EXPLAINING KNOWLEDGE GRAPH EMBEDDING VIA LATENT RULE LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Knowledge Graph Embeddings (KGEs) embed entities and relations into continuous vector space following certain assumptions, and are powerful tools for representation learning of knowledge graphs. However, following vector space assumptions makes KGE a one step reasoner that directly predicts final results without reasonable multi-hop reasoning steps. Thus KGEs are black-box models and explaining predictions made by KGEs remains unsolved. In this paper, we propose *KGExplainer*, the first general approach of providing explanations for predictions from KGE models. *KGExplainer* is a multi-hop reasoner learning latent rules for link prediction and is encouraged to behave similarly to KGEs during prediction through knowledge distillation. For explanation, KGExplainer outputs a ranked list of rules for each relation. Experiments on benchmark datasets with two target KGEs show that our approach is faithfulness to replicate KGEs behaviors for link prediction and is good at outputting quality rules for effective explanations.

1 INTRODUCTION

Knowledge Graphs (KGs) are natural models for background knowledge in many real-world applications, including recommender system(Wong et al., 2021), question answering(Huang et al., 2019), and natural language processing(Annervaz et al., 2018). Thus many large scale KGs have been built, while they are still challenging to work with because of incompleteness. To address this problem, Knowledge Graph Embeddings (KGEs) have emerged as state-of-the-art for link prediction and representation learning on knowledge graphs. With the capability of encoding semantics of entities and relations into continuous vector space as embeddings, KGEs transfer link prediction into simple but efficient calculations over embeddings in vector space.

Despite their strengths, KGEs have been argued for lack of transparency for a long time as no human-intelligible explanations could be easily provided for their one-step predictions which limits trustworthiness from human-being, the maintainer as well as the final consumers of KGs (Nandwani et al., 2020). Thus many more transparent and explainable methods have been applied and preferred including reinforcement learning (Xiong et al., 2017; Das et al., 2018; Wan et al., 2020) and differentiable rule learning (Yang et al., 2017; Sadeghian et al., 2019; Wang et al., 2020). Their explainability is enabled in the explicit multi-hop reasoning steps of the models. Since there is no method for explaining KGEs, this inspires us to explain KGEs via transforming KGEs into a multi-hop reasoning model.

In this paper, we propose KGExplainer, an approach for explaining predictions made by KGEs. We follow the idea to approximate models with a surrogate model which is probed for explanations (Lakkaraju et al., 2017; Schmitz et al., 1999). KGExplainer is set to a multi-hop reasoning model learning latent rules and trained via making the multi-hop reasoning results approach to KGEs' results. Thus during training, we regard the scores from target KGEs as soft labels for KGExplainer and labels from data as hard labels following the setting of knowledge distillation(Gou et al., 2021). In KGExplainer, we represent latent rules for each relation as a rule embedding, based on which a rule decoder is learnt to sequentially decode compositive body relation embeddings from rule embeddings. The body relation embeddings will be orderly used for multi-step reasoning following translation function defined in target KGEs. Explanations are provided by a set of ranked symbolic rules decoded from body relation embeddings based on general vector space similarity as-

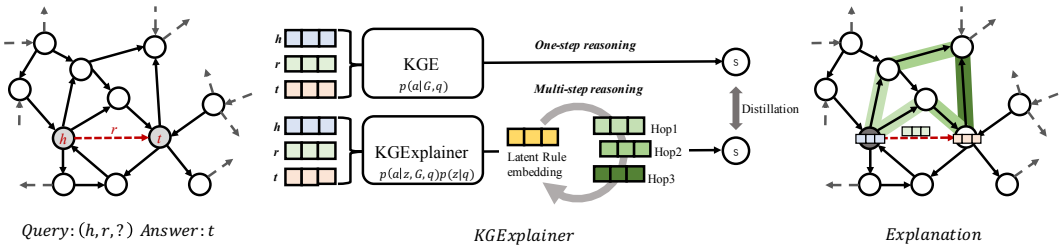


Figure 1: KGExplainer explains the target KGE via learning a multi-step surrogate model with latent rules and providing paths from head entity to tail entity as explanations.

sumption. Given a query, KGExplainer provides explanations as paths from query entity to answer entity, chosen from the existing paths according to the ranked rules list from KGExplainer, as shown in Figure 1. This approach is suitable for KGEs following a two-functional paradigm with a translation function f to translate head entity embedding to tail entity embedding according to relation embedding, and a scoring function g to calculate the similarity between two entity embeddings.

On two commonly used benchmark datasets for link prediction, we firstly evaluate the faithfulness of KGExplainer to approximate target KGEs and then present the evaluation over explanations to show it's capability of providing explanations. Experiments show that KGExplainer successfully replicate KGEs with almost zero score difference averaged on samples, and generate higher quality rules than two statistical baseline and a differentiable rule learning method, and provide consistent and concise explanations to help people understand the predictions and judge their correctness.

2 Related Work

Knowledge Graph Embeddings embed entities and relations into low-dimensional vector space, called embedding, enabling complex predictions through calculation with entity and relation embeddings. Various vector spaces are applied to KGEs, including Real space (Bordes et al., 2013), ComplEx space (Trouillon et al., 2016), Hyperbolic space (Chami et al., 2020), Hypercomplex space (Cao et al., 2021), etc.

The score function of triples Φ in KGEs we are interested in this paper is composed of two functions represented as $f(h,r,t) = g(f(h,r),t)$(Bordes et al., 2013; Sun et al., 2019; Yang et al., 2015; Trouillon et al., 2016). Lets represent entity and relation embeding space as $\mathbb{E}$ and $\mathbb{P}$, and real number space as $\mathbb{R}$. Given an head embedding $\mathbf{h} \in \mathbb{E}$ and a relation embedding $\mathbf{r} \in \mathbb{R}$, transformation function $f : \mathbb{E} \times \mathbb{P} \mapsto \mathbb{E}$ transforms $\mathbf{h}$ to predicted tail entity embedding $\mathbf{t} \in \mathbb{E}$ that the entity and relation correspond to $\mathbf{h}, \mathbf{r}, \mathbf{t}$ are possibly to composed a positive triple. Given two entity embeddings, $g : \mathbb{E} \times \mathbb{E} \mapsto \mathbb{R}$ scores them with a real number to indicate their similarity. The design of f and g in KGEs usually follows specific vector space assumptions. For example, TransE(Bordes et al., 2013) assumes that for a positive triple (h,r,t), the head entity could be linearly translated to tail entity with the relation that $\mathbf{h} + \mathbf{r} = \mathbf{t}$, thus $f_{transe}(\mathbf{h}, \mathbf{r}) = \mathbf{h} + \mathbf{r}$ and the similarity between entity embedding are evaluated by Euclidean distance that $g(\mathbf{t}', \mathbf{t}) = \|\mathbf{t}' - \mathbf{t}\|$.

Some KGEs do not strictly follow this two-functional paradigm, they either shrink two functions into one score function that is unable to split(Dettmers et al., 2018) or first combine head and tail entity embedding and then calculate the similarity in relation space(Abboud et al., 2020) or calculate the similarity with entity embeddings after transformed by relation(Wang et al., 2014; Lin et al., 2015b). In this paper, we focus on explaining KGEs following the general two-functional paradigm described above. Although there are methods providing explanations for completion results, some of them made explanations towards why the prediction is reasonable (Nandwani et al., 2020; Gusmão et al., 2018), or made a trade-off between performance and explainability(Stadelmaier & Padó, 2019), thus faithful explainer towards how KGE predicts is necessary.

