sqsubseteq \bot\}$.

KG Embeddings. KG embeddings aim at representing all entities and relations in a continuous vector space, usually as vectors or matrices called *embeddings*. Embeddings can be used to estimate the likelihood of a triple to be true via a scoring function: $f : \mathsf{N_I} \times \mathsf{N_P} \times \mathsf{N_I} \to \mathbb{R}$. Concrete scoring functions are defined based on various vector space assumptions. The likelihood that the respective assumptions of the embedding methods hold, should be higher for triples in the KG than for negative samples outside the KG. The learning process is done through minimizing the error induced from the assumptions given by their respective loss functions. Below we describe the most prominent assumptions for KG embeddings:

(i) The translation-based assumption, *e.g.*, TransE [9] embeds entities and relations as vectors and assumes $\mathbf{v_s} + \mathbf{v_p} \approx \mathbf{v_o}$ for true triples, where $\mathbf{v_s}, \mathbf{v_p}, \mathbf{v_o}$ are vector embeddings for subject s, predicate p and object o, respectively. The models that rely on the translation assumption are generally optimised by minimizing the following ranking-based loss function

$$\sum_{\langle s_i,p_i,o_i\rangle \in S^+} \sum_{\langle s'_i,p_i,o'_i\rangle \in S^-} [\gamma - f(s_i,p_i,o_i) + f(s'_i,p_i,o'_i)]_+ \tag{1}$$

where $f(s,p,o) = -\left\|\mathbf{v_s} + \mathbf{v_p} - \mathbf{v_o}\right\|_1$, S^+ and S^- correspond to the sets of positive and negative training triples respectively, that are typically disjoint.

(ii) The linear map assumption, *e.g.*, ComplEx [32] embeds entities as vectors and relations as matrices. It assumes that for true triples, the linear mapping $\mathbf{M_p}$ of the subject embedding $\mathbf{v_s}$ is close to the object embedding $\mathbf{v_o}$: $\mathbf{v_s}\mathbf{M_p} \approx \mathbf{v_o}$. The loss function used for training the linear-map embedding models is given as follows:

$$\sum_{\langle s_i,p_i,o_i\rangle \in S^+} \sum_{\langle s'_i,p_i,o'_i\rangle \in S^-} l(1, f(s_i,p_i,o_i)) + l(-1, f(s'_i,p_i,o'_i))) \tag{2}$$

where $f(s,p,o) = \mathbf{v}_s\mathbf{M}_p\mathbf{v}_o$ and $l(\alpha,\beta) = log(1 - exp(-\alpha\beta))$.

3 Ontological Reasoning for Iterative Negative Sampling

While a variety of embedding models exist in the literature [35], one of the major challenges for them to perform accurate fact predictions is finding an effective way for generation of relevant negative samples [28,11,34]. Commonly used approaches for negative sampling randomly corrupt existing triples by perturbing their subject, predicate or

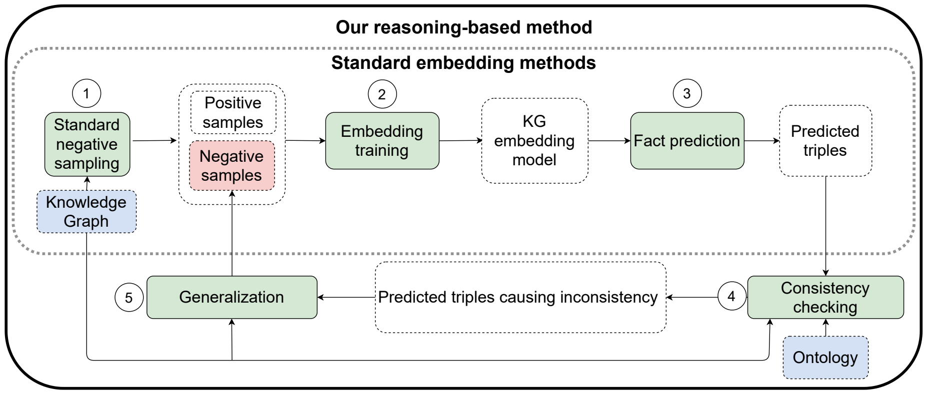

Fig. 2: Standard embedding pipeline (grey dotted frame) and our reasoning-based method (black frame) in a nutshell

object [9,29,13] or rely on the (local) closed world assumption (LCWA). Based on CWA all triples not present in the KG are assumed to be false, while LCWA is a variation of CWA, in which for every $\langle s, p, o\rangle$, only facts of the form $\langle s, p, o'\rangle \notin \mathcal{G}$ are assumed to be false. For instance, given the facts in Figure 1, the corrupted negative triples obtained based on the LCWA could be $\langle john, livesIn, hpi\rangle$ or $\langle bob, worksAt, bosch\rangle$.

However, since KGs follow OWA, the standard sampling methods might often turn out to be sub-optimal, resulting in false positive negative samples [11]. For example, the corrupted triple $\langle bob, worksAt, bosch\rangle$ from above might actually be true in reality.

A natural method to avoid false positives and generate only relevant negative samples is by relying on ontologies with which KGs are typically equipped. A naive approach for that is to generate all facts that can be formed using relations and entities in $\mathcal{G}$ (i.e., construct the Herbrand base) and check which among the resulting candidates are inconsistent with $\mathcal{G} \cup \mathcal{O}$. As modern KGs store millions of facts, the described procedure is clearly infeasible in practice. To still sample some inconsistent triples, in [11] facts $p(s, o) \in \mathcal{G}$ are corrupted by substituting s (resp. o) with s' (resp. o') such that s and s' (resp. o and o') belong to disjoint classes and the resulting corrupted triple is inconsistent. For example, given $\mathcal{G}$ and $\mathcal{O}$ in Fig 1, from $\langle bob, worksAt, germany\rangle$ we can obtain $\alpha_1 = \langle germany, worksAt, germany\rangle$ or $\alpha_2 = \langle bob, worksAt, john\rangle$, as $person$ is disjoint with $location$. However, this method might fail to avoid the inconsistent triples that the model actually predicts. Indeed, for instance, the triple $\langle bosch, locatedIn, john\rangle$ will not be generated by this method as a negative example, and the model can in principle still predict it.

