Simple Rule Injection for ComplEx Embeddings

Anonymous Author(s)

ABSTRACT

Recent works in neural knowledge graph inference attempt to combine logic rules with knowledge graph embeddings to benefit from prior knowledge. However, they usually cannot avoid rule grounding, and injecting a diverse set of rules has still not been thoroughly explored. In this work, we propose InjEx, a mechanism to inject multiple types of rules through simple constraints, which capture definite Horn rules. To start, we theoretically prove that InjEx can inject such rules. Next, to demonstrate that InjEx infuses interpretable prior knowledge into the embedding space, we evaluate InjEx on both the knowledge graph completion (KGC) and few-shot knowledge graph completion (FKGC) settings. Our experimental results reveal that InjEx outperforms both baseline KGC models as well as specialized few-shot models while maintaining its scalability and efficiency.

CCS CONCEPTS

• **Computing methodologies** → **Knowledge representation and reasoning**; **Reasoning about belief and knowledge**.

KEYWORDS

Knowledge graph embeddings, link prediction, knowledge distillation, knowledge graph

ACM Reference Format:

Anonymous Author(s). 2023. Simple Rule Injection for ComplEx Embeddings. In *Proceedings of 30th SIGKDD Conference on Knowledge Discovery and Data Mining (SIGKDD '24).* ACM, New York, NY, USA, 10 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Knowledge Graphs (KGs), which are structured as a set of triples linking head entities to tail entities via relationships, serve as the foundation for numerous real-world applications such as question answering, recommender systems, and natural language processing, exemplified by Freebase [2], Yago [29] and NELL [5]. These repositories, consisting of millions of facts, play a crucial role in the underlying mechanisms of these applications.

The immense amount of information stored in today's large-scale KGs can be used to infer new knowledge directly from the KG. Currently, the two prominent approaches to achieve this are representation learning and pattern mining over KGs. **KG embedding (KGE) models** aim to represent entities and relations as low-dimensional vectors, such that the semantic meaning is captured. This approach has been well studied in the past decade [3, 16, 18, 35, 37]. However, these approaches still lack interpretability and face challenges with unseen entities.

Other KGE works focus on leveraging more complex triple scoring models [32] or using meta learning [6, 19, 33, 36], while possible prior knowledge is relatively ignored. Few works that study injecting rules into embeddings do so for very limited types of rules [9, 14]. We show that one can use simple constraints on the embeddings and objective function to inject a variety of rule types.

On the other hand, **KG rule mining** techniques extract information in the form of human-understandable logical rules. For example, AMIE [11], DRUM [27], and AnyBURL [21] discover and mine meaningful symbolic rules from the background KGs. Specifically, AMIE [11] states composition rules are the most common and important ones among all the rules. Based on those, rule-based works like SAFRAN [24] focus on predicting missing links between entities with certain types of rules. A major bottleneck of the currently used approaches is the relatively lower predictive performance compared to KG representation learning methods.

Recently there have been attempts at combining embedding-based and rule-based methods to achieve both higher performance and better interpretability, for example, ComplEx-NNE [9] and UniKER [7]. In addition, rule injection methods provide a natural way of including prior knowledge into representation learning techniques. Nevertheless, previous attempts using a probabilistic model to approximate the exact logical inference [25, 26, 40] require grounding of rules which is intractable in large-scale real-world knowledge graphs. Other works that treat logical rules as additional constraints into KGE [8, 12] usually deal with certain types of rules or do not explore the effect between different types of rules.

In this work, we propose InjEx, a novel method of rule injection that improves the reasoning performance and provides the ability of soft injection of prior knowledge via multiple different types of rules without grounding (see Table 1). Following the idea of ComplEx-NNE [9], we propose that with proper constraints on entity and relation embeddings, we are able to handle definite Horn rules with ComplEx as the base model. First, we impose a non-negative bounded constraint on both entity and relations embeddings. Second, we add a simple yet novel regularization to the KG embedding objective that enforces the rules' constraints. As we will explain in more detail in Section 4.2, the former guarantees that the base model is able to inject multiple types of patterns, and the latter encodes the connections between relations, which helps the model learn more predictive embeddings.

To demonstrate the effectiveness of InjEx in incorporating prior knowledge, we also evaluate InjEx on multiple widely used benchmarks for link prediction. We show that InjEx achieves, or is on par with, state-of-the-art models on all the benchmarks and multiple evaluation settings.

Our contributions can be summarized as follows:

- We propose a novel model, InjEx, that allows the integration of prior knowledge in the form of definite Horn rules

SIGKDD '24, August 25 - August 29, 2024, Barcelona, Spain

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/XXXXXXX.XXXXXXX

into ComplEx KG embeddings. InjEx only requires minimal modifications of the underlying embedding method which avoids rule grounding and enables soft rule injection.
- Unlike prior methods [9], InjEx is not limited to a specific rule structure. We show that InjEx is able to capture definite Horn rules with any body length and empirically outperforms state-of-the-art models on knowledge graph completion (KGC) tasks. InjEx improves around 5% in Hits@10 on both FB15k-237 and YAGO3-10 compared with its base model ComplEx. We also show the effect of the combination or separately injecting multi-length definite Horn rules on KGC tasks.
- We show that, on a few-shot link prediction task, InjEx still successfully captures the injected prior knowledge and is able to achieve competitive performance against other baselines on such task.

Table 1: Capabilities of different methods.

	Multi-length definite Horn rules	Soft rule injection	Avoid Grounding
Demeester et al.[8]	✗	✗	✓
ComplEx-NNE[9]	✗	✓	✓
KALE[12]	✓	✗	✗
RUGE[13]	✓	✓	✗
RNNLogic[25]	✓	✓	✗
pLogicNet[26]	✓	✓	✗
UniKER[7]	✓	✓	✗
InjEx	✓	✓	✓

2 RELATED WORKS

In this section, we give an overview of embedding models for KGC and few-shot link prediction, rule mining systems, and constraint/rule-assisted link prediction models.

2.1 Knowledge Graph Embedding (KGE) Models

KG embedding models can be generally classified into translation and bilinear models. The representative of translation models is TransE [3], which models the relations between entities as the difference between their embeddings. This method is effective in inferencing composition, anti-symmetry and inversion patterns, but can't handle the 1-to-N, N-to-1, and N-N relations. RotatE [30] models relations as rotations in complex space so that symmetric relations can be captured, but is as limited as TransE otherwise. Other models such as BoxE [1], HAKE [42], and DiriE [32] are able to express multiple types of relation patterns with complex KG embeddings. MulDE [33] proposes to transfer knowledge from multiple teacher KGE models to perform better on KGC tasks. ComplEx [31], as a representative of bilinear models, introduces a diagonal matrix with complex numbers to capture anti-symmetry.

2.2 Few-shot Knowledge Graph Completion

Few-shot KGC refers to the scenario where only a limited number of instances are provided for certain relations or entities, which means the model needs to leverage knowledge about other relations or entities to predict the missing link. In this work, we are mainly concerned about few-shot learning for new relations[4, 6, 15]. One of the early works in this direction [36], leverages a neighbor encoder to learn entity embeddings and a matching component to find similar reference triples to the query triple.

2.3 Rule-based Models

Previous works such as AMIE [11] and DRUM [27] mine logical rules to predict novel links in KGs. SAFRAN [24] and AnyBURL [21] share a similar method to predict links with logical rules. Similar to InjEx, ComplE-NNE[9] approximately applies entailment patterns as constraints on relation representations to improve the KG embeddings. However, it can only inject entailment rules. Another way is to augment knowledge graphs with grounding of rules which is less efficient for large-scale KGs. Representatives like RUGE [13] and KALE [12] treat logical rules as additional regularization by computing satisfaction scores to sample ground rules. IterE [39] [98] proposes an iterative training strategy with three components of embedding learning, axiom induction, and axiom injection, targeting at sparse entity reasoning. UniKER [7] augments triplets from relation path rules to improve embedding quality. But to avoid data noise, it uses only a small number of relation path rules and requires multiple passes of augmenting data and model training.