Differentiable Rule learning is firstly proposed by Yang et al. (2017) to learn the confidence score of a rule and its structure at the same in a differentiable framework. Since rules are human-friendly to provide explanations for link prediction, and rule mining methods in a differentiable manner is more easy to be integrated into other deep learning methods, differentiable rule learning methods(Yang et al., 2017; Sadeghian et al., 2019; Wang et al., 2020) are widely been interested. They are build based on operations defined in TensorLog(Cohen et al., 2020) and the key idea is to generate the weights of relation for each reasoning step given a query, and rules could be decoded based on

accumulation of the weights in each step, which greatly motivate us for rule learning and decoding in KGExplainer.

Explaining Black Box Models. Although explaining prediction of KGEs is not well-studied, building robust and practical methods to explain black-box model's, especially neural works', has developed into an own research area(Guidotti et al., 2019; Samek et al., 2021). To generate post hoc explanations, there are four families of techniques. The first category of methods aims to learn a self-explanatory surrogate model to replace the target model(Ribeiro et al., 2016; Guidotti et al., 2018; Ribeiro et al., 2018). The second category of methods are build based on perturbation analysis where the input is repeatedly updated to test the effect on the neural networks' output, including occasion analysis(Zintgraf et al., 2017), shapley value analysis(Lundberg & Lee, 2017) and other meaningful perturbation(Fong & Vedaldi, 2017). The third category of methods explain by integrating the gradient along some trajectory in input space connecting some root point to data point(Sundararajan et al., 2017; Smilkov et al., 2017). The fourth category is Layerwise Relevance Propagation method that makes explicit use of the layered structure of the neural networks and operates in an iterative manner to produce the explanation(Montavon et al., 2019).

While existing explanation methods based on meaningful input features implicitly assume that those input features are inseparable to the receiver such as human, they are hard to directly implied to representation learning methods such as knowledge graph embedding, since the input of representation learning methods, embeddings, is unexplainable regarding to each dimension of the representation. In KGEs, the embedding of entities and relations are interpretable only when regarding them as a whole and calculating the similarity to other entities and relations in the embedding space. Thus in this paper, we take an alternative way to explain KGEs by expanding KGEs into an explainable multi-hop reasoning method and explaining each step via interpretable similarities in vector space.

3 PRELIMINARY

Knowledge Graph. Let $\mathcal{K} = \{\mathcal{E}, \mathcal{R}, \mathcal{T}\}$ denotes a knowledge graph, where $\mathcal{E}, \mathcal{R}$ and $\mathcal{T}$ are set of entities, relations and triples. Triples are of the form (h, r, t) where $h \in \mathcal{E}$ and $t \in \mathcal{E}$ are head entity and tail entity and $r \in \mathcal{R}$ is the relationship between h and t, an example is $(Iphone, brandIs, Apple)$. Knowledge graphs are incomplete and without loss of generality, we consider the problem of explaining a link prediction task in this paper.

Explanation for Link Prediction. A common explanation for link prediction query $q = (h, r, ?)$ with answer as $a = t$ is a multi-hop path starting from h to t. For example, suppose the answer for $(Bob, fatherIs, ?)$ is $David$, one of the explanation might be $Bob \xrightarrow{motherIs} Marry \xrightarrow{hasHusband} David$ which could be formally written as

$$(Bob, fatherIs, David) \leftarrow (Bob, motherIs, Marry) \wedge (Marry, hasHusband, David) \quad (1)$$

Such explanation could be regarded as a grounding of certain rules. For example, explanation (1) is a grounding of following rule

$$(X, fatherIs, Y) \leftarrow (X, motherIs, Z) \wedge (Z, hasHusband, Y) \quad (2)$$

Since rules are of the human-intelligible form, given a target KGE to be explained, this motivates us to learn a set of rules as the underlying logic for prediction as long as based on which the link prediction results are similar to the target KGE. Thus we frame the task of explaining KGE as learning a model with triple scoring function Ψ including a set of latent rules performs the same as target KGEs with triple score function Φ that

$$\Psi(q, a) \approx \Phi(q, a), \text{ where } \Psi(q, a) = \sum_{z \in \mathcal{Z}} p(a|\mathcal{G}, q, z) p(z|q) \quad (3)$$

where $\mathcal{Z}$ is a set of latent rules, and $p(z|q)$ is a prior over the set of rules conditioned on the query q, and $p(a|\mathcal{G}, q, z)$ is the likelihood of the answer a conditioned on rule z, q and $\mathcal{G}$.

Rules. As shown in Equation (2), rules are of the form $head \leftarrow body$ where $head$ and $body$ are composed of atoms containing variables and relations. Our interests in this paper is the type of rule closely related to paths from one entity to another entity proven to be useful for link predictionGuu et al. (2015); Zhang et al. (2019); Yang et al. (2015); Niu et al. (2020), called chain-liked rules.

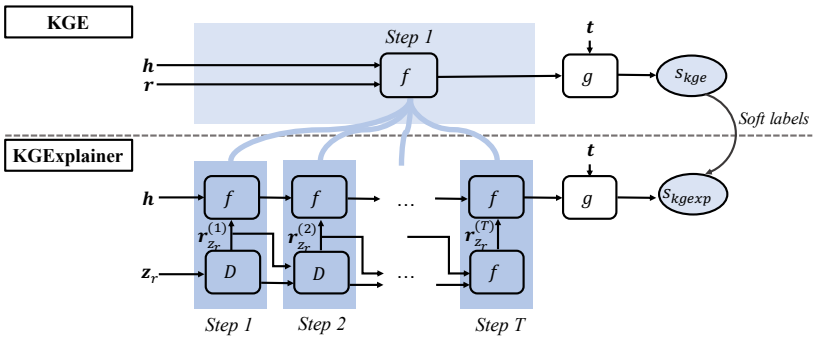


Figure 2: Overview of KGExplainer. f and g are the translation and similarity function from the target KGE. $\mathbf{z}_r$ is the rule emebdding for relation r and $\mathbf{r}_{z_r}^{(i)}$ is the $i$the compositive relation embedding from rule decoder $\mathcal{D}$.

In chain-liked rules, each variable appears twice, example refers to Equation (2). Rules from a learning system usually are associated with a confidence score α given by the system to indicate how confident this rule is. Thus chain-liked rules could be abbreviated as $\alpha : r^{(0)} \leftarrow [r^{(1)}, r^{(2)}, ..., r^{(T)}]$ where T is the length of the rule. Given a set of rules $\mathcal{Z}$, there are usually two steps to answer a target query $q = (h, r, ?)$, firstly to select rules z_r with relation in $head$ as r, called head relation, meaning that z_r could be used to infer new triples containing relation r, and secondly to walk along the relation sequence in rule $body$ starting from h and the finally reached entities are answers given by z and they are scored proportionally according to confidence of z.

4 KGEXPLAINER

In this section, we introduce KGExplainer applied to TransE(Bordes et al., 2013) and RotatE(Sun et al., 2019), two typical KGEs.