Therefore, instead of pre–computing a static set of negative examples, we propose to iteratively generate and improve this set (and subsequently also the embedding model) dynamically by computing a collection of negative samples in a guided fashion from embedding model based on its predictions that are inconsistent with the ontology. On the one hand, this intuitively allows us to overcome the computational challenge of generating all possible negative examples at once, but rather add the most relevant ones on demand to the embedding training process. On the other hand, this approach is capable of reducing frequently encountered errors (in terms of inconsistent predictions) for particularly difficult triples by directly incorporating feedback from incorrect predictions back to the model for further training. Indeed, when trained for increasing number of iterations, such method is capable of generating embeddings that predict fewer inconsistent facts, as empirically demonstrated in Section 4.

3.1 Approach Overview

Next we describe in more details the proposed framework referred to as *ReasonKGE*, whose main steps are depicted in Figure 2. Given a KG, ontology and an embedding method, we aim at generating an enhanced KG embedding, which is trained for predicting facts that are consistent with the KG and the ontology at hand.

The input to our method (represented by blue dashed boxes) is the KG and the ontology, while the output (the red dashed box) is the set of negative samples that is incorporated during the iterative training and tuning of a KG embedding model in each iteration. As negative samples are obtained based on predictions made by an existing embedding, a baseline model is required in the first iteration. For this, in step (1) we obtain the negative samples with **standard negative sampling** using any of the existing methods [11,9,29,13]. We then perform **embedding training** in step (2) to construct the initial KG embedding model.

This model is used for obtaining predictions and computing the set of negative samples for the next training iteration. Specifically, in step (3) the model is used for **fact prediction** as follows. For every triple in the training set, given its subject s and predicate p, we retrieve the top ranked object and obtain the fact $\langle s, p, o\rangle$ as the respective prediction. The same is done inversely for computing the top ranked subject given the object o and predicate p in the training set. Note that only triples that are not in the training set are considered as predictions. In step (4) we check whether the predicted triple complies with the ontology relying on the **consistency checking** procedure. In case the respective triple is found to be inconsistent, in step (5) we generalize it to other semantically similar triples using the **generalization** procedure to obtain an extended set of negative samples. Finally, the computed negative samples, both for subject and object predictions are fed back as input to the next iteration of the embedding training process. The detailed steps are presented in Algorithm 1 and explained in what follows.

3.2 Consistency Checking

The goal of the consistency checking procedure is to verify which predictions made by the embedding model in step (3) are inconsistent with the ontology $\mathcal{O}$ and the original KG $\mathcal{G}$. In principle, any reasoner capable of performing consistency checking effectively for ontologies in the considered $DL\text{-}Lite^{\mathcal{S}\sqcup}$ language can be used in this step. As the task that we consider concerns verifying whether a particular triple causes inconsistency, for the target DL when performing the consistency check one does not need to account for the whole KG, but only a small subset of relevant facts. To this end, we define the *relevant sets* as follows.

Definition 2 (Relevant set). *Let $\mathcal{G}$ be a KG and α be a triple. The* relevant set $\mathsf{Relv}(\alpha, \mathcal{G})$ *of α w.r.t. $\mathcal{G}$ is defined as* $\mathsf{Relv}(\alpha, \mathcal{G}) = \{\alpha\} \cup \{\beta \in \mathcal{G} \,|\mathsf{Ent}(\beta) \cap \mathsf{Ent}(\alpha) \neq \emptyset\}$.

Algorithm 1: Training embedding models with negative samples using ontological reasoning

Input : Baseline embedding model $\mathbf{E}$, a knowledge graph $\mathcal{G}$, and an ontology $\mathcal{O}$

/* Step 1 and Step 2 */

Train the baseline embedding model $\mathbf{E}$ for a certain number of epochs.

/* Retrain the baseline model with negative samples derived from reasoning */

Loop

  /* Step 3 */

  foreach *triple* $\alpha = \langle s, p, o \rangle \in \mathcal{G}$ **do**

    Get a set $\mathsf{Predictions}(\alpha)$ of predicted triples of the form $\langle s, p, \hat{o} \rangle$ and $\langle \hat{s}, p, o \rangle$ by giving $\langle s, p \rangle$ and $\langle p, o \rangle$ as inputs to $\mathbf{E}$ and obtaining predicted entities $\hat{o}$ and $\hat{s}$, respectively.

    /* Step 4 */

    $\mathsf{NegSamples}(\alpha) \leftarrow \emptyset$

    foreach *predicted triple* $\beta \in \mathsf{Predictions}(\alpha)$ **do**

      Compute the relevant set $\mathsf{Relv}(\beta, \mathcal{G})$ of β w.r.t. $\mathcal{G}$.

      if $\mathsf{Relv}(\beta, \mathcal{G}) \cup \mathcal{O}$ *is inconsistent* **then**

        /* Step 5 */

        Compute explanations for inconsistency.

        foreach *inconsistency explanation* $\mathcal{E}_{\mathcal{G}} \cup \mathcal{E}_{\mathcal{O}}$ **do**

          Compute $\mathsf{GeneralizedSamples}(\beta)$ as defined in Definition 4.