2.4 Graph Neural Network Models

The Graph Neural Network (GNN) has gained wide attention on KGC tasks in recent years [34, 38, 43]. With the high expressiveness of GNNs, these methods have shown promising performance. However, SOTA GNN-based models do not show great advantages compared with KGE models while introducing additional computational complexity [43]. For example, RED-GNN [41] achieves competitive performance on KGC benchmarks, but the leverage of the Bellman-Ford algorithm which needs to propagate through the whole knowledge graph, restricts their application on large graphs. Several methods including pLogicNet [26] propose using Markov Logic Network (MLN) to compute variational distribution over possible hidden triples for logic reasoning. RNNLogic [25] learns logic rules for knowledge graph reasoning with EM-based algorithm in reinforcement learning.

3 PRELIMINARIES

3.1 Knowledge Graphs (KGs)

Let $\mathcal{E}$ and $\mathcal{R}$ denote the set of entities and relations, a knowledge graph $\mathcal{G} = \{(e_i, r_k, e_j)\} \subset \mathcal{E}\times\mathcal{R}\times\mathcal{E}$ is a collection of factual triples, where e_i and r_k are the i-th entity and k-th relation, respectively. We usually refer to e_i and e_j as the head and tail entity. A knowledge graph can also be represented as $\mathcal{X} \in \{0,1\}^{|\mathcal{E}|\times|\mathcal{R}|\times|\mathcal{E}|}$, which is called the adjacency tensor of $\mathcal{G}$. The (i, j, k) entry $\mathcal{X}_{i,k,j} = 1$ when triple (e_i, r_k, e_j) is true, otherwise $\mathcal{X}_{i,k,j} = 0$.

3.2 Knowledge Graph Completion (KGC) and ComplEx

3.2.1 ***Knowledge Graph Completion.*** The objective of KGC is to predict valid but unobserved triples in $\mathcal{G}$. Formally, given a head

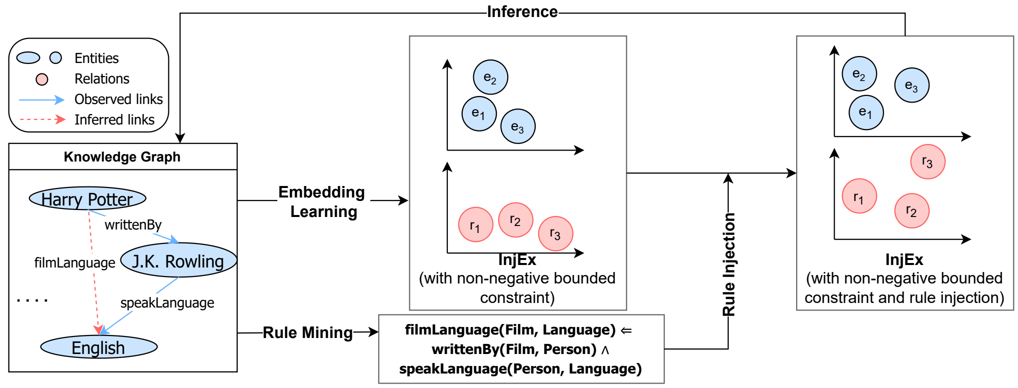


Figure 1: Overview of the InjEx model. Note that we illustrate injecting a composition rule.

entity e_i (tail entity e_j) with a relation r_k, models are expected to find the tail entity e_j (head entity e_i) to form the most plausible triple (e_i, r_k, e_j) in $\mathcal{G}$. KGC models usually define a scoring function $f : \mathcal{E} \times \mathcal{R} \times \mathcal{E} \rightarrow \mathbb{R}$ to assign a score $s(e_i, r_k, e_j)$ to each triple $(e_i, r_k, e_j) \in \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ which indicates the plausibility of the triple.

KGE models usually associate each entity e_i and relation r_j with vector representations $\mathbf{e}_i$, $\mathbf{r}_j$ in the embedding space. Then they define a scoring function to model the interactions among entities and relations. We review ComplEx, which is used as our base model.

3.2.2 ***ComplEx***. ComplEx [31] models entities $e \in \mathcal{E}$ and relations $r \in \mathcal{R}$ as complex-valued vectors $\mathbf{e}, \mathbf{r} \in \mathbb{C}^d$ where d is the embedding space dimension. For each triple $(e_i, r_k, r_j) \in \mathcal{E} \times \mathcal{R} \times \mathcal{E}$, the scoring function is defined as:

$$\begin{aligned}\phi(e_i, r_k, e_j) &:= Re(\langle \mathbf{e}_i, \mathbf{r}_k, \overline{\mathbf{e}}_j \rangle) \\ &:= Re(\sum_l [\mathbf{e}_i]_l [\mathbf{r}_k]_l [\overline{\mathbf{e}}_j]_l) \end{aligned} \quad (1)$$

where $\mathbf{e}_i, \mathbf{r}_k, \mathbf{e}_j \in \mathbb{C}^d$ are the vector representations associated with e_i, r_k, r_j and $\overline{\mathbf{e}}_j$ is the conjugate of $\mathbf{e}_j$. Triples with higher $\phi(\cdot, \cdot, \cdot)$ scores are more likely to be true. We use $\langle \mathbf{a}, \mathbf{b}, \mathbf{c} \rangle$ or $\mathbf{a} \times \mathbf{b}$ for element-wise vector multiplication. For a complex number $x \in \mathbb{C}$, $Re(\cdot), Im(\cdot)$ means taking the real, imaginary components of x.

3.3 Definite Horn Rules

Horn rules, as a popular subset of first-order logic rules, can be automatically extracted from recent rule mining systems, such as AMIE [11] and AnyBurl [21]. Length k Horn rules are usually written in the form of implication as below:

$$\forall x, y, r_1(x, z_1) \wedge r_2(z_1, z_2) \wedge ...r_k(z_{k-1}, y) \Rightarrow r(x, y)$$

where $r(x, y)$ is called the head of the rule and $r_1(x, z_1) \wedge r_2(z_1, z_2) \wedge ...r_k(z_{k-1}, y)$ is the body of the rule. In KGs, a ground Horn rule is then represented as:

$$r_1(e_i, e_{p_1}) \wedge r_2(e_{p_1}, e_{p_2}) \wedge ...r_k(e_{p_{k-1}}, e_j) \Rightarrow r(e_i, e_j)$$

For convenience, we define length-1 and length-2 definite Horn rules as:

Definition 1 (Hierarchy rule)**.** A **hierarchy** rule holds between relation r_1 and r_2 if $\forall e_i, e_j$

$$r_1(e_i, e_j) \Rightarrow r_2(e_i, e_j)$$

Definition 2 (Composition rule)**.** A **composition** rule holds between relation r_1, r_2 and r_3 if $\forall e_i, e_j, e_k$

$$r_1(e_i, e_j) \wedge r_2(e_j, e_k) \Rightarrow r_3(e_i, e_k)$$

4 METHEDOLOGY

Here, we first introduce how the non-negative constraint over both entity and relation embeddings helps make the model fully expressive (4.1). Then we discuss how we integrate Horn rules into the base model as a rule-based regularization (4.2). Figure 1 illustrates an overview of the InjEx model for composition rules.

4.1 Non-negative Bounded Constraint on Entity and Relation Representations

We modify ComplEx further by requiring both entity and relation representations to be non-negative and bounded. More formally we require entity and relation embeddings to satisfy:

$$\begin{aligned} &0 \leq Re(\mathbf{e}), Im(\mathbf{e}) \leq 1, \forall e \in \mathcal{E} \\ &0 \leq Re(\mathbf{r}), Im(\mathbf{r}), 0 \leq |\mathbf{r}| \leq R, \forall r \in \mathcal{R} \end{aligned} \quad (2)$$

where $R \in \mathbb{R}$ is a selected upper bound for the norm of relation representations, and entity representations stay within the hypercube of $[0, 1]^d$.