Learning KGExplainer. Given a well-trained target KGE with score function for a triple as $\Phi_\omega(h, r, t) = g(f(h, r), t)$ where $f : \mathbb{E} \times \mathbb{P} \mapsto \mathbb{E}$ and $g : \mathbb{E} \times \mathbb{E} \mapsto \mathbb{R}$, in order to make KGExplainer with score function Ψ_θ replicate the link prediction performance of that target KGE, we adopt the idea of knowledge distillation(Gou et al., 2021) and make the score from target KGE for triples as soft labels of KGExplainer, and labels from data as hard labels. Thus the objective for KGExplainer's optimization is

$$\min_\theta \sum_{(q,a)\in\mathcal{G}} \|\Phi_\omega(a|q, \mathcal{G}) - \Psi_\theta(a|q, \mathcal{G})\|_1 + L(q, a) \tag{4}$$

where $q = (h, r, ?)$, $a = t$, the first term is the soft label loss of L1 norm of score difference between KGExplainer and the target KGE, and the second term is the hard label loss, which is defined with negative sampling in the following form for both TransE and RotatE:

$$L(q, a) = -log\sigma(\gamma - \Psi(h, r, t)) - \sum_{i=1}^{k} p(h'_i, r_i, t'_i) log\sigma(\Psi(h'_i, r_i, t'_i) - \gamma) \tag{5}$$

where (h', r, t') are the negative triples sampled following the self-adversarial strategy proposed in (Sun et al., 2019). In Equation (4), score of KGExplainer is designed based on a set of latent rules $\mathcal{Z}$ similar to Qu et al. (2021) that

$$\Psi_\theta(a|G, q) = \sum_{z\in\mathcal{Z}} p_\tau(a|\mathcal{G}, q, z) p_\gamma(z|q) \tag{6}$$

where $p_\gamma(z|q)$ is the prior probability of applying rule z for q and $p_\tau(a|\mathcal{G}, q, z)$ is the likelihood of a for query q given $\mathcal{G}$ and latent rules $\mathcal{Z}$. Recalling the reasoning process with rule z as introduced in Section 3, $p_\gamma(z|q)$ is easy to get by checking the head relation and obtain the confidence score of it. Thus we rewrite the Equation (6) as

$$\Psi_\theta(a|G, q) = \frac{1}{|\mathcal{Z}_r|} \sum_{z_r\in\mathcal{Z}_r} \alpha^{z_r} \times p_\tau(a|\mathcal{G}, q, z_r) \tag{7}$$

in which $\mathcal{Z}_r$ is a set of latent rules in KGExplainer for relation r and α^{z_r} is the confidence score of z_r. To normalize the length of rules to T, while enable the variance of length for rules within T steps, a special self-loop relation transforming each entity to itself is added to relation set. Thus with direction of relations considered (inverse relations are considered), there are $|\mathcal{Z}_r| = (2 \times |\mathcal{R}| + 1)^T$ possible rules for r. Our target is to formulate the problem in the context of KGE, thus with a well-trained KGE, Equation (7) could be approached to with

$$\Psi_\theta(a|G, q) \sim \sum_{z_r \in \mathcal{Z}_r} \alpha^{z_r} \times g(RNN_f(\mathbf{h}, z_r), \mathbf{t}) \tag{8}$$

where $\mathbf{h}$ and $\mathbf{t}$ are the embedding of $h \in q$ and answer t. Function RNN_f is a recurrent network defined based on f from KGE that $\mathbf{s}_i$, the hidden state at ith step, is

$$\mathbf{s}_i = f(\mathbf{s}_{i-1}, \mathbf{r}^{(i)}_{z_r}), i \in [1, T], \mathbf{s}_0 = \mathbf{h} \tag{9}$$

where $\mathbf{r}^{(i)}_{z_r}$ is the emebdding of the ith body relation in rule z. For example, suppose KGE is TransE(Bordes et al., 2013) and there is a rule $z_r = [r^{(1)}_{z_r}, r^{(2)}_{z_r}, r^{(3)}_{z_r}]$, for a query $q = (h, r, ?)$, the predicted tail entity embedding given by z_r with TransE(Bordes et al., 2013) and RotatE(Sun et al., 2019) is

$$RNN_{f_{transe}}(\mathbf{h}, z_r) = \left(\left(\left(\mathbf{h} + \mathbf{r}^{(1)}_{z_r} \right) + \mathbf{r}^{(2)}_{z_r} \right) + \mathbf{r}^{(3)}_{z_r} \right) \tag{10}$$

$$RNN_{f_{rotate}}(\mathbf{h}, z_r) = \left(\left(\left(\mathbf{h} \otimes \mathbf{r}^{(1)}_{z_r} \right) \otimes \mathbf{r}^{(2)}_{z_r} \right) \otimes \mathbf{r}^{(3)}_{z_r} \right) \tag{11}$$

where $\otimes$ indicates the rotation operation in complex vector space. Thus the target of KGExplainer is to learn the confidence score α^{z_r} and the structures of rules $z_r = [r^{(1)}, r^{(2)}, ..., r^{(T)}]$, while this is difficult because of the confidence score is associated with particular rule, and enumerating rules is an inherently discrete task(Yang et al., 2017). Thus similarly, we make the undiferentiable rule-dependent summation in Equation (8) into a differentiable step-dependent summation that

$$\Psi_\theta(a|G, q) \sim g\left(RNN_f\left(\mathbf{h}, z'_r\right), \mathbf{t}\right), \text{ where } \mathbf{r}^{(i)}_{z'_r} = \sum_{j=1}^{|\mathcal{R}|} \alpha^r_{ij} \times \mathbf{r}_j \tag{12}$$

where $z'_r = [\mathbf{r}^{(1)}_{z'_r}, \mathbf{r}^{(2)}_{z'_r}, ..., \mathbf{r}^{(T)}_{z'_r}]$ that $\mathbf{r}^{(i)}_{z'_r}$ is the embedding of expected relation embedding at the ith step, and α^r_{ij} is the confidence score of the jth relation r_j as the ith body relation in z_r. Thus currently, the goal of KGExplainer is to learn the confidence scores similarly to what did in NeuralLP(Yang et al., 2017), while instead of keeping the summation operation which is expensive for large knowledge graphs with huge amount of relations, we take a more efficient way to directly decode the compositive relation embedding $\mathbf{r}^{(i)}_{z'}$, the results of summation operation. With $\mathbf{r}^{(i)}_{z'_r}$, α^r_{ij} could be calculated in a post hoc manner based on the basic similarity assumption in vector space.

Latent Rule Decoder $\mathcal{D}$. In KGExplainer, for each relation, we learn a rule embedding to represent the latent rule for it, denoted as $\mathbf{z}'_r$, which is assumed to encode the semantic of z_rs. The sequence of compositive body relation is decoded by a latent rule decoder $\mathcal{D}$, and such sequential output can be effectively modeled by recurrent neural networks

$$\mathbf{r}^{(i)}_{z'_r} = \mathbf{b}^{(i)}_{z'_r} = LSTM(\mathbf{z}'_r \oplus \mathbf{r}^{(i-1)}_{z'_r}, \mathbf{b}^{(i-1)}_{z'_r}), \quad \mathbf{b}^0_{z'_r} = \mathbf{0} \tag{13}$$

where the $\oplus$ means concatenation of two vectors. Finally, the output is a sequence of compositive relation embeddings $[\mathbf{r}^{(1)}_{z'_r}, \mathbf{r}^{(2)}_{z'_r}, ..., \mathbf{r}^{(T)}_{z'_r}]$ for body of z'_r to be applied to Equation (12). The algorithm of learning KGExplainer is shown in Algorithm 1.

Symbolic Rule Decoding and Explanation. Decoding in a post hoc manner, getting the confidence score α^r_{ij} in Equation (12) is the key target. KGEs, as representation learning methods for KGs, could discover entity and relation similarities automatically, thus the decoding is based on similarity between relation embeddings. Suppose the similarity is calculated between $\mathbf{r}^{(i)}_{z'_r}$ and relation embedding matrix $\mathbf{R}$ from the KGE, to make α^r_{ij} consistent to the semantic of compositive relation embeddings $\mathbf{r}^{(i)}_{z'_r}$, there is a basic assumption that

$$\sum_{j=1}^{|\mathcal{R}|} \alpha^r_{ij} f(\mathbf{h}, \mathbf{R}_j) = f(\mathbf{h}, \sum_{j=1}^{|\mathcal{R}|} \alpha^r_{ij} \mathbf{R}_j)) \tag{14}$$

Algorithm 1: Learning KGExplainer.

