Preprints.org

Article

Knowledge Reasoning via Jointly Modeling Knowledge Graphs and Soft Rules

Yinyu Lan , Shizhu He [*] , Kang Liu , Jun Zhao

Article

Knowledge Reasoning via Jointly Modeling Knowledge Graphs and Soft Rules

Yinyu Lan [1,2] ![ORCID], **Shizhu He** [1,2,*], **Kang Liu** [1,2] **and Jun Zhao** [1,2]

[1] Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China; lanyinyu19@mails.ucas.ac.cn; kliu@nlpr.ia.ac.cn; jzhao@nlpr.ia.ac.cn
[2] School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100190, China
[*] Correspondence: shizhu.he@nlpr.ia.ac.cn

Abstract: Knowledge graphs (KGs) play a crucial role in many applications, such as question answering, but incompleteness is an urgent issue for their broad application. Much research in knowledge graph completion (KGC) has been performed to resolve this issue. The methods of KGC can be classified into two major categories: rule-based reasoning and embedding-based reasoning. The former has high accuracy and good interpretability, but a major challenge is to obtain effective rules on large-scale KGs. The latter has good efficiency and scalability, but it relies heavily on data richness and cannot fully use domain knowledge in the form of logical rules. We propose a novel method that injects rules and learns representations iteratively to take full advantage of rules and embeddings. Specifically, we model the conclusions of rule groundings as 0-1 variables and use a rule confidence regularizer to remove the uncertainty of the conclusions. The proposed approach has the following advantages: 1) It combines the benefits of both rules and knowledge graph embeddings (KGEs) and achieves a good balance between efficiency and scalability. 2) It uses an iterative method to continuously improve KGEs and remove incorrect rule conclusions. Evaluations on two public datasets show that our method outperforms the current state-of-the-art methods, improving performance by 2.7% and 4.3% in mean reciprocal rank (MRR).

Keywords: distributed representation; knowledge graph; Link prediction; logical rule

1. Introduction

A knowledge graph (KG) organizes knowledge as a set of interlinked triples, and a triple ((*head entity*, *relation*, *tail entity*), simply represented as *(h, r, t)*) indicates the fact that two entities have a certain relation. Rich structured and formalized knowledge has become a valuable resource to support downstream tasks, for example, question answering [1,2] and recommender systems [3,4].

Although KGs such as DBpedia [5], Freebase [6] and NELL [7] contain large amounts of entities, relations, and triples, they are far from complete, which is an urgent issue for their broad application. To address this, the task of knowledge graph completion (KGC) has been proposed and has attracted growing attention; it utilizes knowledge reasoning techniques to perform automatic discovery of new facts based on existing facts in a KG [8].

At present, the methods of KGC can be classified into two major categories: 1) One type of method uses explicit reasoning rules; it obtains the reasoning rules through inductive learning and then deductively infers new facts. 2) Another method is based on representation learning instead of directly modeling rules, aiming to learn a distributed embedding for entities and relations and perform generalization in numerical space.

Rule-based reasoning is accurate and can provide interpretability for the inference results. Domain experts can handcraft these rules [9] or can mine them from the KG with an induction algorithm such as AMIE [10]. Traditional methods such as expert systems [11,12] use hard logical rules to make predictions. For example, as shown in Figure 1, given the logical rule $(x, born\_in, y) \land (y, city\_of, z) \Rightarrow (x, nationality, z)$ and the two facts that (Chicago, city_of, USA) and (Mary, born_in, Chicago), we can infer the fact (Mary, nationality, USA). A large number of new facts (conclusions) can be derived based

on forward chaining inference. However, for large-scale KGs, sufficient and effective reasoning rules are difficult and expensive to obtain. Moreover, in many cases, the logical rules may be imperfect or even self-contradictory. Therefore, it is essential to model the uncertainty of (soft) logical rules effectively.
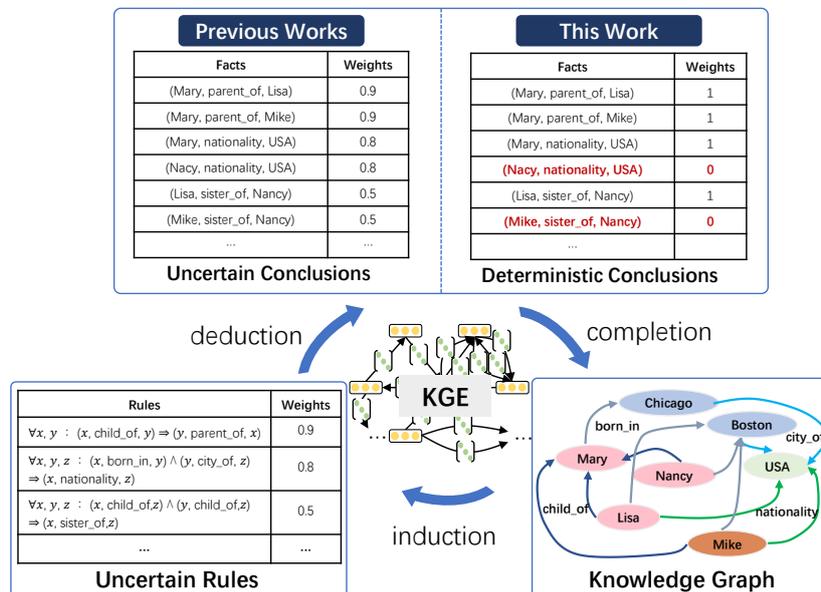

Figure 1. We propose a novel iterative knowledge reasoning framework by fusing logical rules into a KGE. Previous methods associate each conclusion with a weight derived from the corresponding rule. In contrast, our method can infer which conclusion is true via jointly modeling the deterministic KG and uncertain soft rules.

The methods of determining KGEs learn to embed entities and relations into a continuous low-dimensional space [13,14]. These embeddings retain the semantic meaning of entities and relations, which can be used to predict missing triples. In addition, they can be effectively trained using stochastic gradient descent. However, this kind of method cannot fully use logical rules, which compactly encode domain knowledge and are helpful in various applications. Good embedding relies heavily on data richness, so these methods have difficulty learning useful representations for sparse entities [15,16].

In fact, both rule-based methods and embedding-based methods have advantages and disadvantages in the KGC task. Logical rules are accurate and interpretable, and embedding is flexible and computationally efficient. To achieve more precise knowledge completion, recently, there has also been research on combining the advantages of logical rules and KGEs. Mixed techniques can infer missing triples effectively by exploiting and modeling uncertain logical rules. Some existing methods have aimed to iteratively learn KGEs and rules [16], and some other methods also utilize soft rules or groundings of rules to regularize the learning of KGEs [17,18].

The integration of logical rules and knowledge graph embeddings can achieve more efficient and accurate knowledge completion. Current methods model uncertain rules and add soft labels to conclusions by t-norm-based fuzzy logic [19]; they further utilize the conclusions to perform forward reasoning [17] or to enhance the KGE [16]. However, in most KGs, the facts are deterministic. Therefore, we believe that rules are uncertain but conclusions are deterministic in knowledge reasoning, as shown in Figure 1; each fact is only absolutely true or false. Previous methods associate each conclusion with a weight derived from the corresponding rule. In contrast, we propose inferring that all conclusions are true (*e.g.*, ($Mary, nationality, USA$)) or not (*e.g.*, ($Mike, sister\_of, Nancy$)) (the other fact, *i.e.*, ($Mike, gender, male$), indicates that Mike is not Nancy's sister) by jointly modeling the deterministic KG and soft rules.