As pointed out by [23], the positive elements of entities or relations usually contain enough information. Thus, intuitively we don't expect these constraints to significantly affect the performance of ComplEx. While they provide additional benefits such as i) guarantee that the original scoring function of [31] is bounded. ii) as we demonstrate in theorem 1, with the constraints above ComplEx can now also infer composition patterns.

4.2 Rule Injection

In this section, we discuss how Horn rules are integrated into ComplEx as a rule-based regularization. We start with the injection of composition rules, extending the discussion to length-k Horn rules.

We treat each dimension of an entity as a separate attribute. Recall the equation 1, for the simplicity of notations, we decompose

$$
\begin{aligned}
&\phi(e_i, r_k, e_j) =: \Sigma_l(\varphi_l(e_i, r_k, e_j)) \\
&\varphi_l(e_i, r_k, e_j) := Re([e_i]_l [r_k]_l [\bar{e}_j]_l) \\
&\qquad = (|[e_i]_l||[r_k]_l||[e_j]_l| \cos(\theta_{[r_k]_l} + \theta_{[e_i]_l} - \theta_{[e_j]_l}))
\end{aligned} \tag{3}
$$

s.t. $\varphi_l(e_i, r_k, e_j)$ denotes the score of dimension l in relation r_k.

Composition Rule Injection. We now introduce how we inject composition rules. As in definition 2, composition rules express one type of relation between three relations. For example, `BornIn`$(x, y) \wedge$ `CountryOfOrigin`$(y, z) \rightarrow$ `Nationality`(x, z) indicates the country of the city in which a person is born should usually be the nationality of that person. We denote such composition rules with confidence level λ as

$$r_1(x, y) \wedge r_2(y, z) \xrightarrow{\lambda} r_3(x, z)$$

As defined in equation 3, given the range constraints, we have $\varphi_l(x, r, y)/(2R) \in [0, 1]$, so we can treat $\varphi_l/(2R)$ as the probability that the relation holds for the triple's attribute l.

Under such assumption, we now propose we can roughly model a strict composition rule $r_1(e_i, e_j) \wedge r_2(e_j, e_k) \xrightarrow{\infty} r_3(e_i, e_k)$ as follows.

Theorem 1. *A strict composition rule $r_1(e_i, e_j) \wedge r_2(e_j, e_k) \xrightarrow{\infty} r_3(e_i, e_k)$ holds if the following condition is satisfied.*

$$
\begin{aligned}
&\varphi_l(e_i, r_1, e_j) * \varphi_l(e_j, r_2, e_k)/(2R) \le \varphi_l(e_i, r_3, e_k) \\
&\qquad \forall e_i, e_j, e_k \in \mathcal{E}, \forall 1 \le l \le d
\end{aligned} \tag{4}
$$

where d is the dimension of e and r, and R is the range of all relation embeddings as defined in Section 4.1. (See proof in Appendix A)

Note that with the non-negative bounded constraint still, a sufficient condition for Equation 4 to hold, is to further impose

$$
\begin{aligned}
&Re(\mathbf{r}_1 \times \mathbf{r}_2)/R \le Re(\mathbf{r}_3), \\
&Im(\mathbf{r}_1 \times \mathbf{r}_2)/R = Im(\mathbf{r}_3)
\end{aligned} \tag{5}
$$

where $\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3$ are the vectorized representation of r_1, r_2 and r_3. We further introduce the level of confidence and slackness in Equation 5 to model approximate composition rules, which yields

$$
\begin{aligned}
&\lambda(Re(\mathbf{r}_1 \times \mathbf{r}_2)/R^2 - Re(\mathbf{r}_3)/R) \le \alpha, \\
&\lambda(Im(\mathbf{r}_1 \times \mathbf{r}_2)/R^2 - Im(\mathbf{r}_3)/R)^2 \le \beta
\end{aligned} \tag{6}
$$

Definite Horn Rule Injection. We now introduce how we inject Horn rules based on previous observations. As defined in Sec3.3, length-k Horn rules represent one type of relationship between $k + 1$ relations. For example, `FilmWrittenBy`$(x, y) \rightarrow$ `FilmDirectedBy`(x, y) indicates that a person who is the writer of a movie should probably also direct that movie; $ActorOfFilm(x, z) \wedge nationality(z, y) \rightarrow FilmReleaseRegion(x, y)$ shows a more complex relation between the actor, her nationality and the region where the film is released. We further denote Horn rule with confidence level λ as

$$r_1(x, z_1) \wedge r_2(z_1, z_2) \wedge ... \wedge r_k(z_{k-1}, y) \xrightarrow{\lambda} r(x, y)$$

$$or\ r_1 \wedge r_2 \wedge ... \wedge r_k \xrightarrow{\lambda} r$$

With the same assumption as in composition rule injection, we can further model a strict Horn rule $r_1(x, z_1) \wedge r_2(z_1, z_2) \wedge ... \wedge r_k(z_{k-1}, y) \xrightarrow{\infty} r(x, y)$ as follows:

Theorem 2. *A strict Horn rule*

$$r_1(x, z_1) \wedge r_2(z_1, z_2) \wedge ... \wedge r_k(z_{k-1}, y) \xrightarrow{\infty} r(x, y)$$

holds if the following condition is satisfied:

$$
\begin{aligned}
&\Pi_{i=1}^{k}(\varphi_{i,l}(z_{i-1}, r_i, z_i)/R) \le (\varphi_l(x, r, y)/R), \\
&\qquad \forall e_i, e_j, e_{p_1}, ..., e_{p_{k-1}} \in \mathcal{E}, \forall 1 \le l \le d,
\end{aligned} \tag{7}
$$

where $x = z_0$, $y = z_k$, R is still the range of all relation embeddings as defined in Section 4.1.

With the non-negative bounded constraint in 4.1, a sufficient condition for Equation 7 to hold, is to further impose

$$
\begin{aligned}
&Re(\mathbf{r}_1 \times \mathbf{r}_2 \times ... \times \mathbf{r}_k)/R^k \le Re(\mathbf{r})/R, \\
&Im(\mathbf{r}_1 \times \mathbf{r}_2 \times ... \times \mathbf{r}_k)/R^k = Im(\mathbf{r})/R
\end{aligned} \tag{8}
$$

where $\mathbf{r}_i$ is the vectorized representation of r_i. We further introduce the level of confidence and slackness in Equation 8 to model approximate composition rules, which yields

$$
\begin{aligned}
&\lambda(Re(\mathbf{r}_1 \times \mathbf{r}_2 \times ... \times \mathbf{r}_k)/R^k - Re(\mathbf{r})/R) \le \alpha, \\
&\lambda(Im(\mathbf{r}_1 \times \mathbf{r}_2 \times ... \times \mathbf{r}_k)/R^k - Re(\mathbf{r})/R)^2 \le \beta
\end{aligned} \tag{9}
$$

4.3 Optimization Objective

We combine the basic embedding model of ComplEx, the non-negative bounded constraint on both entity and relation representations, and the approximate injection of Horn rules as InjEx. The overall model is presented as follows:

$$
\begin{aligned}
\min_{\Theta,\{\alpha,\beta\}} &\sum_{\mathcal{D}+\cup\mathcal{D}^-} log(1 + exp(-y_{ijk}\phi(e_i, r_k, e_j))) \\
&+ \mu \sum_{\mathcal{T}} \mathbf{1}^\top(\alpha + \beta) + \eta||\Theta||_3^3, \\
s.t.\ &\lambda(Re(\mathbf{r}_1 \times ... \times \mathbf{r}_k)/R^k - Re(\mathbf{r})/R) \le \alpha, \\
&\lambda(Im(\mathbf{r}_1 \times ... \times \mathbf{r}_k)/R^k - Re(\mathbf{r})/R)^2 \le \beta, \\
&\alpha, \beta \ge 0, \\
&\forall r_1 \wedge ... \wedge r_k \xrightarrow{\lambda} r \in \mathcal{T} \\
&0 \le Re(\mathbf{e}), Im(\mathbf{e}) \le 1, \forall e \in \mathcal{E} \\
&0 \le Re(\mathbf{r}), Im(\mathbf{r}), 0 \le |\mathbf{r}| \le R, \forall r \in \mathcal{R}
\end{aligned} \tag{10}
$$

Here, $\Theta := \{\mathbf{e} : e \in \mathbb{E}\} \cup \{\mathbf{r} : r \in \mathbb{R}\}$ is the set of all entity and relation representations; $\mathcal{D}+$ and $\mathcal{D}^-$ are the sets of positive and negative training triples respectively; $\mathcal{T}$ is the rule set of Horn rules. The first term of the objective function is a typical logistic loss as in ComplEx. The second term is the sum of slack variables, used to inject Horn rules with penalty coefficient $\mu \ge 0$. We encourage the slackness to be as small as possible to better satisfy the rules. The last term is N3 regularization [17] to avoid overfitting.