Input: Training set $S = \{(h, r, t)\}$, a well trained KGE with relation embeddings $\mathbf{R}$, entity embeddings $\mathbf{E}$, translation and similarity function f and g, max step M, rule length T

// *Learning KGExplainer*;
Randomly initialize rule embeddings $\mathbf{Z}$ and parameters in LSTM;
$m \leftarrow 0$;
for $m < M$ **do**
  $S_{batch} \leftarrow$ sample(S, b) // sample a minibatch of size b;
  $Z \leftarrow D(\mathbf{Z})$ //decode body relation embeddings according to Equation (13);
  for $(h, r, t) \in S_{batch}$ **do**
    $\mathbf{b} = \mathbf{h}$ // initialize the answer as head entity embedding;
    t=0;
    while $t < T$ **do**
      $\mathbf{b} \leftarrow f(\mathbf{b}, \mathbf{Z}_r^{(t)})$ // the tth step reasoning;
    end
    $s_{kgexplainer}(h, r, t) \leftarrow g(\mathbf{b}, \mathbf{t})$ // get the score of KGExplainer;
    $s_{kge}(h, r, t) \leftarrow g(f(\mathbf{h}, \mathbf{r}), \mathbf{t})$ // get the score from target KGE;
  end
  Update parameters w.r.t. $\sum_{(h,r,t)\in S_{batch}} ||s_{kge}(h, r, t) - s_{kgexplainer}(h, r, t)|| + L(h, r, t)$
end

which means given a head entity embedding $\mathbf{h}$, the weighted sum of $|\mathcal{R}|$ predictions following all relation embeddings in $\mathbf{R}$ should be equal to the the prediction calculated with $\mathbf{h}$ and the weighted sum of all relations in $\mathbf{R}$. This assumption ensures that $\mathbf{r}_{z_r'}^{(i)} \sim \sum_{j=1}^{|\mathcal{R}|} \alpha_{ij}^r \mathbf{R}_j$ which is important for regarding $\mathbf{r}_{z_r}^{(i)}$ as a compositive relation embedding. While this assumption is challenging to satisfy because the smoothness assumption is not always hold for translation function f that $\mathbf{r} \approx \mathbf{r}'$ do not generally imply $f(\mathbf{h}, \mathbf{r}) \approx f(\mathbf{h}, \mathbf{r}')$. Thus we propose to separate f into two functions, that

$$f(\mathbf{h}, \mathbf{r}) = f_1(\mathbf{h}, f_2(\mathbf{r})), \text{ where } \sum_{j=1}^{|\mathcal{R}|} \alpha_{ij} f_1(\mathbf{h}, f_2(\mathbf{R}_j)) = f_1(\mathbf{h}, \sum_{j=1}^{|\mathcal{R}|} \alpha_{ij} f_2(\mathbf{R}_j))) \tag{15}$$

where f_1 is the function that satisfy the assumption in Equation (14), and f_2 is a function to transform relation emebddings. The separation of f_1 and f_2 for different KGEs depend on their assumptions. For TransE, it is simple that $f_1(\mathbf{h}, \mathbf{r}') = \mathbf{h} + \mathbf{r}'$ and $f_2(\mathbf{r}) = \mathbf{r}$, and for RotatE, $f_1(\mathbf{h}, \mathbf{r}') = \mathbf{h} \circ \mathbf{r}'$ $f_2(\mathbf{r}) = cos(\mathbf{r}) + sin(\mathbf{r})i$ where $\circ$ denotes the Hadamard (element-wise) product. We provide the proof in Appendix.

As introduced before, there $(|\mathcal{R}| \times 2 + 1)^T$ possible rules with different rule body for inferring relation r, thus inverse relation emebddings $\mathbf{r}^{-1}$ and self-loop relation embedings $\mathbf{r}^\circ$ should be added into relation embedding matrix for KGEs based on their assumptions. For both TransE and RotatE, $\mathbf{r}^{-1} = -\mathbf{r}$, and $\mathbf{r}^\circ = \mathbf{0}$. We provide the proof in Appendix.

Given latent rule embedding $\mathbf{z}_r'$ for relation r and its decoded body sequence $[\mathbf{r}_{z_r'}^{(1)}, \mathbf{r}_{z_r'}^{(2)}, ..., \mathbf{r}_{z_r'}^{(T)}]$, the confidence score is calculated via dot product $\alpha_{ij}^r = f_2(\mathbf{r}_z^{(i)}) \cdot f_2(\mathbf{R}_j)$ And finally, for a rule $z_r = [r_{z_r}^{(1)}, r_{z_r}^{(2)}, ..., r_{z_r}^{(T)}]$, the confidence score given by KGExplainer is $\alpha_{z_r} = \alpha_{1(1)} \times \alpha_{2(2)} \times ... \times \alpha_{T(T)}$. According to the confidence scores, a ranked list of rules $\mathcal{Z}_r$ in descending order for each relation could be generated. For a prediction task $q = (h, r, ?)$, suppose the answer from KGExplainer is $a = t$, we present the one top path from h to t grounded according to $\mathcal{Z}_r$ as explanation.

5 EXPERIMENT

During the experiment, we apply KGExplainer to TrasnE(Bordes et al., 2013) and RotatE(Sun et al., 2019) and evaluate it from two perspectives, one is faithfulness evaluation showing to what extend that KGExplainer duplicate the behavior of target KGEs, the other one is explanation evaluation to show the quality of explanations KGExplainer provides. All evaluations are conducted on two

Table 2: Link prediction results on *WN18RR* and *FB15K-237*.

Method	WN18RR				FB15K-237			
	MRR	Hit @1	Hit @3	Hit @10	MRR	Hit @1	Hit @3	Hit @10
NeuralLP	.435	.371	.434	.566	.2402	.1769	.2618	.3623
DRUM	.486	.425	.513	.586	.343	.255	.378	.516
MINERVA	.448	. 413	.456	.513	.293	.217	.329	.456
TransE	.2224	.0128	.3993	.5298	.3294	.2300	.3686	.5272
KGExplainer-TransE	.2225±.0001	.0128±.0001	.3992±.0001	.5297±.0000	.3294±.0001	.2302±.0002	.3683±.0002	.5269±.0002
RotatE	.4767	.4296	.4952	.5737	.3361	.2397	.3731	.5301
KGExplainer-RotatE	.4768±.0001	.4297±.0002	.4951±.0003	.5737±.0002	.3362±.0001	.2397±.0001	.3733±.0002	.5301±.0003
Avg. dif.	.0001±.0001	.0001±.0002	.0001±.0003	.0001±.0002	.0001±.0001	.0001±.0002	.0003±.0002	.0002±.0003

benchmark datasets for link prediction that widely used in KGE works, WN18RR and FB15k-237. Their statsitics are shown in Table 1.

5.1 Faithfulness evaluation

One of the desiderata of an explaining method is to reliably and comprehensively represent the local decision structure of the target model, which means, in the context of KGEs, KGExplainer should replicate the behavior of KGEs as much as possible. Thus we first compare the link prediction results of KGExplainer and the target KGEs and then show how distillation loss changes during training.

Table 1: Datasets statistics.

	WN18RR	FB15k-237
# Train	86,835	272,155
# Valid	3,034	17,535
# Test	3,134	20,466
# Relation	11	237
# Entity	40,943	14,541

Setup and implementation details. KGEs are trained with code provided by Sun et al. (2019)[1] following the best config recommendation on two datasets. For KGExplainer, we set the length of rules $T = 2$, and the rule embedding dimension as 2 times over relation embedding dimension. Other KGE related hyperparameters are set consistent to the target KGEs. Link prediction results are measured by the common matrices Mean Reciprocal Rank (MRR), and Hit Ratio (Hit@$x(x = 1/3/10)$), over head entity prediction and tail entity prediction. In order to enable head entity prediction, we inverse the order of rule body relation embeddings from rule decoder and replace each of them with it's inverse relation embedding, specifically, for query $(?, r, t)$, $[(\mathbf{r}_{z_r}^{(T)})^{-1}, ..., (\mathbf{r}_{z_r}^{(2)})^{-1}, (\mathbf{r}_{z_r}^{(1)})^{-1}]$ are applied as z' in Equation (8). For KGExplainer , we repeat the experiment for 3 times, and report the average results.