Specifically, we first mine soft rules from the knowledge graph and then infer conclusions as candidate facts. Second, the KG, conclusions, and weighted rules are also used as resources to learn embeddings. Third, through the definition of deterministic conclusion loss, the conclusion labels are modeled as 0-1 variables, and the confidence loss of a rule is also used to constrain the conclusions. Finally, the embedding learning stage removes the noise in the candidate conclusions, and then the proper conclusions are added back to the original KG. The above steps are performed iteratively. We empirically evaluate the proposed method on public datasets from two real large-scale KGs: DBpedia and Freebase. The experimental results show that our method Iterlogic-E (**Iter**ative using **logic** rule for reasoning and learning **E**mbedding) achieves state-of-the-art results on multiple evaluation metrics. Iterlogic-E also achieves improvements of 2.7%/4.3% in mean reciprocal rank (MRR) and 2.0%/4.0% in HITS@1 compared to the state-of-the-art model.

In summary, our main contributions are as follows:

- We propose a novel KGC method, Iterlogic-E, which jointly models logical rules and KGs in the framework of a KGE. Iterlogic-E combines the advantages of both rules and embeddings in knowledge reasoning. Iterlogic-E models the conclusion labels as 0-1 variables and uses a confidence regularizer to eliminate the uncertain conclusions.
- We propose a novel iterative learning paradigm that achieves a good balance between efficiency and scalability. Iterlogic-E not only makes the KG denser but can also filter incorrect conclusions.
- Compared with traditional reasoning methods, Iterlogic-E is more interpretable in determining conclusions. It not only knows why the conclusion holds but also knows which is true and which is false.
- We empirically evaluate Iterlogic-E with the task of link prediction on multiple benchmark datasets. The experimental results indicate that Iterlogic-E can achieve state-of-the-art results on multiple evaluation metrics. The qualitative analysis proves that Iterlogic-E is more robust for rules with different confidence levels.

2. Related Work

Knowledge reasoning aims to infer certain entities over KGs as the answers to a given query. A query in KGC is a head entity h (or a tail entity t) and a relation r. Given $(h, r, ?)$ (or $(?, r, t)$), KGC aims to find the right tail entity t (or head entity h) in the KG that satisfies the triple (h, r, t). Next, we review the three most relevant classes of KGC methods.

2.1. Rule-Based Reasoning

Logical rules can encode human knowledge compactly, and early knowledge reasoning was primarily based on first-order logical rules. Existing rule-based reasoning methods have primarily utilized search-based inductive logic programming (ILP) methods, usually searching and pruning rules. Based on the partial completeness assumption, AMIE [10] introduces a revised confidence metric, which is well suited for modeling KGs. By query rewriting and pruning, AMIE+ [20] is optimized to expand to larger KGs. Additionally, AMIE+ improves the precision of the forecasts by using joint reasoning and type information. In this paper, we employ AMIE+[1] to mine horn rules from a KG. Rule-based reasoning methods can be combined with multiple probability graph models. A Markov logic network (MLN) [21] is a typical model. Based on preprovided rules, it builds a probabilistic graph model and then learns the weights of rules. However, due to the complicated graph structure among triples, the reasoning in an MLN is time-consuming and difficult, and the incompleteness of KGs also impacts the inference results. In contrast, Iterlogic-E uses rules to enhance KGEs with more effective inference.

---

[1] https://github.com/lajus/amie

2.2. Embedding-Based Reasoning

Recently, embedding-based methods have attracted much attention; they aim to learn distributed embeddings for entities and relations in KGs. Generally, current KGE methods can be divided into three classes: 1) translation-based models that learn embeddings by translating one entity into another entity through a specific relation [22,23]; 2) compositional models that use simple mathematical operations to model facts, including linear mapping [24], bilinear mapping [25–27], and circular correlation [28]; 3) neural network-based models that utilize a multilayer neural structure to learn embeddings and estimate the plausibility of triples with nonlinear features, for example, R-GCN [29], ConvE [30] and and so on [31–33]. The above methods learn representations based only on the triples existing in KGs, and the sparsity of data limits them. To solve this problem and learn semantic-rich representations, recent works further attempted to incorporate information beyond triples, *e.g.*, contextual information [34], entity type information [35,36], ontological information [37], taxonomic information [38], textual descriptions [39] and hierarchical information [49]. In contrast, the proposed Iterlogic-E uses embeddings to remove incorrect conclusions obtained by rules, which combines the advantages of rules and embeddings.

2.3. Hybrid Reasoning

Both rule-based and embedding-based methods have advantages and disadvantages. Recent works have integrated these two kinds of reasoning methods. Guo et al. [40] attempted to learn a KGE from rule groundings and triples together. Wang et al. [41] used asymmetric and transitive information to approximately order relations by maximizing the margin between negative and positive logical rules. Zhang et al. [17] and Guo et al. [42] obtained KGEs with supervision from soft rules, proving the effectiveness of logical rules. Qu et al. [43] used an MLN to model logical rules and inferred new triples to enhance KGEs. Guo et al. [18] enhanced KGEs by injecting grounding rules. Niu et al. [50] enhanced KGEs by extracting commonsense from factual triples with entity concepts. In addition, some previous methods that enhance embeddings by iterative learning were studied in early works. Zhang et al. [16] aimed to improve a sparse entity representation through iterative learning and update the confidence of rules through embeddings. In contrast, Iterlogic-E models the conclusion labels as 0-1 variables and uses confidence regularization loss to eliminate the uncertain conclusions. Such labels are easier to train on.

3. The Proposed Method

This section introduces our proposed method Iterlogic-E. We first give an overview of our method, including the entire iterative learning process. Then, we detail the two parts of Iterlogic-E: rule mining and reasoning and embedding learning. Last, we discuss the space and time complexity of Iterlogic-E and discuss connections to related works [16,17].

3.1. Overview

Given a KG $\mathcal{G} = \{\mathcal{E}, \mathcal{R}, \mathcal{T}\}$, $\mathcal{T} = \{(h, r, t)\}$, $r \in \mathcal{R}$ is a relation and $h, t \in \mathcal{E}$ are entities. As discussed in Section 1, on the one hand, embedding learning methods do not make full use of logical rules and suffer from data sparsity. On the other hand, precise rules are difficult to obtain efficiently and cannot cover all facts in KGs. Our goal is to improve the embedding quality by explicitly modeling the reasoned conclusions of logical rules, removing incorrect conclusions, and improving the confidence of the rules at the same time. Figure 2 shows the overview of Iterlogic-E given a toy knowledge graph. Iterlogic-E is a general framework that can fuse different KGE models.
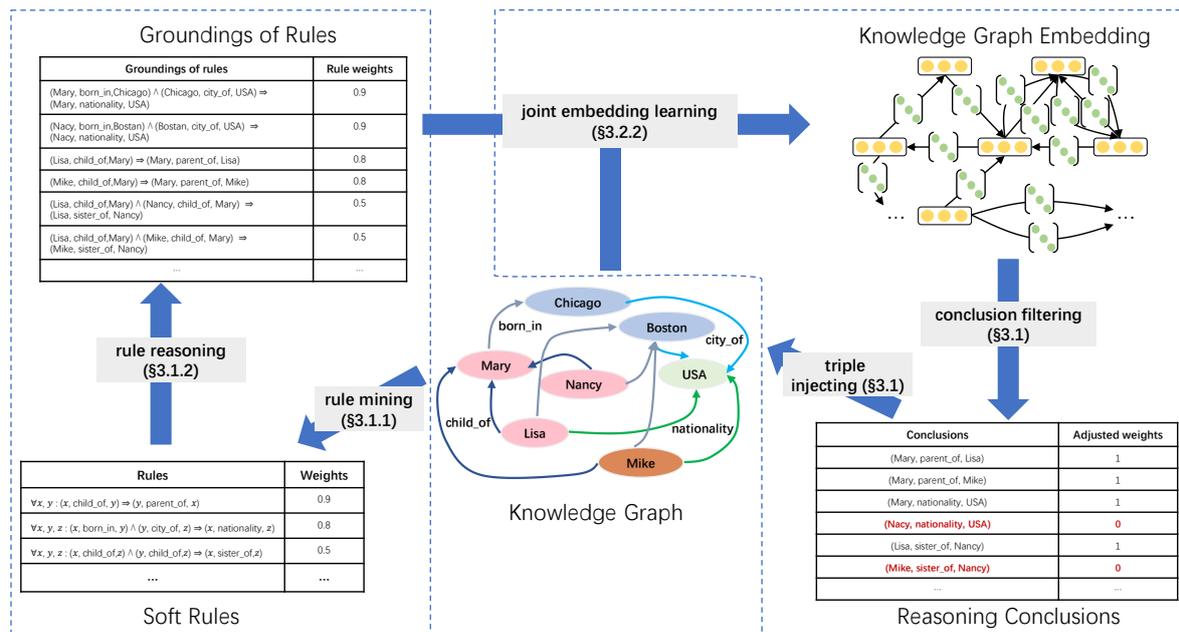
Groundings of Rules