          $\mathsf{NegSamples}(\alpha) \leftarrow \mathsf{NegSamples}(\alpha) \cup \mathsf{GeneralizedSamples}(\beta)$

  Retrain $\mathbf{E}$ in which, for each training step that considers $\alpha \in \mathcal{G}$, $\mathsf{NegSamples}(\alpha)$ is used as negative samples in the loss function, e.g. Equation 1 or Equation 2.

Example 2. For $\alpha = \langle bosch, locatedIn, john \rangle$ and $\mathcal{G}$ in Fig. 1, we have the following relevant set $\mathsf{Relv}(\alpha, \mathcal{G}) = \{\alpha\} \cup \{\langle john, livesIn, germany \rangle, \langle john, friendOf, bob \rangle, \langle john, type, person \rangle, \langle bosch, type, company \rangle\}$.

The following proposition allows us to reduce the consistency checking of $\alpha \cup \mathcal{G} \cup \mathcal{O}$ to the consistency checking of $\mathsf{Relv}(\alpha, \mathcal{G}) \cup \mathcal{O}$.

Proposition 1. *Let $\mathcal{G}$ be a knowledge graph, $\mathcal{O}$ an ontology such that $\mathcal{G} \cup \mathcal{O}$ is consistent, and α a triple. Then, $\alpha \cup \mathcal{G} \cup \mathcal{O}$ is consistent iff $\mathsf{Relv}(\alpha, \mathcal{G}) \cup \mathcal{O}$ is consistent.*

Proof. Since $\mathsf{Relv}(\alpha, \mathcal{G}) \subseteq \mathcal{G}$, we have $\alpha \cup \mathcal{G} \cup \mathcal{O}$ being consistent implies that $\mathsf{Relv}(\alpha, \mathcal{G}) \cup \mathcal{O}$ is also consistent. We start showing the remaining direction by assuming that $\mathsf{Relv}(\alpha, \mathcal{G}) \cup \mathcal{O}$ is consistent and then show that $\alpha \cup \mathcal{G} \cup \mathcal{O}$ is also consistent. Let $\alpha = \langle s, p, o \rangle$, by Definition 2, we have $\mathsf{Relv}(\alpha, \mathcal{G}) = \mathcal{M}(s, \alpha \cup \mathcal{G}) \cup \mathcal{M}(o, \alpha \cup \mathcal{G})$. Since $\mathcal{G} \cup \mathcal{O}$ is consistent, by Lemma 1, we have $\mathcal{M}(e, \mathcal{G}) \cup \mathcal{O}$ is consistent for every entity in $\mathsf{Ent}(\mathcal{G}) \setminus \{s, o\}$. Since $e \notin \{s, o\}$, we have $\mathcal{M}(e, \mathcal{G}) = \mathcal{M}(e, \alpha \cup \mathcal{G})$, which implies $\mathcal{M}(e, \alpha \cup \mathcal{G}) \cup \mathcal{O}$ is consistent ($\star$). From the assumption that $\mathsf{Relv}(\alpha, \mathcal{G}) \cup \mathcal{O}$ is consistent and $\mathsf{Relv}(\alpha, \mathcal{G}) = \mathcal{M}(s, \alpha \cup \mathcal{G}) \cup \mathcal{M}(o, \alpha \cup \mathcal{G})$, we obtain $\mathcal{M}(s, \alpha \cup \mathcal{G})$ and $\mathcal{M}(o, \alpha \cup \mathcal{G})$ are consistent w.r.t. $\mathcal{O}$ (†). From ($\star$) and (†) we have $\alpha \cup \mathcal{G} \cup \mathcal{O}$ is consistent using Lemma 1. $\square$

Relying on Proposition 1, it is sufficient to check the consistency of a triple α with respect to $\mathcal{G} \cup \mathcal{O}$ using $\mathsf{Relv}(\alpha, \mathcal{G})$ rather than the whole KG. We make use of this property in step (4), and for every prediction produced by the embedding model, we first construct the relevant set for the respective prediction, and then perform the consistency check relying only on the corresponding relevant sets.

Example 3. Assume that the fact $\alpha = \langle \mathit{bosch}, \mathit{locatedIn}, \mathit{john}\rangle$ has been predicted by the embedding model in step (3). Then in the consistency checking step (4) we first construct the relevant set for α as $\mathsf{Relv}(\alpha, \mathcal{G})$ given in Example 2 and check the consistency of $\mathsf{Relv}(\alpha, \mathcal{G}) \cup \mathcal{O}$. Clearly, we have $\mathsf{Relv}(\alpha, \mathcal{G}) \cup \mathcal{O}= \{\langle \mathit{bosch}, \mathit{locatedIn}, \mathit{john}\rangle\} \cup \{\langle \mathit{john}, \mathit{livesIn}, \mathit{germany}\rangle, \langle \mathit{john}, \mathit{type}, \mathit{person}\rangle, \langle \mathit{john}, \mathit{friendOf}, \mathit{bob}\rangle, \langle \mathit{bosch}, \mathit{type}, \mathit{company}\rangle\} \cup \mathcal{O}$ is inconsistent, since $\langle \mathit{bosch}, \mathit{locatedIn}, \mathit{john}\rangle$ and $\{\exists \mathit{locatedIn}^- \sqsubseteq \mathit{location}\} \in \mathcal{O}$ imply that $\langle \mathit{john}, \mathit{type}, \mathit{location}\rangle$, which contradicts the fact that $\langle \mathit{john}, \mathit{type}, \mathit{person}\rangle \in \mathcal{G}$ and $\mathit{person} \sqcap \mathit{location} \sqsubseteq \bot \in \mathcal{O}$. Thus, we have that $\alpha \cup \mathcal{G} \cup \mathcal{O}$ is inconsistent by monotonicity. Proposition 1 further guarantees that it is sufficient to check the consistency of $\alpha \cup \mathcal{G} \cup \mathcal{O}$ this way.