To solve the optimization problem, we convert the slack variable for Horn rules into penalty terms added to the original objective function, and rewrite Equation 10 as:

$$\begin{aligned}\min_{\Theta,\{\alpha_1,\beta_1,\alpha_2,\beta_2\}} &\sum_{\mathcal{D}+\cup\mathcal{D}^-} log(1+exp(-y_{ijk}\phi(e_i,r_k,e_j)))\\ &+\mu\sum_{\mathcal{T}}\lambda\mathbf{1}^\top[Re(\mathbf{r}_1\times...\times\mathbf{r}_k)/R^k-Re(\mathbf{r})/R]_+\\ &+\mu\sum_{\mathcal{T}}\lambda\mathbf{1}^\top(Im(\mathbf{r}_1\times...\times\mathbf{r}_k)/R^k-Re(\mathbf{r})/R))^2\\ &+\eta||\Theta||_3^3,\\ s.t.\ &0\le Re(\mathbf{e}),Im(\mathbf{e})\le 1,\forall e\in\mathcal{E}\\ &0\le Re(\mathbf{r}),Im(\mathbf{r}),0\le|\mathbf{r}|\le R,\forall r\in\mathcal{R}\end{aligned}\tag{11}$$

where $[\mathbf{v}]_+ = max(\mathbf{0},\mathbf{v})$ and $max(\cdot,\cdot)$ is an element-wise operation, and λ is the confidence of a rule. We use AdaGrad [10] as our optimizer. After each gradient descent step, we project entity and relation representations into $[0,1]^d$ and $[0,R]^d$, respectively. When training, we set the norm boundary for relations $R = 1$.

While forming a better-structured embedding space, Injecting multiple types of rules only has a small impact on model complexity. InjEx has the same $O(nd+md)$ space complexity as ComplEx, where n is the number of entities, m is the number of relations and d is the dimension of complex-valued vector for entity and relation representations. The time complexity of InjEx is $O((\bar{t}+p+\bar{n})d)$ where $\bar{t}$ is the average number of triples in each batch, $\bar{n}$ is the average number of entities in each batch and p is the total number of rules we inject. In a real scenario, the number of rules is usually much smaller than the number of triples, i.e. $\bar{t} >> p$ and $\bar{n} \le \bar{t}$. Thus, the time complexity of InjEx is also $O(\bar{t}d)$, which is the time complexity of ComplEx.

5 EXPERIMENTS

In this section, we evaluate InjEx on two tasks, KGC and FKGC, and report state-of-the-art results. For the KGC task, InjEx demonstrates an enhancement in embedding quality, while FKGC results validate its effective utilization of prior knowledge in predictions, leading to improvement on both tasks.

In our experimental setup, we restrict rule lengths to a maximum of 2 to balance the efficiency and effectiveness of rule mining. Accordingly, rules are categorized based on their length: (1) The set of length-1 rules are hierarchy rules. (2) The set of rules with length 2 are composition rules. To showcase the impact of different rule types, we conduct an ablation study, comparing the performance of InjEx when exclusively incorporating composition rules (InjEx-C) or hierarchy rules (InjEx-H).

5.1 Knowledge Graph Completion

5.1.1 ***Datasets***. We evaluate the effectiveness of our InjEx for link prediction on three real-world KGC benchmarks: FB15k-237, WN18RR [3] and YAGO3-10 [20]. FB15k-237 excludes inverse relations from FB15k and includes 14541 entities, 237 relations, and 272,155 training triples. WN18RR is constructed from WordNet [22] with 40,943 entities, 11 relations and 93,013 triples. YAGO3-10 [20] is a subset of YAGO, containing 123,182 entities, 37 relations, and 1,079,040 training triples with almost all common relation patterns.

5.1.2 ***Baselines***. We compare InjEx with KGE models: TransE [3], RotatE [30], ComplEx-N3 [17], BoxE [1], and HAKE [42]; Rule-based or rule injection models: DRUM [27], SAFRAN [24], RUGE [13], KALE [12], ComplEx-NNE [9] and UniKER [7]; GNN/GCN model: pLogicNet [26], RNNLogic [25], and RED-GNN [41].

5.1.3 ***Rule Set***. We generate our rule set via AnyBURL to all the aforementioned benchmarks. We further extract length-1 and length-2 Horn rules with confidence higher than 0.5. As such, we extract 2552/10/172 hierarchies and 149/10/65 compositions for FB15k-237/WN18RR/YAGO3-10, respectively.

5.1.4 ***Experiment Setup***. We report two common evaluation metrics for both tasks: mean reciprocal rank (MRR), top-1 Hit Ratio(Hit@1), and top-10 Hit Ratio (Hit@10). For each triple in the test set, we replace its head entity e_o with every entity $e_i \in \mathcal{E}$ to create candidate triples in the link prediction task. We evaluate InjEx in a filtered setting as mentioned in [3], where all corrupted triples that already exist in either training, validation, or test sets are removed. All candidate triples are ranked based on their prediction scores. Higher MRR or Hit@k indicates better performance.

We ran the same grid of hyper-parameters for all models on the FB15K-237, WN18RR, NELL-One, and FB15k-237-Zero datasets. Our grid includes a learning rate $\gamma \in \{0.1, 0.2, 0.5\}$, two batch-sizes: 25 and 100, and regularization coefficients $\eta \in \{0, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5\}$. On YAGO3-10, we limited InjEx to embedding sizes $d = 1000$ and used batch-size 1000, as [1] only reports their result with $d \le 1000$, keeping the rest of the grid fixed. All experiments were conducted on a shared cluster of NVIDIA A100 GPUs.

We train InjEx for 100 epochs, validating every 20 epochs. For all models, we report the best-published results. The best results are highlighted in bold.

For ComplEx-NNE, we experiment on FB15k-237 with the same rule set as InjEx-C, which is not reported in the original paper. Other baselines results are taken from [30], [1] [17], [24], and [7].

5.1.5 ***Main Results***. Table 2 presents our results on FB15k-237, WN18RR and YAGO3-10. On FB15k-237, by injecting definite Horn rules, InjEx surpasses current leading models in Hits@10, outperforming both standalone embedding and rule-mining approaches. This highlights the efficacy of rule injection, particularly given FB15k-237's inherent composition patterns, indicating InjEx's ability to handle compositions even when not directly deduced.

In the WN18RR dataset, although the quality and amount of inference patterns are limited by only 11 relations, InjEx demonstrates competitive results, surpassing basic KGE models and most methods that integrate logical rules with embeddings. This achievement underscores InjEx's capability to excel even with limited inference pattern diversity.

YAGO3-10 results are particularly encouraging, with InjEx outperforming all leading models, including RotatE, BoxE, HAKE, and SAFRAN, showcasing significant enhancements. Given that YAGO3-10 is more complex than FB15k-237 and WN18RR, because of the various combinations of different types of inference patterns, the strong performance of InjEx suggests its efficient pattern capture mechanism, outmatching models recognized for their pattern inference capabilities like [1] and [28].