Result Analysis. The link prediction results of KGExplainer and target KGEs in Table 2 shows that KGExplainer successfully achieves nearly the same performance on link prediction task as the target KGE, with average performance differences among three matrices vary from 0.0001 to 0.0003 on WN18RR and FB15k-237. While considering that different KGEs models might achieve similar results as shown in Sun et al. (2019), to take a deep look at the differences between KGExplainer and KGEs, in Figure 3, we show the changes of distillation loss during training of KGExplainer for each dataset with two target KGEs. The distillation loss decreases quickly from a high value close to 1 to almost 0, showing that for each training sample, both positive and negative ones, KGExplainer gives nearly the same scores as the target KGE, supporting that KGExplainer is faithful to target KGEs and is a reliable and successful replicator of target KGEs.

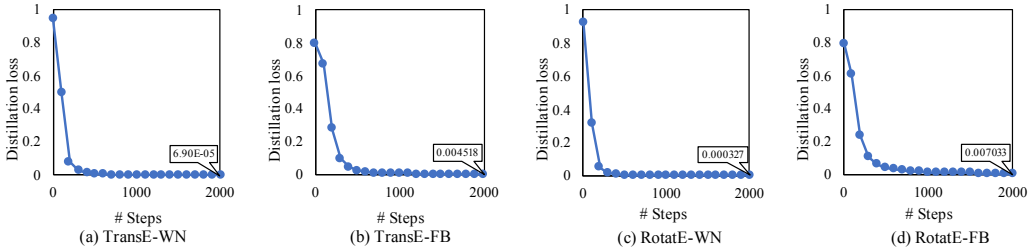


Figure 3: Distillation loss during training, where "WN" and "FB" are abbreviations for WN18RR and FB15k-237, respectively.

[1] https://github.com/DeepGraphLearning/KnowledgeGraphEmbedding

5.2 EXPLANATION EVALUATION

It is important to reach an objective assessment of how good an explanation is for KGExplainer. Unfortunately, similar to other areas, evaluating explanations is made difficult by the impossibility of collecting "ground-truth" explanations, which is possibly to be built only with experts understanding how the model decides(Miller, 2019). Thus instead of end-to-end evaluation schemes, we take an alternative way to first evaluate the quality of latent rules from KGExplainer and then show case study of explanations.

Evaluation metrics. With a decreasingly ranked list of rules for each relation according to the confidence score given by KGExplainer, we calculate the average statistical confidence score of each rule to show the quality of them. Statistical confidence are

$$conf(r(x,y) \leftarrow B) := \frac{\#(x,y) : \exists e_1, ..., e_m : B \wedge r(x,y)}{\#(x,y) : \exists e_1, ..., e_m : B} \tag{16}$$

where denominator is the number of head and tail entity pairs satisfying the rule body B, and the numerator is the number of head and tail entity pairs satisfying both the rule body B and rule head $r(x, y)$. Statistical confidence reflects how common that the entity pairs satisfy rule body and also satisfy rule head, and is a measurement of rule quality from statistical point of view. Given a ranked list of rules, generally, a higher average statistical confidence score means rules contained in the list with better quality. Since the higher quality rules are expected to be ranked at the top of the list, we report the average statistical confidence of Top$k(k = 1, 3, 5, 10)$ rules, denoted as $AvgConf_k = \frac{1}{|\mathcal{R}|\times k} \sum_{z_r \in \mathcal{Z}_r[:k]} Conf(z_r)$ in which $\mathcal{Z}_r[: k]$ is the Top k rules list for relation r provided by a model.

Baseline Approaches. We proposes two statistical baselines for $AvgConf_k$ comparison to KGExplainer, *Traverse-Avg* and *RAR (Resource Allocation over Relations)*. Apart from statistical baselines, we also compare to scoring the traverse paths via TransE and RotatE, and also differentiable rule learning methods *NeuralLP* and *DRUM*.

Traverse. Given a relation r, we traverse all relations at each step resulting in $(|\mathcal{R}| \times 2 + 1)^T$ rules of all possible combinations with relation direction considered. And their average statistical confidence are reported as results. Thus Top$k$ are the same with different k in *Traverse-Avg*. This baseline shows overall average confidence of all combinations. Other methods should achieve better results than *Traverse-Avg* as long as they are effective at learning rules. Based on traversed paths, we also report *Trverse-TransE* and *Trverse-RotatE* as baseline (with inverse and selfloop relation added and their embeddings are got in the same way as KGExplainer) in which we score them via similarity between compositional representation of paths representation and target relation embedding following (Yang et al., 2015; Guu et al., 2015; Zhang et al., 2019).

RAR (Resource Allocation over Relation). Following the idea of resource allocation over networks(Lü & Zhou, 2011) and PCRA algorithm(Lin et al., 2015a), we propose a method for resource allocation over relations with the frequency of connection between two relations within consideration. Specifically, more resources (higher probability) should be assigned to more frequently linked relations. Given two relation r_1 and r_2, the probability of choosing r_2 as the next step of r_1 in the rule body that $\xrightarrow{r_1} \circ \xrightarrow{r_2}$ is defined as

$$p(r_1|r_2, \mathcal{G}) = \frac{\sum_{e \in \mathcal{G}} C^t_{(r_1,e)} \times C^h_{(r_2,e)}}{\sum_{r \in \mathcal{G}} \sum_{e \in \mathcal{G}} C^t_{(r_1,e)} \times C^h_{(r,e)}} \tag{17}$$

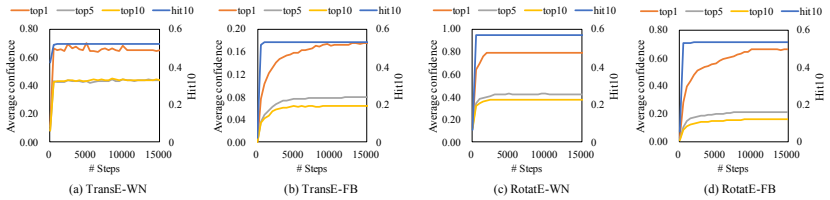
where $C^t_{(r,e)}$ is the times of entity e as the tail entity of r in $\mathcal{G}$, counted over train dataset, and $C^h_{(r,e)}$ is the the times of e as the head entity of r. Finally, *RAR* will rank the most frequent path at the top. This baseline is a naive but powerful solution for rule mining similar to the idea of biased random walk(Liu & Lü, 2010).

NeuralLP is the first differential rule learning method which could generate a probability for each relation as the relation in each step of the rule body, and finally the confidence score of each rules, of the same semantic as the α in our method, could be calculated. For one relation r, the ranked list of rules for it is decoded following the rule decoding algorithm proposed in NeuralLP(Yang et al., 2017) and DRUM(Sadeghian et al., 2019), with threshold set as 0.0001 to ensure enough number of rules for each relation.

Table 3: $AvgConf_k$ results on *WN18RR* and *FB15K-237*.