3.3 Negative Sample Generalization

Given each triple in the training step, one needs to sample not a single corrupted triple but a set of such triples to train the embedding model at hand. Therefore, in addition to checking whether each prediction causes inconsistency, we generalize inconsistent predictions to obtain sufficient number of negative samples for retraining the embedding model. We show that the triples obtained by generalizing each inconsistent prediction in our step 5 are also inconsistent w.r.t. the input KG and the accompanying ontology.

A naive approach to obtain the generalized triples of an inconsistent predicted triple, e.g. $\langle s, p, \hat{o}\rangle$, is to replace $\hat{o}$ by another entity o in the input KG such that o has similar KG neighborhood as $\hat{o}$. However, it might happen that only a subset of triples containing $\hat{o}$ is inconsistent w.r.t. the ontology. Therefore, it is sufficient to find such o that it has similar triples as in that subset. This will increase the number of generalized triples as demonstrated in Example 4. To compute a subset of triples of $\hat{o}$ that is inconsistent w.r.t. the ontology, we compute explanations for the inconsistency of $\mathsf{Relv}(\langle s, p, \hat{o}\rangle, \mathcal{G}) \cup \mathcal{O}$.

Example 4. Consider the KG $\mathcal{G}$ and ontology $\mathcal{O}$ as in Figure 1. Assume that $\alpha = \langle \mathit{bosch}, \mathit{locatedIn}, \mathit{john}\rangle$ is the predicted triple, i.e., the embedding model predicted $john$ as the object entity for the given subject $bosch$ and relation $locatedIn$. The explanation for inconsistency of $\mathsf{Relv}(\alpha, \mathcal{G}) \cup \mathcal{O}$ is $\mathcal{E} = \mathcal{E}_{\mathcal{G}} \cup \mathcal{E}_{\mathcal{O}}$, for which it holds that $\mathcal{E}_{\mathcal{G}} = \{\langle bosch, locatedIn, john\rangle, \langle john, type, person\rangle\}$ and $\mathcal{E}_{\mathcal{O}} = \{\exists \mathit{located}^- \sqsubseteq \mathit{location}, \mathit{person} \sqcap \mathit{location} \sqsubseteq \bot\}$. Note that there is no other entity in $\mathcal{G}$ that has similar triples as those for john. However, if we restrict to the triples in the explanation for inconsistency of $\mathsf{Relv}(\alpha, \mathcal{G}) \cup \mathcal{O}$, then bob has the same neighborhood triple $\langle \mathit{bob}, \mathit{type}, \mathit{person}\rangle$ as $john$ (the predicted triple is ignored). Therefore, we can take $\langle \mathit{bosch}, \mathit{locatedIn}, \mathit{bob}\rangle$ as another negative sample, which together with $\mathcal{G}$ is clearly inconsistent w.r.t. $\mathcal{O}$.

To formally obtain generalized triples as in Example 4, we rely on the notion of *local type* of an entity [15,16,31] as follows.

Definition 3 (Local Types). *Let* $\mathbf{T}$ *be a set of triples and* e *an entity occurring in* $\mathbf{T}$*. Then, the local type of* e *w.r.t.* $\mathbf{T}$*, written as* $\tau(e, \mathbf{T})$ *or* $\tau(e)$ *when* $\mathbf{T}$ *is clear from the context, is defined as a tuple* $\tau(e) = \langle \tau_i(e), \tau_c(e), \tau_o(e) \rangle$*, where* $\tau_i(e) = \{p \mid \langle s, p, e \rangle \in \mathcal{G}\}$*,* $\tau_c(e) = \{t \mid \langle e, type, t \rangle \in \mathcal{G}\}$*, and* $\tau_o(e) = \{p' \mid \langle e, p', o \rangle \in \mathcal{G}\}$*.The local type* $t = \langle t_i, t_c, t_o \rangle$ *is* smaller than or equal to *the local type* $t' = \langle t'_i, t'_c, t'_o \rangle$*, written as* $t \preceq t'$*, iff* $t_i \subseteq t'_i$*,* $t_c \subseteq t'_c$*, and* $t_o \subseteq t'_o$*.*

Intuitively, a local type of an entity represents a set of types (τ_c) as well as the incoming relations (τ_i) and outgoing relations (τ_o) for that entity in a set of triples.

Example 5 (Example 4 continued). For bob in Fig. 1, we have the local type of bob w.r.t. $\mathcal{G}$ being $\tau(bob) = \langle \{friendOf\}, \{person\}, \{worksAt\} \rangle$. The local type of $john$ w.r.t. $\mathcal{E}_\mathcal{G} \setminus \alpha$ is $\tau(john) = \langle \emptyset, \{person\}, \emptyset \rangle$ and it holds that $\tau(john) \preceq \tau(bob)$.

We now define the set of generalized samples of a given inconsistent predicted triple.