Overall, InjEx is competitive on all benchmarks and is state-of-the-art on YAGO3-10. Especially, it performs better than (or at least equally well as) ComplEx-N3 and ComplEx-NNE in both metrics

Table 2: KGC results (MRR, Hits@10) for InjEx and competing approaches on FB15k-237 and YAGO3-10. We re-run ComplEx-NNE on FB15k-237 and report the result. Other approach results are as best published.

	FB15k-237			WN18RR			YAGO3-10		
Model	MRR	Hit@1	Hit@10	MRR	Hit@1	Hit@10	MRR	Hit@1	Hit@10
TransE [3]	.332	.231	.531	.226	.007	.501	.501	.405	.673
RotatE [30]	.338	.237	.533	.476	.428	.571	.495	.402	.670
ComplEx [31]	.348	.253	.534	.477	.438	.543	.576	.505	.704
ComplEx-N3 [17]	.370	.270	.560	.480	.430	.570	.580	.500	.710
BoxE [1]	.337	-	.538	.451	-	.541	.567	-	.699
HAKE [42]	.346	.250	.542	.497	.452	.582	.545	.462	.694
DRUM [27]	.343	.255	.516	.486	.425	.586	-	-	-
SAFRAN [24]	.389	.298	.537	.502	.459	.578	.564	.491	.693
RUGE [13]	.191	.098	.376	.280	.251	.327	-	-	-
KALE [12]	.230	.131	.424	.172	.032	.353	-	-	-
ComplEx-NNE [9]*	.373	.272	.555	.481	.433	.580	.590	.514	.721
UniKER-RotatE [7]	**.539**	**.495**	.612	.492	.437	.580	-	-	-
pLogicNet [26]	.332	.261	.524	.441	.301	.537	-	-	
RNNLogic [25]	.349	.258	.533	.513	.471	.597	-	-	-
RED-GNN [41]	.374	.283	.558	**.533**	**.485**	**.624**	-	-	-
InjEx-H	.390	.300	.560	.482	.445	.580	.632	.524	.754
InjEx-C	.408	.305	.598	.481	.438	.581	.610	.518	.751
InjEx	.420	.308	**.615**	.483	.464	.588	**.660**	**.608**	**.761**

on all three benchmarks. It shows that with the soft constraint and penalty-term injection, InjEx does improve the quality of KG embedding. Hence, it is an effective and strong model that leverages prior knowledge for KGC on large real-world KGs.

5.1.6 ***Ablation Studies***. Table 2 also presents the performance of InjEx-C and InjEx-H on FB15k-237, WN18RR and YAGO3-10. The performance is under the identical parameter setting of InjEx, which means we did not fine-tune for one type of relation.

For FB15k-237 and WN18RR, the injection of either hierarchy or composition rules, represented by InjEx-H and InjEx-C, results in competitive performances, showcasing the adaptability and effectiveness of InjEx in utilizing distinct rule types. For YAGO3-10, both InjEx-H and InjEx-C outperform all state-of-the-art models. Further, the results of InjEx-H and InjEx-C against ComplEx-N3 show that InjEx's capability to enhance KG embedding quality through distinct pattern utilization.

The result further demonstrates InjEx's versatility and effectiveness in accommodating the varied rule distributions inherent across different KGs, affirming its capacity to significantly improve KG embeddings through strategic rule application.

5.2 Few-shot Knowledge Graph Completion

5.2.1 ***Dataset***. We evalute InjEx on two FKGC benchmarks:NELL-One [36] and FB15k-237-Zero. NELL-One [36] is generated from NELL [5] by removing inverse relations. It contains 68,545 entities, 358 relations, and 181,109 triples. 51/5/11 relations are selected as task relations for training, validation, and testing. Each task relation has only one triple in the corresponding set. FB15k-Zero is a dataset constructed from FB15k-237 following the same setting as NELL-One. We randomly select 8 relations as the task relations in the test set. We extract all triples with task relations from FB15k-237 as our test set. We randomly add 0, 1, 3, and 5 triples for each task relation into the training set and remove those from the test set to evaluate the effectiveness of InjEx on leveraging prior knowledge in different few-shot settings.

5.2.2 ***Baselines***. We compare InjEx with FKGC models GMatching [36] and MetaR [6], KGC models TransE [3], ComplEx [31] and Dismult [37] on NELL-One and with ComplEx-N3 [17] on FB15k-237-Zero.

5.2.3 ***Rule Set***. To get Horn rules for FB15K-237-Zero, we select only those from FB15K-237 rule set, with task relations in FB15k-237-Zero as its head relation, obtaining 209 hierarchy rules and 11 composition rules. For NELL-One, we use AnyBURL in the same fashion as in the KGC task to extract Horn rules. Since a small number of composition rules with high confidence compared to hierarchy rules are found for NELL-One, we include all the composition rules but only hierarchy rules with confidence ≥ 0.8, resulting in 3023 hierarchies and 38 compositions for NELL-One.

5.2.4 ***Experiment Setup***. We evaluate InjEx on 1-shot setting for NELL-One, providing 1 triple for each target relation in the training set. To leverage the supporting triples in ComplEx-N3 and InjEx, we use them in the training phase. On FB15k-237-Zero, we evaluate from 0-shot to 5-shot setting on both ComplEx-N3, InjEx-H, and InjEx-C. We report the MRR and Hits@10 for both tasks.

We ran the same grid of hyper-parameters for all models on the NELL-One and FB15k-237-Zero datasets. Our grid includes a learning rate $\gamma \in \{0.1, 0.2, 0.5\}$, two batch sizes: 25 and 1000, and regularization coefficients $\eta \in \{0, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5\}$. Other training settings are the same as in the KGC tasks.

5.2.5 ***Results***. Table 3 summarizes our results on NELL-One. InjEx outperforms all traditional KGC approaches, highlighting that prior rules improve the KG embedding of few-shot relations. InjEx-H (with hierarchy rules) also significantly improves compared to GMatching and is competitive with MetaR. Although GMatching considers the neighborhood information in their model, the lack of learning other patterns that exist in the original KG limits their performance. In contrast, InjEx is capable of leveraging information provided by various types of patterns.

Table 4 further illustrates InjEx effectively leveraging prior knowledge on the FKGC task. On FB15k-237-Zero, with 0 to 5 supporting

Table 3: FKGC experiment results on NELL-One with 1-shot setting

	TransE	DisMult	ComplEx	GMatching	MetaR	InjEx-H	InjEx-C	InjEx
MRR	.105	.165	.179	.185	**.250**	.240	.203	.245
Hit@10	.226	.285	.239	.279	.261	.304	.267	**.320**

Table 4: FKGC results (MRR, Hits@10) on FB15k-237-Zero with k-shot settings

	0-shot		1-shot		3-shot		5-shot	
FB15k-237-Zero	MRR	Hit@10	MRR	Hit@10	MRR	Hit@10	MRR	Hit@10
ComplEx-N3	.0011	.0016	.127	.180	.134	.181	.162	.243
InjEx-H	.07	.197	.08	.204	.114	.231	.151	.302
InjEx-C	**.123**	**.224**	**.146**	**.247**	**.158**	**.252**	**.207**	**.325**

triples provided for each task relation, InjEx-Composition consistently outperforms ComplEx-N3. Injecting hierarchy rules improves performance on Hits@10 for all settings but only for 0-shot on MRR. One possible reason is that Freebase does not have a clear ontology, affecting the quality of the hierarchy patterns. Furthermore, we observe that the gap between ComplEx-N3 and InjEx-C grows larger. This is against the intuition that as more triples are provided, prior knowledge may become less helpful since more information can be obtained from triples. This result demonstrates that prior rule knowledge can consistently provide a positive impact on the few-shot link prediction tasks.

We also observe that composition rules more positively impact FB15k-237-Zero, while hierarchy rules more positively impact NELL-One. This illustrates that different patterns have a diverse effect on supporting better few-shot relations, which motivates InjEx to inject various types of rules.