Method	WN18RR			FB15K-237		
	Top1	Top5	Top10	Top1	Top5	Top10
Traverse-Avg	.1123	.1123	.1123	.0032	.0032	.0032
RAR	.2939	.2673	.2371	.0649	.0689	.0649
NeuralLP	.6931	.5228	.4376	.0510	.0477	.0457
DRUM	.6163	**.5694**	**.5193**	.0703	.0531	.0471
Traverse-TransE	.0909	.1089	.0872	.0129	.0026	.0013
Traverse-Rotate	.0599	.0236	.0527	.0000	.1582	.1306
KGExplainer-TransE	.6569±.0008	.5077±.0007	.4289±.0008	.1715±.0098	.0960±.0034	.0794±.0031
KGExplainer-RotatE	**.7897**±.2166	.4786±.0711	.4293±.0578	**.6660**±.0849	**.2968**±.0436	**.2173**±.0347

Result Analysis. As shown in Table 3, for WN18RR, NeuralLP, KGExplainer-TransE and KGExplainer RotatE achieves comparable quality of rules, in which DRUM performs best on Top5 and Top10 and KGExplainer-RotatE is the best on Top1. And for FB15k-237, KGExplainer-RotatE shows a significantly better quality of rules than other methods. If we have a look at the link prediction results in Table 2, we could find that generally the rule quality provided by these models are consistent to their link prediction results, for example, on FB15k-237, RotatE and KGExplainer-RotatE performs the best, then is TransE and KGExplainer-TransE, and finally is DRUM and NeuralLP, which is the same in Table 3. This illustrates the effectiveness of KGExplainer as an explainer for KGEs. Results of Traverse-TransE and Traverse-RotatE is significantly worse than KGExplainer, showing that getting a good rank of rules that reflect how KGE makes predictions is nontrivial. The score variance of Top1 to Top10 is gradually decreasing, showing that the average quality of Top10 rules are stable while the Top1 rule might be diverse.



(a) TransE-WN (b) TransE-FB (c) RotatE-WN (d) RotatE-FB

Figure 4: *AvgConf* $_k$ and the link prediction results on valid dataset at different steps during training.

Figure 4 shows the changes of link prediction results $Hit10$ and $AvgConf_k$ of rules during training. It is interesting that for FB15k-237, the link prediction result of Hit10 gets stable after a few steps, while the quality of rules keeps increasing, showing that even if the link prediction results keeps the same, the latent rule quality could be improved further. And for WN18RR, this phenomenon is less obvious which might be because of the small combination space over 11 relations for path-like rules.

Case Study. Figure 5 shows a case study for prediction $q = (00621734, \_derivationally\_related\_form)$ and $a = 05685030$, where KGExplainer-TransE and KGExplainer-RotatE rank the answer at the 3rd and 1st one over 40943 candidate entities respectively. The two explanations are both composed by *_derivationally_related_form* and *_hypernym*, while with different directions for relations. The statistical confidence of rules from KGExplainer-RotatE which the explanations are grounded according to is higher than KGExplainer-TransE, this ensures a better prediction. The bar chart shows the average confidence for tail entities ranked in different intervals, in which the average confidence of predictions ranked in top5 has a significantly higher average confidence than those out of the top 5, which means that the average confidence of correct predictions is higher than incorrect ones.

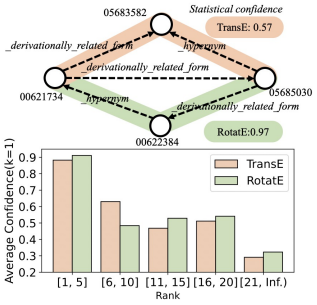


Figure 5: Analysis from WN18RR.

6 Conclusion and Future Work

In this paper, we propose to explain KGE via latent rule learning and propose a framework KGExplainer which works well on explaining TransE and RotatE. In KGExplainer, we learn a latent rule embedding for each relation and train a rule decoder with rule embedding as input to output the compositive body relation embedding for link prediction, based on which KGExplainer achieves almost the same link prediction results as the target KGE. Finally, after training, we decode a ranked list of symbolic rules from compositive body relation embedding based on similarity in vector space for providing explanations. The rule quality from KGExplainer is proved consistent to the capability of target KGEs As a possible future work we would like to adapt KGExplainer for more KGEs with variance rule length, to make it a more general and flexible framework.

REFERENCES

Ralph Abboud, İsmail İlkan Ceylan, Thomas Lukasiewicz, and Tommaso Salvatori. Boxe: A box embedding model for knowledge base completion. In *NeurIPS*, 2020.

K. M. Annervaz, Somnath Basu Roy Chowdhury, and Ambedkar Dukkipati. Learning beyond datasets: Knowledge graph augmented neural networks for natural language processing. In *NAACL-HLT*, pp. 313–322. Association for Computational Linguistics, 2018.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, pp. 2787–2795, 2013.

Zongsheng Cao, Qianqian Xu, Zhiyong Yang, Xiaochun Cao, and Qingming Huang. Dual quaternion knowledge graph embeddings. In *AAAI*, pp. 6894–6902. AAAI Press, 2021.

Ines Chami, Adva Wolf, Da-Cheng Juan, Frederic Sala, Sujith Ravi, and Christopher Ré. Low-dimensional hyperbolic knowledge graph embeddings. In *ACL*, pp. 6901–6914. Association for Computational Linguistics, 2020.

William W. Cohen, Fan Yang, and Kathryn Mazaitis. Tensorlog: A probabilistic database implemented using deep-learning infrastructure. *J. Artif. Intell. Res.*, 67:285–325, 2020.

Rajarshi Das, Shehzaad Dhuliawala, Manzil Zaheer, Luke Vilnis, Ishan Durugkar, Akshay Krishnamurthy, Alex Smola, and Andrew McCallum. Go for a walk and arrive at the answer: Reasoning over paths in knowledge bases using reinforcement learning. In *ICLR (Poster)*. OpenReview.net, 2018.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. Convolutional 2d knowledge graph embeddings. In *AAAI*, pp. 1811–1818. AAAI Press, 2018.

Ruth C. Fong and Andrea Vedaldi. Interpretable explanations of black boxes by meaningful perturbation. In *ICCV*, pp. 3449–3457. IEEE Computer Society, 2017.

Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6):1789–1819, 2021.

Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Dino Pedreschi, Franco Turini, and Fosca Giannotti. Local rule-based explanations of black box decision systems. *CoRR*, abs/1805.10820, 2018.

Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. A survey of methods for explaining black box models. *ACM Comput. Surv.*, 51(5): 93:1–93:42, 2019.

Arthur Colombini Gusmão, Alvaro Henrique Chaim Correia, Glauber De Bona, and Fábio Gagliardi Cozman. Interpreting embedding models of knowledge bases: A pedagogical approach. *CoRR*, abs/1806.09504, 2018.

Kelvin Guu, John Miller, and Percy Liang. Traversing knowledge graphs in vector space. In *EMNLP*, pp. 318–327. The Association for Computational Linguistics, 2015.

Xiao Huang, Jingyuan Zhang, Dingcheng Li, and Ping Li. Knowledge graph embedding based question answering. In *WSDM*, pp. 105–113. ACM, 2019.

Himabindu Lakkaraju, Ece Kamar, Rich Caruana, and Jure Leskovec. Interpretable & explorable approximations of black box models. *CoRR*, abs/1707.01154, 2017.

Yankai Lin, Zhiyuan Liu, Huan-Bo Luan, Maosong Sun, Siwei Rao, and Song Liu. Modeling relation paths for representation learning of knowledge bases. In *EMNLP*, pp. 705–714. The Association for Computational Linguistics, 2015a.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. Learning entity and relation embeddings for knowledge graph completion. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015b.

Weiping Liu and Linyuan Lü. Link prediction based on local random walk. *EPL (Europhysics Letters)*, 89(5):58007, 2010.

Linyuan Lü and Tao Zhou. Link prediction in complex networks: A survey. *Physica A: statistical mechanics and its applications*, 390(6):1150–1170, 2011.

Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *NIPS*, pp. 4765–4774, 2017.

Tim Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artif. Intell.*, 267:1–38, 2019.

Grégoire Montavon, Alexander Binder, Sebastian Lapuschkin, Wojciech Samek, and Klaus-Robert Müller. Layer-wise relevance propagation: An overview. In *Explainable AI*, volume 11700 of *Lecture Notes in Computer Science*, pp. 193–209. Springer, 2019.