Groundings of rules	Rule weights
(Mary, born_in,Chicago) ∧ (Chicago, city_of, USA) ⇒ (Mary, nationality, USA)	0.9
(Nacy, born_in,Bostan) ∧ (Bostan, city_of, USA) ⇒ (Nacy, nationality, USA)	0.9
(Lisa, child_of,Mary) ⇒ (Mary, parent_of, Lisa)	0.8
(Mike, child_of,Mary) ⇒ (Mary, parent_of, Mike)	0.8
(Lisa, child_of,Mary) ∧ (Nancy, child_of, Mary) ⇒ (Lisa, sister_of, Nancy)	0.5
(Lisa, child_of,Mary) ∧ (Mike, child_of, Mary) ⇒ (Mike, sister_of, Nancy)	0.5
...	...

joint embedding learning (§3.2.2)

Knowledge Graph Embedding

rule reasoning (§3.1.2)

rule mining (§3.1.1)

conclusion filtering (§3.1)

triple injecting (§3.1)

Knowledge Graph

Chicago, Boston, born_in, city_of, Mary, Nancy, USA, child_of, Lisa, nationality, Mike

Soft Rules

Rules	Weights
$\forall x, y : (x, child\_of, y) \Rightarrow (y, parent\_of, x)$	0.9
$\forall x, y, z : (x, born\_in, y) \wedge (y, city\_of, z) \Rightarrow (x, nationality, z)$	0.8
$\forall x, y, z : (x, child\_of,z) \wedge (y, child\_of,z) \Rightarrow (x, sister\_of,z)$	0.5
...	...

Reasoning Conclusions

Conclusions	Adjusted weights
(Mary, parent_of, Lisa)	1
(Mary, parent_of, Mike)	1
(Mary, nationality, USA)	1
(Nacy, nationality, USA)	0
(Lisa, sister_of, Nancy)	1
(Mike, sister_of, Nancy)	0
...	...

Figure 2. The framework details Iterlogic-E with two iterative stages: (i) rule mining and reasoning and (ii) embedding learning. Stage (i) generates rules and grounding rules to obtain new conclusions. Stage (ii) jointly models the conclusions of grounding rules and the KG in learning embeddings. After embedding learning, the conclusions are injected into the KG, and then the rule reasoning module is executed to start the next round of iterative training

Iterlogic-E has two iterative steps: (i) rule mining and reasoning and (ii) embedding learning. In the rule mining and reasoning step, there are two modules: rule mining and rule reasoning. The mining configuration, such as the maximum length of rules and the confidence threshold of rules, and KG triples are input to the rule mining module. Then, it automatically obtains the soft rules from these inputs. The KG triples and the extracted soft rules are input into the rule reasoning module to infer new triples. After that, the new triples are appended to the embedding learning step as candidate conclusions. In the embedding learning step, relations are modeled as a linear mapping operation, and triple plausibility is represented as the correlation between the head and tail entities after the operation. Finally, the incorrect conclusions are filtered out by labeling the conclusions with their scores[2]. The right conclusion will be added back to the original KG triples, and then the rule reasoning module is performed to start the next cycle of iterative training.

3.2. Rule Mining and Reasoning

The first step is composed of the rule mining module and the rule reasoning module. We introduce these two modules in detail below.

3.2.1. Rule Mining

We extract soft rules from the KG using the state-of-the-art rule mining method AMIE+ [20] in this module. AMIE+ applies principal component analysis (PCA) confidence to estimate the reliability of a rule since its partial completeness assumption is more suited to real-world KGs. Additionally, AMIE+ defines a variety of restriction types to help extract applicable rules, *e.g.*, the maximum length of the rule. After the rule mining module receives the KG triples and the mining configuration, it executes

---

[2] In the experiments, we choose the conclusion with a normalized score of more than 0.99 as the true conclusion.

the AMIE+ algorithm and outputs soft rules. Although rules can be re-mined in each iteration, we only run the rule mining module once for efficiency reasons.

3.2.2. Rule Reasoning

The logical rule set is denoted as $\mathcal{F} = \{(f, c)\}$. f is in the form of $\forall x, y, z : (x, r_{p1}, y) \wedge (y, r_{p2}, z) \overset{c}{\Rightarrow} (x, r_c, z)$. x, y and z represent variables of different entities, and r_{p1}, r_{p2} and r_{r_c} represent different relations. The left side of the symbol $\Rightarrow$ is the premise of the rule, which is composed of several connected atoms. The right side is only a single atom, which is the rule's conclusion. The horn rules are closed [25], where continuous relations share the intermediate entity and the first and last entities of the premise appear as the head and tail entities of the conclusion. Such rules can provide interpretive insights. A rule's length is equal to the number of atoms in the premise. For example, $\forall x, y, z : (x, born_i n, y) \wedge (y, city, z) \overset{0.8}{\Rightarrow} (x, nationality, z)$ is a length-2 rule. This rule reflects the reality that, most likely, a person's nationality is the country in which he or she was born. The rule f has a confidence level of 0.8. The higher the confidence of the rule, the more likely it is to hold.

The reasoning procedure consists of instantiating the rule's premise and obtaining a large number of fresh conclusion triples. One of the most common approaches is forward chaining, also known as the match-select-act cycle, which works in three-phase cycles. Forward chaining matches the currently existing facts in the KG with all known rule premises in one cycle to determine the rules that can be satisfied. Finally, the selected rule's conclusions are derived, and if the conclusions are not already in the KG, they are added as new facts. This cycle should be repeated until no new conclusions emerge. However, if soft rules are used, forward chain reasoning will lead to incorrect conclusions. Therefore, we run one reasoning cycle in every iteration.

3.3. Embedding Learning

In this section, we present a joint embedding learning approach that allows the embedding model to learn from KG triples, conclusion triples, and soft rule confidence all at the same time. First, we will examine a basic KGE model, and then we will describe how to incorporate soft rule conclusions. Finally, we detail the overall training goal.