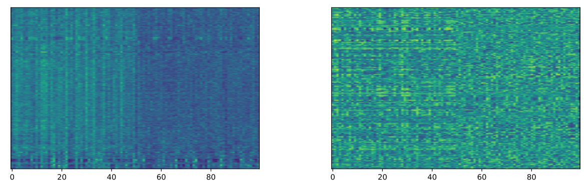
Further, on FB15k-237-Zero with 0 supporting triple (0-shot), the improvement between ComplEx-N3 and InjEx is statistically significant. With 0 supporting triples in the training set, such prior knowledge can only be captured from the rules. The improvement indicates that InjEx is able to effectively make use of prior knowledge during prediction.

5.3 Analysis on Entity Representations

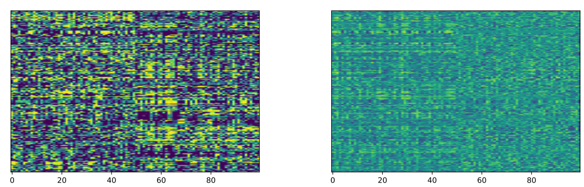
Investigating the entity embedding space alterations after rule injection, we visually present FB15k-237 entity embeddings. As Figure2a displays, entity embeddings under InjEx exhibit a more structured dimensionality in contrast to ComplEx-N3, demonstrating InjEx's ability to prioritize meaningful dimensions for entities. This targeted approach results in more efficient representations, illustrating InjEx's improvement over conventional ComplEx models.

5.4 Analysis on Relation Representations

We further present the visual analysis of relation embeddings within FB15k-237, where each relation is tagged with a specific type label. We aim to demonstrate how InjEx's norm constraints and added loss terms refine relation embeddings, as illustrated in Figure2b. The visualized distribution highlights InjEx's proficiency in achieving structured and interpretable relation representations, activating fewer dimensions than ComplE-N3 and thus indicating a more representative extraction of knowledge from the KG.

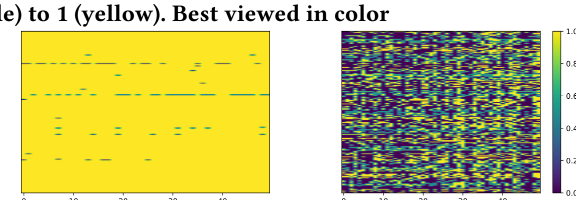


(a) Entity representations

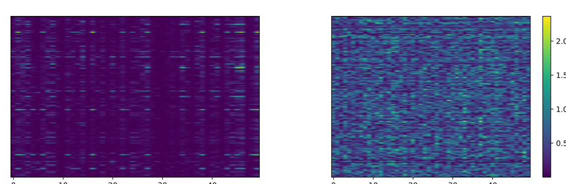


(b) relation representations

Figure 2: Visualization of "active" dimensions in entity and relation representations (rows) learned by InjEx (left) and ComplEx-N3 (right) on FB15k-237. Values range from 0 (purple) to 1 (yellow). Best viewed in color

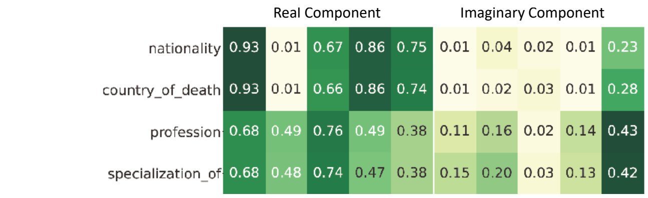


(a) Difference between real components

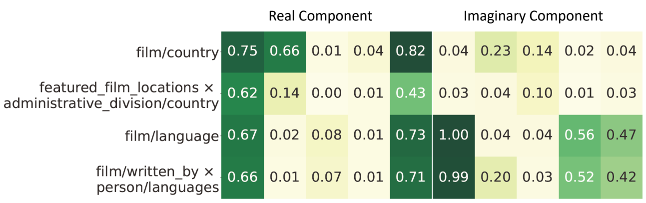


(b) Difference between imaginary components

Figure 3: Visualization of differences between real, imaginary components of head and body relation(s) representations learned by InjEx with Horn rule set (left) and ComplEx-N3 (right). Values range from low (purple) to high (yellow). Best viewed in color.



(a) Examples of relation wrepresentations of hierarchy rules



(b) Examples of relation representations of composition rules

Figure 4: Relation representations learned by InjEx. (a) Example of hierarchy rules. (b) Example of composition rules

Additionally, Figure 3 visualizes the difference between the representations of head and body relations included in the Horn rule set, learned by ComplEx-N3 and InjEx. We randomly pick 50 dimensions from both their real and imaginary components.

For the real components, the injection of Horn rules anticipates the head relation's representation to surpass or match the element-wise multiplication of the body relation(s). By evaluating this in Figure3a, we observe InjEx's adherence to the rules outlined in Eq5 and Eq8, showcasing its alignment with theoretical expectations, unlike the arbitrary nature of ComplEx-N3's representations.

Regarding the imaginary components, Horn rules tend to force head relation(s) to be similar to the element-wise multiplication of body relation(s). The analysis presented in Figure 3b reveals that InjEx's constraints and loss terms effectively narrow the disparity between head and body relations, showcasing an improvement in relational representation learning over ComplEx-N3.

To further show how InjEx affects the learning of relation representations, we visualize the representations of two pairs of relations from hierarchy rules and two pairs of relations from composition rules learned by InjEx in Figure 4a and Figure4b. For each relation, we randomly pick 5 dimensions from both its real and imaginary components. We present:

```
## hierarchy rules
/people/person/nationality <-
    /people/deceased_person/place_of_death
/people/person/profession <-
    /people/profession/specialization_of
## composition rules
/film/film/country <- /film/film/featured_film_locations ∧
    /location/administrative\_division/country
/film/film/language <- /film/film/written_by ∧
    /people/person/languages
```

By imposing such Horn rules, these relations can encode such logical regularities quite well. InjEx learns to force similar representations for head and body relations to follow $Re(\mathbf{r}_1 \times \mathbf{r}_2 \times \ldots \times \mathbf{r}_k)/R^k \leq Re(\mathbf{r})/R$, $Im(\mathbf{r}_1 \times \mathbf{r}_2 \times \ldots \times \mathbf{r}_k)/R^k \approx Im(\mathbf{r})/R$ for Horn rules, thus, impose prior knowledge into relation embeddings.

6 CONCLUSION AND FUTURE WORK

In this paper, we present InjEx, and prove its expressive power and ability to inject different types of rules. We empirically showed that InjEx achieves state-of-the-art performance for KGC and is competitive for few-shot KGC by injecting different types of rules. InjEx can be further updated and improved by leveraging additional types of rules for both tasks. Because the results indicate that Horn rules with different lengths are all valuable for KGC, it is worth exploring further interactions among other rule combinations in order to maximize the possible impact of various types of rules.

REFERENCES

[1] Ralph Abboud, Ismail Ceylan, Thomas Lukasiewicz, and Tommaso Salvatori. 2020. Boxe: A box embedding model for knowledge base completion. *Advances in Neural Information Processing Systems* 33 (2020), 9649–9661.

[2] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*. 1247–1250.

[3] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems* 26 (2013).

[4] Avishek Joey Bose, Ankit Jain, Piero Molino, and William L Hamilton. 2019. Meta-graph: Few shot link prediction via meta learning. *arXiv preprint arXiv:1912.09867* (2019).

[5] Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R Hruschka, and Tom M Mitchell. 2010. Toward an architecture for never-ending language learning. In *Twenty-Fourth AAAI conference on artificial intelligence*.

[6] Mingyang Chen, Wen Zhang, Wei Zhang, Qiang Chen, and Huajun Chen. 2019. Meta Relational Learning for Few-Shot Link Prediction in Knowledge Graphs. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. 4217–4226.