Definition 4 (Generalized Samples). *Let* $\mathcal{G}$ *be a KG,* $\mathcal{O}$ *an ontology, and* $\alpha = \langle s, p, \hat{o} \rangle$ *be a triple in which* $\hat{o}$ *is predicted by an embedding model given the subject entity* s *and relation* p*. Furthermore, let* $\mathcal{E} = \mathcal{E}_\mathcal{G} \cup \mathcal{E}_\mathcal{O}$ *be an inconsistency explanation of* $\mathsf{Relv}(\alpha, \mathcal{G}) \cup \mathcal{O}$*. Then, the set of* generalized samples *of* α *(w.r.t.* $\hat{o}$*,* $\mathcal{E}$*, and* $\mathcal{G}$*) is defined as* $\mathsf{GeneralizedSamples}(\alpha, \hat{o}) = \{\langle s, p, o \rangle \mid \tau(\hat{o}, \mathcal{E}_\mathcal{G} \setminus \alpha) \preceq \tau(o, \mathcal{G})\}$*. The generalized samples* $\mathsf{GeneralizedSamples}(\beta, \hat{s})$ *of* $\beta = \langle \hat{s}, p, o \rangle$*, in which* $\hat{s}$ *is predicted by an embedding model, is defined analogously. When it is clear from the context, we often write* $\mathsf{GeneralizedSamples}(\alpha)$ *without mentioning the corresponding entity.*

Example 6 (Example 5 continued). According to Definition 4 and the local types of $john$ and bob computed in Example 5, we have $\mathsf{GeneralizedSamples}(\alpha) = \{\alpha\} \cup \{\langle bosch, LocatedIn, bob \rangle\}$.

The following Lemma guagrantees that if a triple is inconsistent (together with the input KG) w.r.t. an ontology $\mathcal{O}$ then all generalized triples of that triple are also inconsistent.

Lemma 2. *Let* $\mathcal{G}$ *be a KG,* $\mathcal{O}$ *an ontology,* α *a triple such that* $\mathsf{Relv}(\alpha, \mathcal{G}) \cup \mathcal{O}$ *is inconsistent with an explanation* $\mathcal{E} = \mathcal{E}_\mathcal{G} \cup \mathcal{E}_\mathcal{O}$*, and* $\mathsf{GeneralizedSamples}(\alpha)$ *is the set of generalized triples of* α *w.r.t.* $\mathcal{E}$*,* $\mathcal{G}$*, and some entity occurring in* α*. Then, we have* $\mathsf{Relv}(\beta, \mathcal{G}) \cup \mathcal{O}$ *is inconsistent for every* $\beta \in \mathsf{GeneralizedSamples}(\alpha)$*.*

Proof (Sketch). W.l.o.g. let $\alpha = \langle s, p, \hat{o} \rangle$, $\mathsf{GeneralizedSamples}(\alpha)$ is w.r.t. $\hat{o}$, and $\beta = \langle s, p, o \rangle$. Using the result in [31], one can show that if $\langle s, p, \hat{o} \rangle \in \mathcal{E}_\mathcal{G}$ then $\mathcal{E}_\mathcal{G}$ does not contain $\langle s', p, o \rangle$, where $s \neq s'$ due to the minimality of explanations. Together with the condition $\tau(\hat{o}) \preceq \tau(o)$, we can construct a homomorphism from $\mathsf{Relv}(\alpha, \mathcal{G})$ to $\mathsf{Relv}(\beta, \mathcal{G})$, which implies that $\mathsf{Relv}(\beta, \mathcal{G}) \cup \mathcal{O}$ is inconsistent. □

We proceed to describe the details of step (5). For each predicted triple that is inconsistent w.r.t. the input KG and the accompanied ontology, we compute explanations for inconsistency and for each such explanation, we obtain the generalized triples using Definition 4. These generalized triples are then used as negative samples to retrain the embedding model.

Table 2: Knowledge graph statistics.

	LUBM3U	Yago3-10	DBpedia15K
# Entities	127,645	123,182	12,842
# Predicates	28	37	279
# Training Facts	621,516	1,079,040	69,320
# Validation Facts	77,689	5,000	9,902
# Test Facts	77,689	5,000	19,805

4 Experiments

We have implemented the proposed method in a prototype system *ReasonKGE*[3], and evaluated its performance on the commonly used datasets enriched with ontologies. In this section, we present the results of the evaluation in terms of the impact of our method on the quality of fact predictions compared to the baselines.

4.1 Experimental Setup

Datasets. Among commonly used dataset for evaluating embedding models, we chose the following datasets as they are equipped with their respective ontologies.

– **LUBM3U:** A synthesized dataset derived from the Lehigh University Benchmark [17]. The ontology describing the university domain contains 325 axioms. The respective KG stores data for 3 universities.
– **Yago3-10:** A subset of the widely used Yago dataset. We use the ontology with 4551 axioms introduced in [30] based on Yago schema and class hierarchy.
– **DBpedia15K:** A subset of DBpedia KG proposed in [23]. We exploit the general DBpedia ontology enriched with axioms reflecting the disjointness of classes. The ontology comprises of 3006 axioms.

The statistics of KGs of the respective datasets is presented in Table 2.

Embedding models. To demonstrate the benefits of the proposed iterative ontology-driven negative sampling, we apply our method over the following widely used embeddings: ComplEx [32] and TransE [9]. These models have been selected as prominent examples of translation-based and linear-map embeddings. The LibKGE [28] framework has been used for their tuning.

Measures. We evaluate the performance of the embedding models in terms of the traditional metrics i.e ***MRR*** and ***Hits@k*** in the filtered setting [9]. In addition, we also compute the proportion of inconsistent facts (***Inc@K***) ranked in the top K predictions produced by the presented methods. The measure ***Inc@K*** intuitively reflects how well the model is capable of avoiding inconsistent predictions (the lower the better).

System configuration. In the experiments, we used HermiT [14] as the reasoner and the explanation method in [19] to compute inconsistency explanations. We run *ReasonKGE* for multiple iterations. In every iteration, the model is trained for $n = 100$

[3] The link to the code will become available in case of acceptance.