Yatin Nandwani, Ankesh Gupta, Aman Agrawal, Mayank Singh Chauhan, Parag Singla, and Mausam. Oxkbc: Outcome explanation for factorization based knowledge base completion. In *AKBC*, 2020.

Guanglin Niu, Yongfei Zhang, Bo Li, Peng Cui, Si Liu, Jingyang Li, and Xiaowei Zhang. Rule-guided compositional representation learning on knowledge graphs. In *AAAI*, pp. 2950–2958. AAAI Press, 2020.

Meng Qu, Junkun Chen, Louis-Pascal A. C. Xhonneux, Yoshua Bengio, and Jian Tang. Rnnlogic: Learning logic rules for reasoning on knowledge graphs. In *ICLR*. OpenReview.net, 2021.

Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should I trust you?": Explaining the predictions of any classifier. In *KDD*, pp. 1135–1144. ACM, 2016.

Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. Anchors: High-precision model-agnostic explanations. In *AAAI*, pp. 1527–1535. AAAI Press, 2018.

Ali Sadeghian, Mohammadreza Armandpour, Patrick Ding, and Daisy Zhe Wang. DRUM: end-to-end differentiable rule mining on knowledge graphs. In *NeurIPS*, pp. 15321–15331, 2019.

Wojciech Samek, Grégoire Montavon, Sebastian Lapuschkin, Christopher J. Anders, and Klaus-Robert Müller. Explaining deep neural networks and beyond: A review of methods and applications. *Proc. IEEE*, 109(3):247–278, 2021.

Gregor P. J. Schmitz, Chris Aldrich, and F. S. Gouws. ANN-DT: an algorithm for extraction of decision trees from artificial neural networks. *IEEE Trans. Neural Networks*, 10(6):1392–1401, 1999.

Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda B. Viégas, and Martin Wattenberg. Smooth-grad: removing noise by adding noise. *CoRR*, abs/1706.03825, 2017.

Josua Stadelmaier and Sebastian Padó. Modeling paths for explainable knowledge base completion. In *BlackboxNLP@ACL*, pp. 147–157. Association for Computational Linguistics, 2019.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. Rotate: Knowledge graph embedding by relational rotation in complex space. In *ICLR (Poster)*. OpenReview.net, 2019.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *ICML*, volume 70 of *Proceedings of Machine Learning Research*, pp. 3319–3328. PMLR, 2017.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. In *ICML*, volume 48 of *JMLR Workshop and Conference Proceedings*, pp. 2071–2080. JMLR.org, 2016.

Guojia Wan, Shirui Pan, Chen Gong, Chuan Zhou, and Gholamreza Haffari. Reasoning like human: Hierarchical reinforcement learning for knowledge graph reasoning. In *IJCAI*, pp. 1926–1932. ijcai.org, 2020.

Po-Wei Wang, Daria Stepanova, Csaba Domokos, and J. Zico Kolter. Differentiable learning of numerical rules in knowledge graphs. In *ICLR*. OpenReview.net, 2020.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph embedding by translating on hyperplanes. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014.

Chi-Man Wong, Fan Feng, Wen Zhang, Chi-Man Vong, Hui Chen, Yichi Zhang, Peng He, Huan Chen, Kun Zhao, and Huajun Chen. Improving conversational recommender system by pretraining billion-scale knowledge graph. In *ICDE*, pp. 2607–2612. IEEE, 2021.

Wenhan Xiong, Thien Hoang, and William Yang Wang. Deeppath: A reinforcement learning method for knowledge graph reasoning. In *EMNLP*, pp. 564–573. Association for Computational Linguistics, 2017.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. In *ICLR (Poster)*, 2015.

Fan Yang, Zhilin Yang, and William W. Cohen. Differentiable learning of logical rules for knowledge base reasoning. In *NIPS*, pp. 2319–2328, 2017.

Wen Zhang, Bibek Paudel, Liang Wang, Jiaoyan Chen, Hai Zhu, Wei Zhang, Abraham Bernstein, and Huajun Chen. Iteratively learning embeddings and rules for knowledge graph reasoning. In *WWW*, pp. 2366–2377. ACM, 2019.

Luisa M. Zintgraf, Taco S. Cohen, Tameem Adel, and Max Welling. Visualizing deep neural network decisions: Prediction difference analysis. In *ICLR (Poster)*. OpenReview.net, 2017.

A APPENDIX

This section presents the proofs of 1) self-loop relation embedding, 2) inverse relation embedding and 3) f_1 and f_2 function separation of translation function f for TransE and RotatE. Before starting, we first briefly introduce the TransE and RotatE.

Given a knowledge graph, TransE embed entities and relations as vectors in low-dimensional vector space. Thue for a triple (h, r, t), elements' embedding are represented as $\mathbf{h} \in \mathbb{R}^d, \mathbf{r} \in \mathbb{R}^d$ and $\mathbf{t} \in \mathbb{R}^d$ respectively. For a positive triple (h, r, t), TransE assumes that the head entity embedding $\mathbf{h}$ could be translated to $\mathbf{t}$ according to $\mathbf{r}$ that $\mathbf{h} + \mathbf{r} = \mathbf{t}$. Thus

$$\Phi_{transe} = g_{transe}(f_{transe}(\mathbf{h}, \mathbf{r}), \mathbf{t}) \tag{18}$$

$$f_{transe}(\mathbf{h}, \mathbf{r}) = \mathbf{t}' = \mathbf{h} + \mathbf{r} \tag{19}$$

$$g_{transe}(\mathbf{t}, \mathbf{t}') = \|\mathbf{t} - \mathbf{t}'\| \tag{20}$$

For a triple (h, r, t), in RotatEm entities are embeded in complex vector space that $\mathbf{h} \in \mathbb{C}$ and relation as rotation in complex space which is calculated from $\mathbf{r} \in \mathbb{R}^d$. Thus

$$\Phi_{rotate} = g_{rotate}(f_{rotate}(\mathbf{h}, \mathbf{r}), \mathbf{t}) \tag{21}$$

$$f_{rotate}(\mathbf{h}, \mathbf{r}) = \mathbf{t}' = \mathbf{h} \circ (cos(\mathbf{r}) + sin(\mathbf{r})i)) \tag{22}$$

$$g_{rotate}(\mathbf{t}, \mathbf{t}') = \|\mathbf{t} - \mathbf{t}'\| \tag{23}$$

A.1 SELF-LOOP RELATION EMBEDDING.

Embedding for self-loop relation $\mathbf{r}^\circ$ could be inferred by the vector space assumption of KGEs. Since self-loop relation means an entity is linked to itself through this relation that (e, r°, e), thus

- **In TransE,** $\mathbf{e} + \mathbf{r}^\circ = \mathbf{e}$. Thus $\mathbf{r}^\circ = \mathbf{0}$ in TransE.
- **In RotatE**, $\mathbf{e} \circ (cos(\mathbf{r}^\circ) + sin(\mathbf{r}^\circ)i) = \mathbf{e}$. Suppose $\mathbf{e} = \mathbf{e}_r + \mathbf{e}_i i$, $\mathbf{r}_r^\circ = cos(\mathbf{r}^\circ)$ and $\mathbf{r}_i^\circ = sin(\mathbf{r}^\circ)$, thus

$$\mathbf{e} \circ (cos(\mathbf{r}^\circ) + sin(\mathbf{r}^\circ)i) = (\mathbf{e}_r \circ \mathbf{r}_r^\circ - \mathbf{e}_i \circ \mathbf{r}_i^\circ) + (\mathbf{e}_i \circ \mathbf{r}_r^\circ + \mathbf{e}_r \circ \mathbf{r}_i^\circ)i \tag{24}$$

thus

$$\mathbf{e}_r \circ \mathbf{r}_r^\circ - \mathbf{e}_i \circ \mathbf{r}_i^\circ = \mathbf{e}_r, \text{ and } \mathbf{e}_i \circ \mathbf{r}_r^\circ + \mathbf{e}_r \circ \mathbf{r}_i^\circ = \mathbf{e}_i \tag{25}$$

thus

$$\mathbf{r}_r^\circ = \mathbf{1} = cos(\mathbf{r}^\circ), \mathbf{r}_i = \mathbf{0} sin(\mathbf{r}^\circ) \tag{26}$$

Finally, we could infer that $\mathbf{r}^\circ = \mathbf{0}$ in RotatE.