[7] Kewei Cheng, Ziqing Yang, Ming Zhang, and Yizhou Sun. 2021. UniKER: A Unified Framework for Combining Embedding and Definite Horn Rule Reasoning for Knowledge Graph Inference. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, 9753–9771. https://doi.org/10.18653/v1/2021.emnlp-main.769

[8] Thomas Demeester, Tim Rocktäschel, and Sebastian Riedel. 2016. Lifted rule injection for relation embeddings. *arXiv preprint arXiv:1606.08359* (2016).

[9] Boyang Ding, Quan Wang, Bin Wang, and Li Guo. 2018. Improving Knowledge Graph Embedding Using Simple Constraints. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 110–121.

[10] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research* 12, 7 (2011).

[11] Luis Antonio Galárraga, Christina Teflioudi, Katja Hose, and Fabian Suchanek. 2013. AMIE: association rule mining under incomplete evidence in ontological knowledge bases. In *Proceedings of the 22nd international conference on World Wide Web*. 413–422.

[12] Shu Guo, Quan Wang, Lihong Wang, Bin Wang, and Li Guo. 2016. Jointly Embedding Knowledge Graphs and Logical Rules. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Austin, Texas, 192–202. https://doi.org/10.18653/v1/D16-1019

[13] Shu Guo, Quan Wang, Lihong Wang, Bin Wang, and Li Guo. 2018. Knowledge graph embedding with iterative guidance from soft rules. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32.

[14] Zhiting Hu, Xuezhe Ma, Zhengzhong Liu, Eduard H Hovy, and Eric P Xing. 2016. Harnessing Deep Neural Networks with Logic Rules. In *ACL (1)*.

[15] Dora Jambor, Komal Teru, Joelle Pineau, and William L Hamilton. 2021. Exploring the Limits of Few-Shot Link Prediction in Knowledge Graphs. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. 2816–2822.

[16] Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Knowledge graph embedding via dynamic mapping matrix. In *Proceedings of the 53rd annual*

meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (volume 1: Long papers). 687–696.

[17] Timothée Lacroix, Nicolas Usunier, and Guillaume Obozinski. 2018. Canonical tensor decomposition for knowledge base completion. In *International Conference on Machine Learning*. PMLR, 2863–2872.

[18] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *Twenty-ninth AAAI conference on artificial intelligence*.

[19] Xin Lv, Yuxian Gu, Xu Han, Lei Hou, Juanzi Li, and Zhiyuan Liu. 2019. Adapting Meta Knowledge Graph Information for Multi-Hop Reasoning over Few-Shot Relations. In *EMNLP/IJCNLP (1)*.

[20] Farzaneh Mahdisoltani, Joanna Biega, and Fabian Suchanek. 2014. Yago3: A knowledge base from multilingual wikipedias. In *7th biennial conference on innovative data systems research*. CIDR Conference.

[21] Christian Meilicke, Melisachew Wudage Chekol, Daniel Ruffinelli, and Heiner Stuckenschmidt. 2019. Anytime Bottom-Up Rule Learning for Knowledge Graph Completion. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*. International Joint Conferences on Artificial Intelligence Organization, 3137–3143. https://doi.org/10.24963/ijcai.2019/435

[22] George A Miller. 1995. WordNet: a lexical database for English. *Commun. ACM* 38, 11 (1995), 39–41.

[23] Brian Murphy, Partha Talukdar, and Tom Mitchell. 2012. Learning effective and interpretable semantic models using non-negative sparse embedding. In *Proceedings of COLING 2012*. 1933–1950.

[24] Simon Ott, Christian Meilicke, and Matthias Samwald. 2021. SAFRAN: An interpretable, rule-based link prediction method outperforming embedding models. In *3rd Conference on Automated Knowledge Base Construction*.

[25] Meng Qu, Junkun Chen, Louis-Pascal Xhonneux, Yoshua Bengio, and Jian Tang. 2020. Rnnlogic: Learning logic rules for reasoning on knowledge graphs. *arXiv preprint arXiv:2010.04029* (2020).

[26] Meng Qu and Jian Tang. 2019. Probabilistic logic neural networks for reasoning. *Advances in neural information processing systems* 32 (2019).

[27] Ali Sadeghian, Mohammadreza Armandpour, Patrick Ding, and Daisy Zhe Wang. 2019. Drum: End-to-end differentiable rule mining on knowledge graphs. *Advances in Neural Information Processing Systems* 32 (2019).

[28] Tengwei Song, Jie Luo, and Lei Huang. 2021. Rot-Pro: Modeling Transitivity by Projection in Knowledge Graph Embedding. *Advances in Neural Information Processing Systems* 34 (2021).

[29] Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2008. Yago: A large ontology from wikipedia and wordnet. *Journal of Web Semantics* 6, 3 (2008), 203–217.

[30] Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2018. RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space. In *International Conference on Learning Representations*.

[31] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *International conference on machine learning*. PMLR, 2071–2080.

[32] Feiyang Wang, Zhongbao Zhang, Li Sun, Junda Ye, and Yang Yan. 2022. DiriE: Knowledge Graph Embedding with Dirichlet Distribution. In *Proceedings of the ACM Web Conference 2022*. 3082–3091.

[33] Kai Wang, Yu Liu, Qian Ma, and Quan Z Sheng. 2021. Mulde: Multi-teacher knowledge distillation for low-dimensional knowledge graph embeddings. In *Proceedings of the Web Conference 2021*. 1716–1726.

[34] Shen Wang, Xiaokai Wei, Cicero Nogueira Nogueira dos Santos, Zhiguo Wang, Ramesh Nallapati, Andrew Arnold, Bing Xiang, Philip S Yu, and Isabel F Cruz. 2021. Mixed-curvature multi-relational graph neural network for knowledge graph completion. In *Proceedings of the Web Conference 2021*. 1761–1771.

[35] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 28.

[36] Wenhan Xiong, Mo Yu, Shiyu Chang, Xiaoxiao Guo, and William Yang Wang. 2018. One-Shot Relational Learning for Knowledge Graphs. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 1980–1990.

[37] Bishan Yang, Scott Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding Entities and Relations for Learning and Inference in Knowledge Bases. In *Proceedings of the International Conference on Learning Representations (ICLR) 2015*.

[38] Donghan Yu, Yiming Yang, Ruohong Zhang, and Yuexin Wu. 2021. Knowledge embedding based graph convolutional network. In *Proceedings of the Web Conference 2021*. 1619–1628.

[39] Wen Zhang, Bibek Paudel, Liang Wang, Jiaoyan Chen, Hai Zhu, Wei Zhang, Abraham Bernstein, and Huajun Chen. 2019. Iteratively learning embeddings and rules for knowledge graph reasoning. In *The world wide web conference*. 2366–2377.

[40] Yuyu Zhang, Xinshi Chen, Yuan Yang, Arun Ramamurthy, Bo Li, Yuan Qi, and Le Song. 2019. Can graph neural networks help logic reasoning? *arXiv preprint arXiv:1906.02111* (2019).

[41] Yongqi Zhang and Quanming Yao. 2022. Knowledge graph reasoning with relational digraph. In *Proceedings of the ACM Web Conference 2022*. 912–924.

[42] Zhanqiu Zhang, Jianyu Cai, Yongdong Zhang, and Jie Wang. 2020. Learning hierarchy-aware knowledge graph embeddings for link prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 3065–3072.

[43] Zhanqiu Zhang, Jie Wang, Jieping Ye, and Feng Wu. 2022. Rethinking Graph Convolutional Networks in Knowledge Graph Completion. In *Proceedings of the ACM Web Conference 2022*. 798–807.