3.3.1. A Basic KGE Model

Different KGE models have different score functions that aim to obtain a suitable function to map the triple score to a continuous true value in [0, 1], *i.e.*, $\phi : \mathcal{E} \times \mathcal{R} \times \mathcal{E} \to (0, 1)$, which indicates the probability that the triple holds. We follow [17,18] and choose ComplEx [26] as a basic KGE model. It is important to note that our proposed framework can be combined with an arbitrary KGE model. Theoretically, using a better base model can continue improving performance. Therefore, We also experiment with RotatE as a base model. Below we take ComplEx as an example to introduce. ComplEx assumes that the entity and relation embeddings exist in a complex space, *i.e.*, $e \in \mathbb{C}^d$ and $r \in \mathbb{C}^d$, where d is the dimensionality of the complex space. Using plurals to represent entities and relations can better model antisymmetric and symmetric relations (*e.g.*, kinship and marriage) [26]. Through a multilinear dot product, ComplEx scores every triple:

$$F(h, r, t) = \text{Re}(h^{\mathrm{T}} \text{diag}(r)\bar{t}) = \text{Re}(\textstyle\sum_i [h]_i [r]_i [t]_i), \tag{1}$$

where the $\text{Re}(\cdot)$ function takes the real part of a complex value and the $\text{diag}(\cdot)$ function constructs a diagonal matrix from r; $\bar{t}$ is the conjugate of t; and $[\cdot]_i$ is the i-th entry of a vector. To predict the probability, ComplEx further uses the sigmoid function for normalization:

$$\phi(h, r, t) = \sigma(F(h, r, t)) = \sigma(\text{Re}(h^{\mathrm{T}} \text{diag}(r)\bar{t})), \tag{2}$$

where $\sigma(\cdot)$ is the sigmoid function. By minimizing the logistic loss function, ComplEx learns the relation and entity embeddings:

$$\sum_{(h,r,t)\in\mathcal{T}\cup\mathcal{T}'} \log(1 + \exp(-y_{hrt} \cdot f(h, r, t))), \tag{3}$$

where $\mathcal{T}'$ is a set of sampled negative examples and y_{hrt} is the label of a positive or negative triple.

3.3.2. Joint Modeling KG and Conclusions of Soft Rules

To model the conclusion label as a 0-1 variable, based on the current KGE model's scoring function, we follow ComplEx and use the function $f(\cdot)$ as the scoring function for conclusion triples:

$$S_i = \sigma(F(h_i, r_i, t_i)), (h_i, r_i, t_i) \in \mathcal{C}_f, \tag{4}$$

where $\mathcal{C}_f$ is the set of conclusion triples derived from rule f and $F(\cdot)$ is the score function defined in Equation (1). Aiming to regularize this scoring function so that it approaches 0 or 1, and to distinguish between true and false conclusions, we use a quadratic function with a symmetry axis of 0.5. Therefore, the conclusion score is the smallest when it is close to 0 or 1. Therefore, we define the deterministic conclusion loss L_{dc} as follows:

$$L_{dc} = -\frac{1}{|\mathcal{C}_f|} \sum_{(h_i,r_i,t_i)\in\mathcal{C}_f} \|S_i - 0.5\|^2. \tag{5}$$

According to the definition of rule confidence in [10], the confidence of a rule f in a KB $\mathcal{G}$ is the proportion of true conclusions among the true conclusions and false conclusions. Therefore, we can define the confidence loss of a rule as follows:

$$L_{rc} = \left\| \frac{1}{|\mathcal{C}_f|} \sum_{(h_i,r_i,t_i)\in\mathcal{C}_f} S_i - c_f \right\|^2, \tag{6}$$

where c_f is the confidence of rule f. Therefore, the loss of the conclusions of all the rules L_{ac} can be defined as follows:

$$L_{ac} = \frac{1}{|\mathcal{F}|} \sum_{f\in\mathcal{F}} (L_{dc} + L_{rc}), \tag{7}$$

where $\mathcal{F}$ is the set of all rules. To learn the KGE and rule conclusions at the same time, we minimize the global loss over a soft rule set $\mathcal{F}$ and a labeled triple set $\mathcal{L} = (x_l, y_l)$ (including negative and positive examples). The overall training objective of Iterlogic-E is:

$$\min_\theta \frac{1}{|\mathcal{L}|} \sum_{(x_l,y_l)\in\mathcal{L}} L(-f(x_l) \cdot y_l)$$
$$+ \frac{1}{|\mathcal{F}|} \sum_{f\in\mathcal{F}} (-\frac{1}{|\mathcal{C}_f|} \sum_{(h_i,r_i,t_i)\in\mathcal{C}_f} \|S_i - 0.5\|^2$$
$$+ \left\| \frac{1}{|\mathcal{C}_f|} \sum_{(h_i,r_i,t_i)\in\mathcal{C}_f} S_i - c_f \right\|^2). \tag{8}$$

where the $f(\cdot)$ function denotes the score function and $L(x) = \log(1 + \exp(x))$ is the soft-plus function. In Algorithm 1, we detail the embedding learning procedure of our method. To avoid overfitting, we

further impose l_2 regularization on embedding Θ. Following [18,44], we also imposed nonnegative constraints (NNE) on the entity embedding to learn more effective features.

Algorithm 1: Iterative learning algorithm of Iterlogic-E

    Input: KG triples $\mathcal{T} = \{(e_i, r_k, e_j)\}$, logical rules $\mathcal{F} = \{(f_p, c_p)\}$, the
        number of iterative learning steps M.

    Output: Relation and entity embeddings Θ

1   Randomly initialize relation and entity embeddings $\Theta(0)$;
2   **for** $n \leftarrow 1$ *to* N **do**
3       $\mathcal{C} \leftarrow \varnothing$;
4   **if** $n/[N/M] == 0$ **then**
5   Generate a set of conclusions $\mathcal{C}' = \{(e'_i, r'_k, e'_j)\}$ by rule grounding
              from $\mathcal{T}, \mathcal{F}$;
6           $\mathcal{C} = \mathcal{C} \cup \mathcal{C}'$;
7   **end**
8   Sample a mini-batch $\mathcal{T}^b, \mathcal{C}^b$ from $\mathcal{T}, \mathcal{C}$;
9   Generate a set of negative triples $\mathcal{T}^b_{neg}$;
10      $\mathcal{L}^b \leftarrow \varnothing$;
11   **for** *each* $x_l \in \mathcal{T}^b_{neg} \cup \mathcal{T}^b \leftarrow 1$ *to* N **do**
12         $y_l = +1/-1$;
13         $\mathcal{L}^b \leftarrow \mathcal{L}^b \cup (x_l, y_l)$;
14   **end**
15      $\Theta^{(n)} \leftarrow \Theta^{(n-1)} - \eta\big(\frac{1}{|\mathcal{L}^b|}\sum_{(x_l,y_l)\in\mathcal{L}^b}\nabla_\Theta L(-f(x_l) \cdot y_l) +$
          $\frac{1}{|\mathcal{F}|}\sum_{f\in\mathcal{F}}\big(-\frac{1}{|\mathcal{C}_f|}\sum_{(h_i,r_i,t_i)\in\mathcal{C}_f}\nabla_\Theta \|S_i - 0.5\|^2 +$
          $\nabla_\Theta \left\|\frac{1}{|\mathcal{C}_f|}\sum_{(h_i,r_i,t_i)\in\mathcal{C}_f} S_i - c_f\right\|^2\big)$;   **!** $cf.$ Eq. (8)
16      $\mathcal{C}_t \leftarrow \varnothing$;
17   **for** *each* $x_m \in \mathcal{C}$ **do**
18   **if** $\sigma(f(x_m)) >= 0.99$ **then**
19           $\mathcal{C}_t \cup (x_m)$;
20   **end**
21   **end**
22      $\mathcal{T} = \mathcal{T} \cup \mathcal{C}_t$;
23 **end**
24 **return** $\Theta^{(N)}$

3.4. Discussion

3.4.1. Complexity

In the embedding learning step, we represent relations and entities as complex value vectors, following ComplEx. As a result, the space complexity is $O(n_e d + n_r d)$, where d is the embedding space's dimensionality. The number of relations is n_r, and the number of entities is n_e. Each iteration of the learning process has a time complexity of $O(n_l d + n_c d)$, where n_l/n_c is the number of new conclusions or the number of labeled triples in a mini-batch, as shown in Algorithm 1. Iterlogic-E is similar to ComplEx in that its space and time complexity increase linearly with d. The number of new conclusions in a minibatch is usually considerably lower than the number of initial triples; *i.e.*, $n_c \ll n_l$. As a result, Iterlogic-E's time complexity is very close to that of ComplEx, which needs only $O(n_l d)$ per iteration. Because of the rule mining module's great efficiency and practical constraints, such as the

PCA confidence threshold not being lower than 0.5 and the length of rules not exceeding two, the rule grounding stage's space and time complexity is trivial compared to that of the embedding learning stage. Therefore, we may disregard it when considering the space and time complexity of Iterlogic-E.