Table 3: Link prediction results

Model	KG	Default Training				*ReasonKGE*			
		MRR	*Hits@1*	*Hits@3*	*Hits@10*	*MRR*	*Hits@1*	*Hits@3*	*Hits@10*
TransE	**LUBM3U**	0.119	0.069	0.139	0.214	0.135	0.079	0.162	0.256
	Yago3-10	0.226	0.044	0.349	0.537	0.367	0.197	0.511	0.629
	DBpedia15k	0.109	0.061	0.124	0.206	0.118	0.101	0.132	0.299
ComplEx	**LUBM3U**	0.159	0.119	0.168	0.242	0.233	0.195	0.240	0.313
	Yago3-10	0.482	0.400	0.521	0.643	0.530	0.453	0.577	0.668
	DBpedia15k	0.099	0.061	0.109	0.174	0.115	0.125	0.162	0.221

epochs during which, for each subject and object of a triple, $m >= 1$ negative examples are generated. We exploit the optimal value of m tuned for the respective baseline model. In the first iteration, m negative samples are generated using the default random sampling strategy[4]. In the subsequent iterations, we use the trained model to obtain the top $k = 1$ subject and object predictions and compute the inconsistent negative samples to be used for the next iteration of the embedding training as described in Section 3. The number m of negative samples for the next iteration is dynamically computed based on the statistical mean of the size of the generalized samples sets as an indicator.

4.2 Results

The results of the conducted experiments illustrate the benefit of *ReasonKGE* in producing higher quality predictions with less inconsistencies compared to the baselines.

Link prediction quality. Table 3 reports the results for the link prediction task over both TransE and ComplEx trained using the default random sampling strategy [9] (left) and using *ReasonKGE* (right) for 3 iterations. For fair comparison, the default training was also carried out for the same duration as *ReasonKGE* (i.e., 300 epochs).

One can observe that reasoning-based sampling consistently achieves better results than random sampling for training all considered embeddings on all KGs. For the Yago3 dataset the improvements are the most significant achieving more than 10% enhancement for all measures over TransE. This indicates the advantage of ontology-based reasoning for enhancing the existing KG embeddings.

By keeping the same training configuration and total number of training epochs, we ensure that the reflected performance gains are not merely due to additional training steps, but rather a result of the proposed reasoning-based approach.

Consistency of predictions. In Table 4, we measure the proportion of inconsistent facts that were obtained when retrieving *top-k* ($k = \{1, 10\}$) predictions for the triples in the test set. We report the inconsistency values both for the prediction of the subject and the object of the triple separately. From the results, we can observe that for all models *ReasonKGE* managed to reduce the ratio of inconsistent predictions over the test sets compared to the results of training the models using default random sampling.

[4] For each triple the subject (resp. object) is randomly perturbed to obtain m samples [9]

Table 4: Ratio of inconsistent predictions (the lower, the better).

Model	KG	Prediction	Default Training		ReasonKGE	
			Inc@1	*Inc@10*	*Inc@1*	*Inc@10*
TransE	**LUBM3U**	*subject*	0.169	0.270	0.037	0.133
		object	0.095	0.097	0.005	0.007
	YAGO3-10	*subject*	0.075	0.280	0.075	0.273
		object	0.026	0.136	0.020	0.117
	DBpedia15K	*subject*	0.311	0.652	0.217	0.585
		object	0.413	0.538	0.170	0.460
ComplEx	**LUBM3U**	*subject*	0.041	0.097	0.036	0.069
		object	0.008	0.012	0.005	0.007
	YAGO3-10	*subject*	0.113	0.198	0.071	0.143
		object	0.037	0.115	0.015	0.074
	DBpedia15K	*subject*	0.488	0.667	0.344	0.583
		object	0.397	0.585	0.318	0.533

This illustrates that the proposed procedure improves embeddings with respect to the overall consistency of their predictions.

Static vs dynamic reasoning-based negative sampling. The previously proposed methods [11] for ontology-based negative sampling, generate the negative samples for all triples of the KG in the pre-processing step. We refer to such sampling as *static*, which is in contrast to our *dynamic ReasonKGE* method. In the *static* approach, the ontological axioms are leveraged to compute the classes of the negative entities for each entity (the negative samples remain the same regardless of the triple and context the entity occurs in). Following the previous work [11], we generated such *static* negative samples for the DBpedia15k dataset and trained the ComplEx and TransE models using them, while keeping the number of epochs the same as for *ReasonKGE* method. For the link prediction task, the *static* approach resulted in the MRR score of 0.101 for TransE and 0.098 for ComplEx, while the Hits@10 measure was 0.254 for TransE and 0.193 for ComplEx. For both cases, *ReasonKGE* method marginally outperforms the *static* approach, which witnesses the benefits of exploiting inconsistent predictions as negative samples dynamically using our method compared to their pre-computation[5].

5 Related Work

Negative sampling strategies.. The closest to our method is the work [11], in which ontologies are used to generate a selection of negative samples in the pre-processing step for training a certain embedding model. While we use this pre-processing based sampling [11] as a baseline for comparison in Section 4, our method is different in that we do not generate all negative examples at once, but rather compute them iteratively on demand relying on the inconsistent predictions produced by the given embedding.

[5] Further analysis is available in the technical report at https://tiny.one/tp4vark

Another related method is concerned with type-constrained negative sampling [21]. Given a triple from the KG, the negative candidates (subjects or objects) are mined by constraining the entities to belong to the same type as that of the subject or object of the original triple. However, unlike our inconsistency-driven method, the typed-constrained sampling can generate false negatives. This sampling method can be in principle also used as the starting point for our method instead of the random sampling.