A.2 INVERSE RELATION EMBEDDING.

The inverse relation r^{-1} of relation r means that when (h, r, t) holds, then (t, r^{-1}, h) also holds. Thus

- **In TransE**, we could get $\mathbf{h} + \mathbf{r} = \mathbf{t}$ and $\mathbf{t} + \mathbf{r}^{-1} = \mathbf{t}$, thus $\mathbf{r}^{-1} = -\mathbf{r}$ in TransE.
- **In RotatE**, we could get that $\mathbf{h} \circ (cos(\mathbf{r}) + sin(\mathbf{r})i) = \mathbf{t}$ and $\mathbf{t} \circ (cos(\mathbf{r})^{-1} + sin(\mathbf{r}^{-1})i) = \mathbf{h}$, and suppose that $\mathbf{h} = \mathbf{h}_r + \mathbf{r}_i i$, $\mathbf{t} = \mathbf{t}_r + \mathbf{t}_i i$, $\mathbf{r}_r = cos(\mathbf{r})$, $\mathbf{r}_i = sin(\mathbf{r})$, $\mathbf{r}_r^{-1} = cos(\mathbf{r}^{-1})$, $\mathbf{r}_i^{-1} = sin(\mathbf{r}^{-1})$ thus

$$\mathbf{h}_r \circ \mathbf{r}_r - \mathbf{h}_i \circ \mathbf{r}_i = \mathbf{t}_r \tag{27}$$
$$\mathbf{h}_r \circ \mathbf{r}_i + \mathbf{h}_i \circ \mathbf{r}_r = \mathbf{t}_i \tag{28}$$
$$\mathbf{t}_r \circ \mathbf{r}_r^{-1} - \mathbf{t}_i \circ \mathbf{r}_i^{-1} = \mathbf{h}_r \tag{29}$$
$$\mathbf{t}_r \circ \mathbf{r}_i^{-1} + \mathbf{t}_i \circ \mathbf{r}_r^{-1} = \mathbf{h}_i \tag{30}$$

thus

$$(\mathbf{t}_r \circ \mathbf{r}_r^{-1} - \mathbf{t}_i \circ \mathbf{r}_i^{-1}) \circ \mathbf{r}_r - (\mathbf{t}_r \circ \mathbf{r}_i^{-1} + \mathbf{t}_i \circ \mathbf{r}_r^{-1}) \circ \mathbf{r}_i = \mathbf{t}_r \tag{31}$$
$$(\mathbf{t}_r \circ \mathbf{r}_r^{-1} - \mathbf{t}_i \circ \mathbf{r}_i^{-1}) \circ \mathbf{r}_i + (\mathbf{t}_r \circ \mathbf{r}_i^{-1} + \mathbf{t}_i \circ \mathbf{r}_r^{-1}) \circ \mathbf{r}_r = \mathbf{t}_i \tag{32}$$

thus

$$\mathbf{t}_r \circ (\mathbf{r}_r^{-1} \circ \mathbf{r}_r - \mathbf{r}_i^{-1} \circ \mathbf{r}_r) - \mathbf{t}_i (\mathbf{r}_i^{-1} \circ \mathbf{r}_i + \mathbf{r}_r^{-1} \circ \mathbf{r}_i) = \mathbf{t}_r \tag{33}$$
$$\mathbf{t}_r \circ (\mathbf{r}_r^{-1} \circ \mathbf{r}_i + \mathbf{r}_i^{-1} \circ \mathbf{r}_r) + \mathbf{t}_i (\mathbf{r}_r^{-1} \circ \mathbf{r}_r - \mathbf{r}_i^{-1} \circ \mathbf{r}_r) = \mathbf{t}_i \tag{34}$$

thus

$$\mathbf{r}_r^{-1} \circ \mathbf{r}_r - \mathbf{r}_i^{-1} \circ \mathbf{r}_r = \mathbf{1} \tag{35}$$
$$\mathbf{r}_r^{-1} \circ \mathbf{r}_i + \mathbf{r}_i^{-1} \circ \mathbf{r}_r = \mathbf{0} \tag{36}$$
$$\therefore cos(\mathbf{r}^{-1}) cos(\mathbf{r}) - sin(\mathbf{r}^{-1}) \circ sin(\mathbf{r}) = \mathbf{1} \tag{37}$$
$$cos(\mathbf{r}^{-1}) \circ sin(\mathbf{r}) + sin(\mathbf{r}^{-1}) \circ cos(\mathbf{r}) = \mathbf{0} \tag{38}$$

thus

$$cos(\mathbf{r}^{-1}) = cos(\mathbf{r}), sin(\mathbf{r}^{-1}) = -sin(\mathbf{r}) \tag{39}$$

Finally, we could get that $\mathbf{r}^{-1} = -\mathbf{r}$ in RotatE.

A.3 f_1 AND f_2 SEPARATION.

In this section, we prove that the separation of f_1 and f_2 for KGEs satisfy Equation (15).

- **In TransE,** we set $f_2(\mathbf{r}) = \mathbf{r}$ and $f_1(\mathbf{h}, \mathbf{r}') = \mathbf{h} + \mathbf{r}'$, suppose α_i is the weight for ith relation

$$\sum_{i=1}^{|\mathcal{R}|} \alpha_i f_1(\mathbf{h}, f_2(\mathbf{r})) = \sum_{i=1}^{|\mathcal{R}|} \alpha_i (\mathbf{h} + \mathbf{r}) \tag{40}$$

$$= \sum_{i=1}^{|\mathcal{R}|} \alpha_i \mathbf{h} + \sum_{i=1}^{|\mathcal{R}|} \alpha_i \mathbf{r} \tag{41}$$

$$= \mathbf{h} + \sum_{i=1}^{|\mathcal{R}|} \alpha_i \mathbf{r}, \text{ if } \sum_{i=1}^{|\mathcal{R}|} \alpha_i = 1 \tag{42}$$

$$= f_1(\mathbf{h}, \sum_{i=1}^{|\mathcal{R}|} \alpha_i f_2(\mathbf{r})) \tag{43}$$

  Thus such separation of f_1 and f_2 in TransE satisfy Equation (15).

- **In Rotate,** we set $f_2(\mathbf{r}) = cos(\mathbf{r}) + sin(\mathbf{r})$ and $f_1(\mathbf{h}, \mathbf{r}') = \mathbf{h} \circ \mathbf{r}'$, suppose α_i is the weight for ith relation

$$\sum_{i=1}^{|\mathcal{R}|} \alpha_i f_1(\mathbf{h}, f_2(\mathbf{r})) = \sum_{i=1}^{|\mathcal{R}|} \alpha_i (\mathbf{h} \circ (cos(\mathbf{r}) + sin(\mathbf{r}))) \tag{44}$$

$$= \mathbf{h} \circ \sum_{i=1}^{|\mathcal{R}|} \alpha_i (cos(\mathbf{r}) + sin(\mathbf{r})) \tag{45}$$

$$= f_1(\mathbf{h}, \sum_{i=1}^{|\mathcal{R}|} \alpha_i f_2(\mathbf{r})) \tag{46}$$

  Thus such separation of f_1 and f_2 in RotatE satisfy Equation (15).