4. Experiments and Results

4.1. Datasets

Iterlogic-E is tested on two common datasets: FB15K and DB100K. The first is based on Freebase, which was released by Bordes et al. [22]. The second was taken from DBpedia by Ding et al. [44], and it includes 99,604 entities and 470 relations. For model training, hyperparameter tuning and evaluation, we utilize fixed training, validation, and test sets on both datasets.

With each training dataset, we obtain soft rules and examine rules with a length of no more than 2 to allow efficient extraction. These rules, together with their confidence levels, are automatically retrieved from each dataset's training set using AMIE+ [20], and only those with confidence levels greater than 0.8 are used. Shorter rules are thought to more directly represent logical connections among relations. Therefore, we remove longer rules when all of their relations also exist in shorter ones. Table 1 summarizes the datasets' comprehensive statistics, and Table 2 also includes several rule instances. We can observe from the statistics that the number of rules on both datasets is extremely minimal when compared to the number of triples.

Table 1. Statistics of the datasets, where the columns represent the numbers of entities, relations, training/validation/test triples, and soft rules

Dataset	# Ent	# Rel	# Train/Valid/Test	# Rule
FB15K	14,951	1,345	483,142/50,000/59,071	441
DB100K	99,604	470	597,572/50,000/50,000	25

Table 2. Examples of rules, with confidences, that were extracted from FB15K (top) and DB100K (bottom)

/location/contains$(y, x) \overset{0.84}{\Rightarrow}$ /location/containedby(x,y)
/production_company/films$(y, x) \overset{0.89}{\Rightarrow}$ /location/containedby(x,y)/hud_county_place/place$(x, y) \wedge$ hud_county_place/county$(y, z) \overset{1.0}{\Rightarrow}$ /hud_county_place/county(x, z)
sisterNewspaper$(x, y) \wedge$sisterNewspaper$(z, y) \overset{0.82}{\Rightarrow}$ sisterNewspaper(x, z)
distributingCompany$(x, y) \overset{0.91}{\Rightarrow}$distributingLabel$(x, y)$
nationality$(x, y) \overset{0.99}{\Rightarrow}$stateOfOrigin$(x, y)$

4.2. Link Prediction

Our method was evaluated on link prediction. The goal of this task was to restore a missing triple $(e_i, r_k, ?)$ with the tail entity e_j or $(?, r_k, e_j)$ with the head entity e_i.

4.2.1. Evaluation Protocol

The standard protocol established by [22] is used for evaluation. The head entity e_i is replaced with each entity for every test triple (e_i, r_k, e_j), and the corrupted triple's score is calculated. We record the rank of the right entity e_i by ranking these scores in decreasing order. The mean reciprocal rank (MRR) and the percentage of ranks no greater than N (H@N, N = 1, 3, 10) are used to evaluate the ranking quality of all test triples.

4.2.2. Comparison Settings

We compare the performance of our method to that of a number of previous KGE models, as shown in Table 3. The translation-based model (row 1), the compositional models utilizing basic mapping operations (rows 2-10) and the neural network-based models (rows 11-13) are among the first batch of baselines that rely only on triples seen in the KGs. The second batch of baselines are rule-based methods (rows 14-15). The last batch of baselines further incorporates logical rules (rows 16-26).

Table 3. Link prediction results on the test sets of FB15K and DB100K

	Method	FB15K				DB100K			
		MRR	HITS@1	HITS@3	HITS@10	MRR	HITS@1	HITS@3	HITS@10
1	TransE [22]	0.380	0.231	0.472	0.641	0.111	0.016	0.164	0.270
2	DistMult [25]	0.654	0.546	0.733	0.824	0.233	0.115	0.301	0.448
3	HolE [28]	0.524	0.402	0.613	0.739	0.260	0.182	0.309	0.4118
4	ComplEx [26]	0.627	0.550	0.671	0.766	0.272	0.218	0.303	0.362
5	ANALOGY [27]	0.725	0.646	0.785	0.854	0.252	0.143	0.323	0.427
6	ComplEx-NNE [44]	0.727	0.659	0.772	0.845	0.298	0.229	0.330	0.426
7	ComplEx-CAS [51]	-	-	-	0.866	-	-	-	-
8	RotatE [24]	0.664	0.551	0.751	0.841	0.327	0.200	0.417	0.526
9	HAKE[49]	0.408	0.312	0.463	0.579	-	-	-	-
10	CAKE[50]	0.741	0.646	0.825	0.896	-	-	-	-
11	R-GCN+ [29]	0.696	0.601	0.760	0.842	-	-	-	-
12	ConvE [30]	0.745	0.670	0.801	0.873	-			
13	DPMPN[45]	0.764	0.726	0.784	0.834	-	-	-	-
14	BLP [46]	0.242	0.151	0.269	0.424	-	-	-	-
15	MLN [21]	0.321	0.210	0.370	0.550	-	-	-	-
16	PTransE [47]	0.679	0.565	0.768	0.855	0.195	0.063	0.278	0.416
17	KALE [40]	0.518	0.382	0.606	0.756	0.249	0.100	0.346	0.497
18	ComplEx^R [48]	-	-	-	-	0.253	0.167	0.294	0.420
19	TARE [41]	0.781	0.617	0.728	0.842	-	-	-	-
20	RUGE [42]	0.768	0.703	0.815	0.865	0.246	0.129	0.325	0.433
21	ComplEx-NNE+AER [44]	0.801	0.757	0.829	0.873	0.311	0.249	0.339	0.426
22	IterE [16]	0.576	0.443	0.665	0.818	0.274	0.215	0.299	0.386
23	pLogicNet [43]	0.776	0.706	0.817	0.885	-	-	-	-
24	SoLE [17]	0.801	0.764	0.821	0.867	0.306	0.248	0.328	0.418
25	SLRE [18]	0.810	0.774	0.829	0.871	0.340	0.261	0.372	0.490
26	LogicENN [52]	0.766	-	-	0.874	-	-	-	-
	Iterlogic-E(ComplEx)	0.814	0.778	0.835	0.873	0.374	**0.301**	0.409	0.509
	Iterlogic-E(RotatE)	**0.837**	**0.794**	**0.868**	**0.904**	**0.387**	0.287	**0.449**	**0.559**