More distant random negative samplings generate false candidate triples based on the (local) closed world assumption [26]. Alternatives include Distributional Negative Sampling (DNS) [12] and its variation [2], where during training, given a positive triple, negative examples are generated by replacing it's entity with other similar entities. Unlike in our method, no ontological information is considered in these sampling strategies. The same holds for the triple perturbation or triple corruption approach [29].

Nearest Neighbor and *Near Miss sampling* [20] resp. exploit a pre-trained embedding model for generating negative samples by selecting triples that are close to the positive target triple in vector space. Intuitively, this strategy is supposed to help the model to learn to discriminate between positives and negatives that are very similar to each other. These approaches are similar to ours, in that the embedding training procedure itself is exploited for the generation of negative samples. However, in [20] no ontological knowledge is taken into account which is in contrast to our work.

Another research direction concerns making use of Generative Adversarial Networks (GANs) [38,34,10] for negative sampling. The work [1] presents structure-aware negative sampling (SANS), which utilizes the graph structure by selecting negative samples from a node's neighborhood. The NSCaching sampling method [39] suggests to sample negatives from a cache that can dynamically hold large-gradient samples. While in these works negative triples are updated dynamically like in our method, these approaches are totally different from ours, as they rely purely on the machine learning techniques, and do not consider any extra ontological knowledge. Thus, the proposals are rather complementary in nature.

Integration of ontological knowledge into KG embeddings. Another relevant line of work concerns the integrating of the ontological knowledge directly into the embedding models [11,24,40,36,21,18], which is typically done via changes of the loss function, rather than negative sampling. The recent work [36] suggests to exploit ontological reasoning for verifying consistency of predictions made by a machine learning method (e.g., embedding or rule learning). However, instead of feeding inconsistent predictions back to the given embedding model, the authors propose to get rid of them and feed other consistent predictions along with the original KG as input to a further KG completion method. In [18] the ontology is explicitly included in the training data to jointly embed entities and concepts. By treating the ontology and KG in the same way, only very restricted ontological knowledge is accounted for.

Our work can be also positioned broadly within neural-symbolic methods, and we refer the reader to [37,6] for other less related neural-symbolic approaches.

Inconsistency in ontologies. The problems of explaining and handling inconsistency in ontologies have been tackled in different settings [19,8,27,31,7,22]. However, typically these works focus on detecting inconsistency [19,8], scalable reasoning [27,31], or performing reasoning in the presence of such inconsistency [7,22] assuming that the

KG is constructed and complete. In other words, these approaches deal purely with data cleaning rather than KG completion. In contrast, our method integrates the reasoning process into the embedding models to improve the accuracy of predicted triples.

6 Conclusion

We have presented a method for ontology-driven negative sampling that proceeds in an iterative fashion by providing at each iteration negative samples to the embedding model on demand from its inconsistent predictions along with their generalizations. The main takeaway message of this work is that targeted negative example generation is beneficial for training the model to predict consistent facts as witnessed by our empirical evaluation on state-of-the-art KGs equipped with ontologies. Importantly, one of the advantages of our method is that it is independent of the embedding model used, and can be exploited for improving any existing embedding-based methods.

There are several exciting directions for future work. First, integrating the developed negative sampling method into the combination of rule learning and embedding-based approaches [36] for KG completion is promising. Second, extending the proposed approach to target other more expressive ontology languages is a relevant future direction. Last but not least, adapting our method to jointly clean and complete KGs can be helpful for facilitating the automatic KG curation.

References

1. Ahrabian, K., Feizi, A., Salehi, Y., Hamilton, W.L., Bose, A.J.: Structure-aware negative sampling in knowledge graphs. In: EMNLP 2020. pp. 6093–6101 (2020)
2. Alam, M.M., Jabeen, H., Ali, M., Mohiuddin, K., Lehmann, J.: Affinity dependent negative sampling for knowledge graph embeddings. In: Proceedings of the Workshop on DL for KGs (DL4KG2020) - (ESWC 2020) (2020)
3. Artale, A., Calvanese, D., Kontchakov, R., Zakharyaschev, M.: The DL-Lite family and relations. CoRR **abs/1401.3487** (2014)
4. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.G.: Dbpedia: A nucleus for a web of open data. In: ISWC. pp. 722–735 (2007)
5. Baader, F., Horrocks, I., Sattler, U.: Description logics. In: Handbook on Ontologies, pp. 21–43 (2009)
6. Bianchi, F., Rossiello, G., Costabello, L., Palmonari, M., Minervini, P.: Knowledge graph embeddings and explainable AI. In: Tiddi, I., Lécué, F., Hitzler, P. (eds.) KGs for XAI: Foundations, Applications and Challenges, vol. 47, pp. 49–72. IOS Press (2020)
7. Bienvenu, M.: A short survey on inconsistency handling in ontology-mediated query answering. Künstliche Intell. **34**(4), 443–451 (2020)
8. Bischof, S., Krötzsch, M., Polleres, A., Rudolph, S.: Schema-agnostic query rewriting in SPARQL 1.1. In: The Semantic Web - ISWC 2014 - 13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part I. pp. 584–600 (2014)
9. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: NeurIPS. pp. 2787–2795 (2013)
10. Cai, L., Wang, W.Y.: KBGAN: adversarial learning for knowledge graph embeddings. In: NAACL-HLT 2018. pp. 1470–1480 (2018)