A PROOF OF THEOREM 1

Proof. With the restrictions: $|r_i| \in (0, R]$ and $Re(e_i), Im(e_i) \in (0, 1]$, for every dimension l we have

$$\begin{aligned} \varphi_{1l} &:= \varphi_l(e_1, r_1, e_2) \\ &= |(|r_{1l}||e_{1l}||e_{2l}|cos(\theta_{r_{1l}} + \theta_{e_{1l}} - \theta_{e_{2l}}))| \in [0, 2R] \\ \varphi_{2l} &:= \varphi_l(e_2, r_2, e_3) \\ &= |(|r_{2l}||e_{2l}||e_{3l}|cos(\theta_{r_{2l}} + \theta_{e_{2l}} - \theta_{e_{3l}}))| \in [0, 2R] \end{aligned}$$

which means we can treat $\Pi_l(|\varphi_{il}/(2R)|)$ like the probability of triple i being True. Naturally, for rule $r_1, r_2 \rightarrow r_3$ to hold, it is sufficient if we prove that for each dimension l,

$$\begin{aligned} |\varphi_{3l}/(2R)| &:= |(|r_{3l}||e_{1l}||e_{3l}|cos(\theta_{r_{3l}} + \theta_{e_{1l}} - \theta_{e_{3l}}))/(2R)| \\ &\geq |\varphi_{1l}/(2R)| * |\varphi_{2l}/(2R)| \end{aligned} \tag{12}$$

We will prove the above (12) in two steps (the proof is element-wise, for simplicity, we drop footnote l in all notations). Let $r := \langle r_1, r_2 \rangle / R$, we will show:

$$|\varphi_1/(2R)| * |\varphi_2/(2R)| \overset{?}{\leq} |\varphi_r/(2R)| \tag{13}$$

$$|\varphi_r/(2R)| \overset{?}{\leq} |\varphi_3/(2R)| \tag{14}$$

Let $\alpha := \theta_{r_1} + \theta_{e_1} - \theta_{e_2}$, $\beta := \theta_{r_2} + \theta_{e_2} - \theta_{e_3}$, and for $r = \langle r_1, r_2 \rangle / R$, we have $\theta_r = \theta_{r_1} + \theta_{r_2}$.

$r_k = (cos\theta_{r_k} + \mathrm{i}\, sin\theta_{r_k}) * |r_k|$, so

$r = \langle r1, r2 \rangle / R = \{(cos\theta_{r_1} * cos\theta_{r_2} - sin\theta_{r_1} * sin\theta_{r_2}) + \mathrm{i}\, (cos\theta_{r_1} * sin\theta_{r_2} + cos\theta_{r_2} * sin\theta_{r_1})\} * |r_1| * |r_2| / R = \{cos(\theta_{r_1} + \theta_{r_2}) + \mathrm{i}\, sin(\theta_{r_1} + \theta_{r_2})\} * |r|$ where $|r| = |r_1| * |r_2| / R$ and $\theta_r = \theta_{r_1} + \theta_{r_2}$

First, we prove (13). When triples (e_1, r_1, e_2) and (e_2, r_2, e_3) are both True, φ_1 and φ_2 reach maximum value, which means $\alpha \approx \beta \approx 0$.

Notice that for φ_r, $\theta_r + \theta_{e_1} - \theta_{e_3} = \theta_{r_1} + \theta_{r_2} + \theta_{e_1} - \theta_{e_3} = \alpha + \beta \approx 0$. Then 13 becomes $|e_1||e_2|^2|e_3|/2 \leq |e_1||e_3|$, which holds given that $|e_i|^2 \leq 2$.

To prove (14) we only need to show $|\varphi_r| \leq |\varphi_3|$, which according to [9] is valid as long as they are positive. □

B PROOF OF THEOREM 2

Proof. If we have triples (z_{i-1}, r_i, z_i) for $1 \leq i \leq k$,

$$\begin{aligned} \varphi_{i,l} &:= Re(z_{i-1,l} * r_{i,l} * \bar{z}_{i,l}) \\ &= |r_{i,l}||z_{i-1,l}||z_{i,l}| \cos(\theta_{r_{i,l}} + \theta_{z_{i-1,l}} - \theta_{z_{i,l}}) \\ \phi_i &:= \Sigma_l(\varphi_{i,l}) \end{aligned}$$

Consider $rule(r_1 \wedge r_2 \wedge ... \wedge r_k \longrightarrow r)$ for triples (x, r, y) s.t. $x = z_0$ and $y = z_k$. Let Π be the element-wise multiplication, let

$$\hat{r} := R * \Pi_{i=1}^{k}(r_i/R), |\hat{r}| \in (0, R] \tag{15}$$

$$\begin{aligned}\hat{\varphi}_l &:= Re(z_{0,l} * \hat{r}_l * z_{k,l}) \\ &= |\hat{r}_l||z_{0,l}||z_{k,l}| \cos(\theta_{\hat{r}_l} + \theta_{z_{0,l}} - \theta_{z_{k,l}}) \end{aligned} \tag{16}$$

then by definition, we have,

$$\varphi_l := Re(x_l * r_l * \bar{y}_l)$$
$$\phi := \Sigma_l(\varphi_l)$$

let r satisfies that (with confidence level λ):

$$\begin{aligned}\lambda(Re(\hat{r})/R - Re(r)/R) &\leq \alpha \\ \lambda(Im(\hat{r})/R - Im(r)/R) &\leq \beta\end{aligned} \tag{17}$$

then for each relation (x', r', y') and $1 \leq l \leq n$, where n is the dimension of the vectors, we have $\varphi_{r'_l}/(2R) \in [0, 1]$, so we can treat $\varphi_{r'_l}/(2R)$ as the probability that the relation holds for the triple's attribute l. We imply $rule(r_1 \wedge r_2 \wedge ... \wedge r_k \longrightarrow r)$ by enforcing that on each dimension l, $\Pi_{i=1}^{k}(\varphi_{i,l}/(2R)) \leq \varphi_l/(2R)$.

Now we prove that our definitions above can imply the rule. For each $1 \leq i \leq k$, relation holds for triple (z_{i-1}, r_i, z_i), so φ_i is maximized on each dimension. So we have for each $1 \leq l \leq n$,

$$(\theta_{r_{i,l}} + \theta_{z_{i-1,l}} - \theta_{z_{i,l}}) = 0$$

Therefore,

$$\begin{aligned}\Pi_{i=1}^{k}(\varphi_{i,l}/(2R)) &= \Pi_{i=1}^{k}((|r_{i,l}|/R)|z_{i-1,l}||z_{i,l}|/2) \\ &= \Pi_{i=1}^{k}(|r_{i,l}|/R) * |z_{0,l}| * \Pi_{i=1}^{k-1}(|z_{i,l}|^2/2) * |z_{k,l}|/2\end{aligned}$$

Notice that by the definition of (15) and (16),

$$\begin{aligned}&\theta_{\hat{r}_l} = \Sigma_{i=1}^{k}\theta_{r_{i,l}} \\ &|\hat{r}_l|/R = \Pi_{i=1}^{k}(|r_{i,l}|/R) \\ &\theta_{\hat{r}_l} + \theta_{z_{0,l}} - \theta_{z_{k,l}} = \Sigma_{i=1}^{k}(\theta_{r_{i,l}} + \theta_{z_{i-1,l}} - \theta_{z_{i,l}}) = 0 \\ \Longrightarrow\ &\hat{\varphi}_l/R = (|\hat{r}_l|/R)|z_{0,l}||z_{k,l}| \cos(\theta_{\hat{r}_l} + \theta_{z_{0,l}} - \theta_{z_{k,l}}) \\ &\qquad = \Pi_{i=1}^{k}(|r_{i,l}|/R)|z_{0,l}||z_{k,l}|\end{aligned}$$

Given that $|z_{i,l}| \leq 1$,

$$|z_{0,l}| * \Pi_{i=1}^{k-1}(|z_{i,l}|^2) * |z_{k,l}| \leq |z_{0,l}||z_{k,l}|$$

So we have

$$\Pi_{i=1}^{k}(\varphi_{i,l}/R) \leq \hat{\varphi}_l/R$$

According to [9], our choice of r in (17) guarantees that

$$\Pi_{i=1}^{k}(\varphi_{i,l}/R) \leq \hat{\varphi}_l/R \leq \varphi_l/R$$

□