4.2.3. Implementation Details

On FB15K and DB100K, we compared Iterlogic-E against all of the baselines. We immediately obtained the results of a set of baselines on FB15K and DB100K from SoLR, SLRE and CAKE. We reimplemented ComplEx on the PyTorch framework based on thebased on the code code[3] supplied by [24] since our approach was dependent on it. Then, depending on our implementation, we provided the ComplEx result. Furthermore, the IterE result was tested on the sparse version of FB15K (FB15K-sparse) released by [16], which included only sparse entities with 18,544 and 22,013 triples in the validation and test sets. Therefore, we reimplemented IterE on FB15K and DB100K based on the code and hyperparameters[4] released by the author. As a result, we compared our approach to IterE and SoLE on the FB15K-sparse dataset. Both approaches use a logistic loss and optimize in the same way (SGD with AdaGrad). The other results of the baselines were obtained directly from prior literature. We tuned the embedding dimensionality d within {100, 150, 200, 250, 300}, the number of negatives per positive triple η within {2, 4, 6, 8, 10}, the initial learning rate γ within $\{10^{-4}, 10^{-3}, 10^{-2}, 5 \times 10^{-2}, 10^{-1}\}$, and the L2 regularization coefficient μ within $\{10^{-5}, 3 \times 10^{-5}, 10^{-4}, 10^{-3}, 3^{-3}, 10^{-2}\}$. We further tuned the margin within {0.1, 0.2, 0.5, 1, 2, 5, 12, 18, 24} for the approaches that utilize the margin-based ranking loss. The best hyperparameters were selected to maximize the MRR on the validation set. The best settings for Iterlogic-E were $d = 300$, $\gamma = 10^{-3}$, $\eta = 10$, and $\mu = 3 \times 10^{-5}$ on FB15K and $d = 300$, $\gamma = 10^{-4}$, $\eta = 10$, and $\mu = 10^{-4}$ on DB100K.

---

[3] https://github.com/DeepGraphLearning/KnowledgeGraphEmbedding
[4] https://github.com/wencolani/IterE

4.2.4. Main Results

The results of all compared methods on the test sets of FB15K, DB100K, and FB15K-sparse are shown in Tables 3 and 4. For each test triple, the mean reciprocal rank or H@N value with N = 1, 3, 10 is utilized as paired data. The experimental results show that 1) Iterlogic-E outperforms numerous strong baselines in the vast majority of cases. This shows that Iterlogic-E can achieve very good accuracy. 2) Iterlogic-E significantly outperforms the basic models that use triples alone, and the improvement comes from the ability to learn the conclusions obtained by soft rules. 3) Iterlogic-E also beats many baselines that incorporate logical rules. Specifically, Iterlogic-E performs better than SoLE and IterE under most metrics. This demonstrates the superiority of Iterlogic-E in reducing the noise of candidate conclusions. 4) IterE can only enhance sparse entities, so the experimental results are much lower than those of other baseline models. However, Iterlogic-E is also effective on FB15K-sparse. 5) On DB100K, the improvements over SLRE and SoLE are more significant than those on FB15K. The reason for this is probably that the groundings of the rules on DB100K contain more incorrect conclusions. Simple rules between a pair of relations are adequate to capture these simple patterns on the FB15K dataset. 6) The performance of Iterlogic-E(ComplEx) is worse than some baselines in HIT@10, and we consider that this limitation is mainly due to the shortcomings of the base model ComplEx [26]. Sun et al. [24] point out that ComplEx can not model the composition relation. We have experimented with replacing the base model with the RotatE model(Iterlogic-E(RotatE)), which is capable of modeling four relation patterns. The experimental results have been further improved, and our method consistently achieves the best results in all evaluation metrics.

Table 4. Link prediction results on the test sets of FB15K-sparse

Method	FB15K-sparse			
	MRR	HITS@1	HITS@3	HITS@10
TransE	0.398	0.258	0.486	0.645
DistMult	0.600	0.618	0.651	0.759
ComplEx	0.616	0.540	0.657	0.761
IterE	0.628	0.551	0.673	0.771
SoLE	0.668	0.604	0.699	0.794
Iterlogic-E(ComplEx)	**0.674**	**0.611**	**0.701**	**0.800**

4.2.5. Ablation Study

To explore the influence of different constraints and iterative learning, we perform an ablation study of Iterlogic-E on DB100K with 9 configurations in Table 5. The first and second variants, compared to the completed model Iterlogic-E, remove the non-negativity and l_2-norm constraints (l_2). The third setting is the combination setting. The fourth removes iterative learning (IL), which uses a rule to reason only once. The fifth, sixth, and seventh variants remove the additional loss item based on the sixth variant. The eighth setting is another variant of the Iterlogic-E model based on the ninth setting, which is that after the ComplEx fitting, according to the scores of the conclusions, the top n (where n is the rounded product of the rule and its confidence) conclusions of each rule are selected to be added to the KG to continue training. The ninth setting (AC) is to add all conclusions inferred from the rules as positive examples, and the tenth setting (WC) is to use the rule confidence as soft labels of conclusions.

Table 5. Ablation study

		DB100K			
		MRR	HITS@1	HITS@3	HITS@10
	Iterlogic-E	**0.374**	**0.301**	**0.409**	0.509
1	w/o l_2 on Θ	0.323	0.278	0.342	0.409
2	w/o NNE	0.371	0.295	0.409	0.509
3	1+2	0.324	0.279	0.342	0.410
4	w/o IL	0.372	0.300	0.407	0.505
5	w/o IL+L_{dc}	0.347	0.258	0.395	0.510
6	w/o IL+L_{rc}	0.351	0.257	0.406	**0.515**
7	w/o IL+L_{dc}+L_{rc}	0.328	0.279	0.348	0.422
8	Iterlogic-E*	0.369	0.298	0.404	0.502
9	ComplEx+AC+IL	0.285	0.223	0.312	0.407
10	ComplEx+WC+IL	0.295	0.231	0.327	0.422
	ComplEx	0.272	0.218	0.303	0.362

As seen in Table 5, we can conclude the following: 1) When removing NNE constraints, the performance of Iterlogic-E decreases slightly. Without l_2-norm constraints on entities, the performance of Iterlogic-E degrades by 2.3% in H@1 and by 5.1% in MRR. One explanation may be that l_2-norm constraints are sufficient to constrain embedding norms on DB100K. However, the performance will suffer dramatically if there are no l_2-norm constraints. 2) Removing iterative learning decreases performance slightly. One reason may be that the number of rules on DB100K is relatively small, so the number of conclusions added through iterative learning is relatively small. 3) Removing the additional loss item of the conclusions decreases performance slightly. This illustrates that Iterlogic-E can filter out incorrect conclusions and makes the KG dense. Surprisingly, even if we directly use ComplEx to filter and learn the conclusions that can achieve such high performance, this method is not as flexible as Iterlogic-E. 4) Compared with the basic model, all variants have different degrees of improvement. This demonstrates the critical importance of logical rules in link prediction tasks.