11. d'Amato, C., Quatraro, N.F., Fanizzi, N.: Injecting background knowledge into embedding models for predictive tasks on knowledge graphs. In: ESWC (2021), https://openreview.net/forum?id=C9g-pwemYxd
12. Dash, S., Gliozzo, A.: Distributional negative sampling for knowledge base completion. CoRR **abs/1908.06178** (2019)
13. Dettmers, T., Minervini, P., Stenetorp, P., Riedel, S.: Convolutional 2d knowledge graph embeddings. In: AAAI. pp. 1811–1818 (2018)
14. Glimm, B., Horrocks, I., Motik, B., Stoilos, G., Wang, Z.: Hermit: An OWL 2 reasoner. J. Autom. Reasoning **53**(3), 245–269 (2014)
15. Glimm, B., Kazakov, Y., Liebig, T., Tran, T.K., Vialard, V.: ISWC. pp. 180–195 (2014)
16. Glimm, B., Kazakov, Y., Tran, T.: Ontology materialization by abstraction refinement in horn SHOIF. In: AAAI. pp. 1114–1120 (2017)
17. Guo, Y., Pan, Z., Heflin, J.: LUBM: A benchmark for OWL knowledge base systems. J. Web Semant. **3**(2-3), 158–182 (2005)
18. Hao, J., Chen, M., Yu, W., Sun, Y., Wang, W.: Universal representation learning of knowledge bases by jointly embedding instances and ontological concepts. In: KDD. pp. 1709–1719 (2019)
19. Horridge, M., Parsia, B., Sattler, U.: Explaining inconsistencies in owl ontologies. In: Scalable Uncertainty Management. pp. 124–137 (2009)
20. Kotnis, B., Nastase, V.: Analysis of the impact of negative sampling on link prediction in knowledge graphs. CoRR **abs/1708.06816** (2017), http://arxiv.org/abs/1708.06816
21. Krompaß, D., Baier, S., Tresp, V.: Type-constrained representation learning in knowledge graphs. In: ISWC. pp. 640–655 (2015)
22. Lembo, D., Lenzerini, M., Rosati, R., Ruzzi, M., Savo, D.F.: Inconsistency-tolerant query answering in ontology-based data access. J. Web Semant. **33**, 3–29 (2015)
23. Liu, Y., Li, H., Garcia-Duran, A., Niepert, M., Onoro-Rubio, D., Rosenblum, D.S.: Mmkg: multi-modal knowledge graphs. In: ESWC. pp. 459–474 (2019)
24. Minervini, P., Demeester, T., Rocktäschel, T., Riedel, S.: Adversarial sets for regularising neural link predictors. In: UAI (2017)
25. Motik, B., Patel-Schneider, P.F., Grau, B.C.: OWL 2 Web Ontology Language Direct Semantics (Second Edition). Tech. rep. (2012), https://www.w3.org/TR/owl-direct-semantics/
26. Nickel, M., Murphy, K., Tresp, V., Gabrilovich, E.: A review of relational machine learning for knowledge graphs. Proc. IEEE **104**(1), 11–33 (2016)
27. Paulheim, H., Gangemi, A.: Serving dbpedia with dolce – more than just adding a cherry on top. In: The Semantic Web - ISWC 2015. pp. 180–196. Springer (2015)
28. Ruffinelli, D., Broscheit, S., Gemulla, R.: You CAN teach an old dog new tricks! on training knowledge graph embeddings. In: ICLR (2020)
29. Socher, R., Chen, D., Manning, C.D., Ng, A.Y.: Reasoning with neural tensor networks for knowledge base completion. In: NIPS. pp. 926–934 (2013)
30. Suchanek, F.M., Kasneci, G., Weikum, G.: Yago: A Core of Semantic Knowledge. In: WWW
31. Tran, T., Gad-Elrab, M.H., Stepanova, D., Kharlamov, E., Strötgen, J.: Fast computation of explanations for inconsistency in large-scale knowledge graphs. In: WWW 2020. pp. 2613–2619 (2020)
32. Trouillon, T., Welbl, J., Riedel, S., Gaussier, É., Bouchard, G.: Complex embeddings for simple link prediction. In: ICML. pp. 2071–2080 (2016)
33. Vrandecic, D., Krötzsch, M.: Wikidata: a free collaborative knowledgebase. Commun. ACM **57**(10), 78–85 (2014)
34. Wang, P., Li, S., Pan, R.: Incorporating GAN for negative sampling in knowledge representation learning. In: AAAI. pp. 2005–2012 (2018)
35. Wang, Q., Mao, Z., Wang, B., Guo, L.: Knowledge graph embedding: A survey of approaches and applications. IEEE Trans. Knowl. Data Eng. **29**(12), 2724–2743 (2017)

36. Wiharja, K., Pan, J.Z., Kollingbaum, M.J., Deng, Y.: Schema aware iterative knowledge graph completion. J. Web Semant. **65**, 100616 (2020)
37. Zhang, J., Chen, B., Zhang, L., Ke, X., Ding, H.: Neural-symbolic reasoning on knowledge graphs. CoRR **abs/2010.05446** (2020)
38. Zhang, Y., Yao, Q., Chen, L.: Efficient, simple and automated negative sampling for knowledge graph embedding. CoRR **abs/2010.14227** (2020), https://arxiv.org/abs/2010.14227
39. Zhang, Y., Yao, Q., Shao, Y., Chen, L.: Nscaching: Simple and efficient negative sampling for knowledge graph embedding. In: ICDE. pp. 614–625 (2019)
40. Ziegler, K., Caelen, O., Garchery, M., Granitzer, M., He-Guelton, L., Jurgovsky, J., Portier, P., Zwicklbauer, S.: Injecting semantic background knowledge into neural networks using graph embeddings. In: 26th IEEE International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises, WETICE. pp. 200–205 (2017)