4.3. Influence of the Number of Iterations

To demonstrate how Iterlogic-E can enhance the embedding effect during the training process, we show the link prediction results on DB100K with different iterations. Figure 3 shows that as the number of training iterations increases, the prediction results, including Hit@1, Hit@3, Hit@10, and MRR, will improve. From Figure 3, we can infer the following: 1) Iterative learning enhances embedding learning since the quality of embeddings improved with time. 2) In the first two iterations, the embedding learning module was quickly fitted to the conclusions of the rules and the triples of the training set, and the prediction accuracy rapidly improved. 3) After two iterations, as the number of new conclusions decreased, the results of the inference tended to be stable, and the true conclusions and the initial KG triples were well preserved in the embedding.
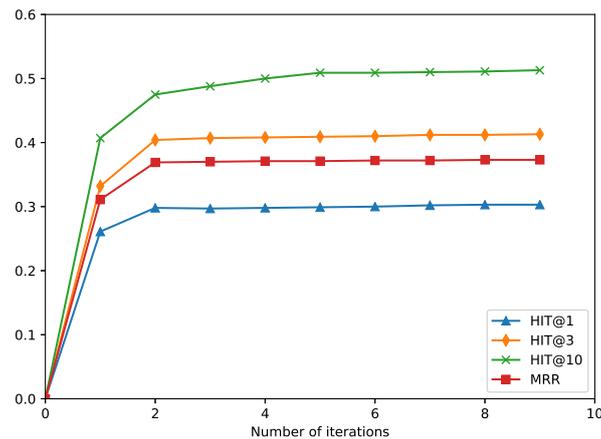
Figure 3. Link prediction results in different iterations

4.4. Influence of Confidence Levels

Additionally, we evaluate the effect of the rules' confidence thresholds on FB15K. Since there are different rules and ComplEx does not merge rules, we refer only to their fixed results on FB15K. We set all hyperparameters to their optimum values and change the confidence threshold within [0.5, 1] in 0.05-step increments. Both SoLE and Iterlogic-E use rules with confidence levels greater than this threshold. Figure 4 displays the MRR and H@1 values obtained by Iterlogic-E and other baselines on the FB15K test set. We make the following observations: 1) Iterlogic-E beats both ComplEx and IterE at varying confidence levels. This demonstrates that Iterlogic-E is sufficiently robust to deal with uncertain soft rules.
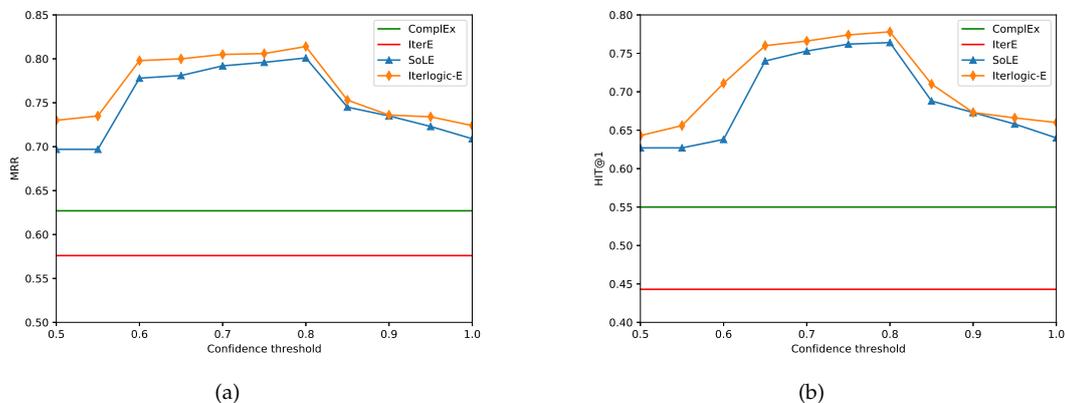

Figure 4. Results of MRR (**a**) and HITS@1 (**b**) achieved by Iterlogic-E with different confidence thresholds on FB15K.

4.5. Case Study

In Table 6, we present a case study with 4 conclusions (true or false as predicted by Iterlogic-E), which are inferred with 2 rules during training. Table 6 shows some conclusions derived from rule inference and the score change (the average of the head entity prediction score and the tail entity prediction score) of the conclusions. Using the first conclusion as an example, the true conclusion is (Albany_Devils,/hockey_roster_position/posit on, Centerman), which is obtained by the rule "/sports_team_roster/position$(x,y) \stackrel{0.808}{\Rightarrow}$ /hockey_roster _position/position(x,y)".

The Albany Devils are a professional ice hockey team in the American Hockey League [5], and the centerman is the center in ice hockey. Therefore, this is indeed a true conclusion. Compared to ComplEx, Iterlogic-E increased the score of this fact from 5.48 to 9.40. Also, (San_Diego_State_Aztecs_football, /hockey_roster_position/position,Linebacker) can be inferred from the fact (San_Diego_State_Aztecs_football, /sports_team_roster/team, Linebacker) by the same rule. However, the San Diego State Aztecs are a football team, not a hockey team, and this is an incorrect conclusion. Compared to ComplEx, Iterlogic-E decreased the score of this fact from -4.65 to -8.35. This illustrates that Iterlogic-E can distinguish whether the conclusion is true and can improve the prediction performance. Furthermore, Iterlogic-E has good interpretability, and we can understand why the conclusion is inferred by it.

Table 6. A case study with 4 conclusions (true or false as predicted by Iterlogic-E), which are reasoned on by 2 rules.

	True	False
Conclusions	(Albany_Devils,/hockey _roster_position/ position,Centerman)	(San_Diego_State_Aztecs_football, /hockey_roster_position/position, Linebacker)
Predicted by rules		/sports_team_roster /position$(x,y) \overset{0.808}{\Rightarrow}$ /hockey_roster_position/position(x,y)
Score change	5.4888 → 9.4039	-4.6518 → -8.3547
Conclusions	(Chris_Nurse,/football_roster_position/ team,Stevenage_F.C.)	(Brett_Favre,/football_roster_position/ team,Green_Bay_Packers)
Predicted by rules		/sports_team_roster/ team$(x,y) \overset{0.847}{\Rightarrow}$ /football_roster_position/team(x,y)
Score change	5.8123 → 9.5991	-3.0131 → -11.3952

5. Conclusion and Future Work

This paper proposes a novel framework that iteratively learns logical rules and embeddings, which models the conclusion labels as 0-1 variables. The proposed Iterlogic-E uses the confidences of rules and the context of the KG to eliminate the uncertainty of the conclusion in the stage of learning embeddings. Specifically, our method is based on iterative learning, which not only supplements conclusions but also filters incorrect conclusions, resulting in a good balance between efficiency and scalability. The evaluation on benchmark KGs demonstrates that the method can learn correct conclusions and improve against a variety of strong baselines. In the future, we would like to explore how to use embeddings to learn better rules and rule confidences than AMIE+. Additionally, we will continuously explore more advanced models to integrate rules and KGEs for knowledge reasoning.

References

1. Berant J, Chou A, Frostig R, Liang P. Semantic parsing on freebase from question-answer pairs. In *EMNLP*, 2013, pp. 1533–1544.
2. Huang X, Zhang J, Li D, Li P. Knowledge graph embedding based question answering. In *WSDM*, 2019, pp. 105–113.
3. Wang X, Wang D, Xu C, He X, Cao Y, Chua T S. Explainable reasoning over knowledge graphs for recommendation. In *AAAI*, 01, 2019, pp. 5329–5336.
4. Cao Y, Wang X, He X, Hu Z, Chua T S. Unifying knowledge graph learning and recommendation: Towards a better understanding of user preferences. In *WWW*, 2019, pp. 151–161.
5. Auer S, Bizer C, Kobilarov G, Lehmann J, Cyganiak R, Ives Z. Dbpedia: A nucleus for a web of open data. In *SEMANT WEB*, 2007, pp. 722–735.
6. Bollacker K, Evans C, Paritosh P, Sturge T, Taylor J. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD*, 2008, pp. 1247–1250.
7. Carlson A, Betteridge J, Kisiel B, Settles B, Hruschka E, Mitchell T. Toward an architecture for never-ending language learning. In *AAAI*, number 1, 2010.
8. Ji S, Pan S, Cambria E, Marttinen P, Yu P S. A survey on knowledge graphs: Representation, acquisition and applications. arXiv:2002.00388, 2020.

---

5   https://en.wikipedia.org/wiki/Albany_Devils

9. Taskar B, Abbeel P, Wong M F, Koller D. Relational markov networks. *Introduction to Statistical Relational Learning*, 2007, pp. 175–200.
10. Galárraga L A, Teflioudi C, Hose K, Suchanek F. Amie: association rule mining under incomplete evidence in ontological knowledge bases. In *WWW*, 2013, pp. 413–422.
11. Giarratano J C, Riley G. *Expert Systems*. MA, United States: PWS Publishing, 1998.
12. Jackson P. *Introduction to Expert Systems*. Boston: Addison-Wesley Longman Publishing, 1986.
13. Nickel M, Murphy K, Tresp V, Gabrilovich E. A review of relational machine learning for knowledge graphs. *Proc. IEEE*, 2016, 104(1):11–33.
14. Wang Q, Mao Z, Wang B, Guo L. Knowledge graph embedding: A survey of approaches and applications. *IEEE Trans. Knowl. Data*, 2017, 29(12):2724–2743.
15. Pujara J, Augustine E, Getoor L. Sparsity and noise: Where knowledge graph embeddings fall short. In *EMNLP*, 2017, pp. 1751–1756.
16. Zhang W, Paudel B, Wang L, Chen J, Zhu H, Zhang W, Bernstein A, Chen H. Iteratively learning embeddings and rules for knowledge graph reasoning. In *WWW*, 2019, pp. 2366–2377.
17. Zhang J, Li J. Enhanced knowledge graph embedding by jointly learning soft rules and facts. *Algorithms*, 2019, (12):265.
18. Guo S, Li L, Hui Z, Meng L, Ma B, Liu W, Wang L, Zhai H, Zhang H. Knowledge graph embedding preserving soft logical regularity. In *CIKM*, 2020, pp. 425–434.
19. Hájek P. *Metamathematics of Fuzzy Logic*, volume 4 of *Trends in Logic*. Boston: Kluwer, 1998.
20. Galárraga L, Teflioudi C, Hose K, Suchanek F M. Fast rule mining in ontological knowledge bases with AMIE+. *VLDB*, 2015, (6):707–730.
21. Richardson M, Domingos P. Markov logic networks. *Mach. Learn.*, 2006, (1–2):107–136.
22. Bordes A, Usunier N, Garcia-Duran A, Weston J, Yakhnenko O. Translating embeddings for modeling multi-relational data. In *NeurIPS*, 2013, pp. 1–9.
23. Yang S, Tian J, Zhang H, Yan J, He H, Jin Y. Transms: Knowledge graph embedding for complex relations by multidirectional semantics. In *IJCAI*, 2019, pp. 1935–1942.
24. Sun Z, Deng Z H, Nie J Y, Tang J. Rotate: Knowledge graph embedding by relational rotation in complex space. In *ICLR*, 2019.
25. Yang B, Yih W, He X, Gao J, Deng L. Embedding entities and relations for learning and inference in knowledge bases. In Bengio Y, LeCun Y, editors, *ICLR*, 2015.
26. Trouillon T, Welbl J, Riedel S, Gaussier É, Bouchard G. Complex embeddings for simple link prediction. In *ICML*, 2016, pp. 2071–2080.
27. Liu H, Wu Y, Yang Y. Analogical inference for multi-relational embeddings. In *ICML*, 2017, pp. 2168–2178.
28. Nickel M, Rosasco L, Poggio T. Holographic embeddings of knowledge graphs. In *AAAI*, number 1, 2016.
29. Schlichtkrull M, Kipf T N, Bloem P, Van Den Berg R, Titov I, Welling M. Modeling relational data with graph convolutional networks. In *ESWC*, 2018, pp. 593–607.
30. Dettmers T, Minervini P, Stenetorp P, Riedel S. Convolutional 2d knowledge graph embeddings. In *AAAI*, number 1, 2018.
31. Guo L, Sun Z, Hu W. Learning to exploit long-term relational dependencies in knowledge graphs. In *ICML*, 2019, pp. 2505–2514.
32. Shang C, Tang Y, Huang J, Bi J, He X, Zhou B. End-to-end structure-aware convolutional networks for knowledge base completion. In *AAAI*, number 01, 2019, pp. 3060–3067.
33. Shi B, Weninger T. Proje: Embedding projection for knowledge graph completion. In *AAAI*, number 1, 2017.
34. Wang Q, Huang P, Wang H, Dai S, Jiang W, Liu J, Lyu Y, Zhu Y, Wu H. Coke: Contextualized knowledge graph embedding. preprint, arXiv:1911.02168, 2019.
35. Guo S, Wang Q, Wang B, Wang L, Guo L. Semantically smooth knowledge graph embedding. In *ACL-IJCNLP*, 2015, pp. 84–94.
36. Xie R, Liu Z, Sun M. Representation learning of knowledge graphs with hierarchical types. In *IJCAI*, 2016, pp. 2965–2971.
37. Hao J, Chen M, Yu W, Sun Y, Wang W. Universal representation learning of knowledge bases by jointly embedding instances and ontological concepts. In *SIGKDD*, 2019, pp. 1709–1719.
38. Fatemi B, Ravanbakhsh S, Poole D. Improved knowledge graph embedding using background taxonomic information. In *AAAI*, number 01, 2019, pp. 3526–3533.

39. Veira N, Keng B, Padmanabhan K, Veneris A G. Unsupervised embedding enhancements of knowledge graphs using textual associations. In *IJCAI*, 2019, pp. 5218–5225.

40. Guo S, Wang Q, Wang L, Wang B, Guo L. Jointly embedding knowledge graphs and logical rules. In *EMNLP*, 2016, pp. 192–202.

41. Wang M, Rong E, Zhuo H, Zhu H. Embedding knowledge graphs based on transitivity and asymmetry of rules. In *PAKDD*, 2018, pp. 141–153.

42. Guo S, Wang Q, Wang L, Wang B, Guo L. Knowledge graph embedding with iterative guidance from soft rules. In *AAAI*, number 1, 2018.

43. Qu M, Tang J. Probabilistic logic neural networks for reasoning. In *NeurIPS*, 2019.

44. Ding B, Wang Q, Wang B, Guo L. Improving knowledge graph embedding using simple constraints. In *ACL*, 2018, pp. 110–121.

45. Xu X, Feng W, Jiang Y, Xie X, Sun Z, Deng Z. Dynamically pruned message passing networks for large-scale knowledge graph reasoning. In *ICLR*, 2020.

46. De Raedt L, Kersting K. Probabilistic inductive logic programming. In *Probabilistic Inductive Logic Programming*, pp. 1–27. 2008.

47. Lin Y, Liu Z, Luan H, Sun M, Rao S, Liu S. Modeling relation paths for representation learning of knowledge bases. In *EMNLP*, pp. 705–714.

48. Minervini P, Costabello L, Muñoz E, Novácek V, Vandenbussche P. Regularizing knowledge graph embeddings via equivalence and inversion axioms. In *ECML-PKDD*, 2017, pp. 668–683.

49. Zhang Z, Cai J, Zhang Y, Wang J. Learning hierarchy-aware knowledge graph embeddings for link prediction. In *AAAI*, 2020, pp. 3065–3072.

50. Niu G, Li B, Zhang Y, Pu S. CAKE: A Scalable Commonsense-Aware Framework For Multi-View Knowledge Graph Completion. In *ACL*, 2022, pp. 2867–2877.

51. Yang J, Ying X, Shi Y, Tong X, Wang R, Chen T, Xing B. Knowledge Graph Embedding by Adaptive Limit Scoring Loss Using Dynamic Weighting Strategy. In *ACL*, 2022, pp. 1153–1163.

52. Nayyeri M, Xu C, Alam M M, Yazdi H S. LogicENN: A Neural Based Knowledge Graphs Embedding Model With Logical Rules. *IEEE TPAMI*, 2023, 45(6): 7050-7062.