KNOWLEDGE GRAPH COMPLETION AS TENSOR DECOMPOSITION: A GENREAL FORM AND TENSOR N-RANK REGULARIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Knowledge graph completion (KGC) is a 3rd-order binary tensor completion task. Tensor decomposition based (TDB) models have shown great performance in KGC. In this paper, we summarize existing TDB models and derive a general form for them. Based on the general form, we show the principles of model design to satisfy logical rules. However, these models suffer from the overfitting problem severely. Therefore, we propose a regularization term based on the tensor n-rank which enforces the low-rankness of the tensor. First, we relax the tensor n-rank to the sum of the nuclear norms of the unfolding matrix along each mode of the tensor. In order to be computationally efficient, we further give an upper bound of the sum of the nuclear norms. Finally, we use the upper bound as the regularization term to achieve low-rank matrix decomposition of each unfolding matrix. Experiments show that our model achieves state-of-the-art performance on benchmark datasets.

1 INTRODUCTION

A knowledge graph (KG) can be represented by a 3rd-order binary tensor, where each entry corresponds to a triplet with the form $(head\ entity, relation, tail\ entity)$, 1 indicating a known true triplet and 0 indicating a false triplet and ? indicating a missing triplet (either a true or a false triplet). Although commonly used KGs contain a large number of known triplets, they still suffer from the incompleteness problem that a lot of triplets are missing, $i.e.$, the 3rd-order tensor is incomplete. The task of KGC is to predict which of the missing triplets are true or false based on known triplets, $i.e.$, to infer which of the missing entries in the tensor are 1 or 0.

A number of models have been proposed for KGC (Zhang et al., 2021). Among the models, tensor decomposition based (TDB) models achieve the state-of-the-art performance (Trouillon et al., 2017; Kazemi & Poole, 2018). Different types of tensor decomposition methods have been applied to KGC. Lacroix et al. (2018); Kazemi & Poole (2018) developed approaches based on CANDECOMP/PARAFAC (CP) decomposition (Hitchcock, 1927). DisMult (Yang et al., 2014) can be regarded as a particular version of CP that learns a symmetric tensor by sharing the embedding matrices of head and tail entities. In order to learn asymmetric relations, ComplEx (Trouillon et al., 2017) extends DistMult to the complex space and QuatE (Zhang et al., 2019) further explores the hypercomplex space. Balažević et al. (2019a) proposed TuckER based on Tucker decomposition (Tucker, 1966). For a thorough understanding of the thriving TDB models, we summarize mainstream TDB models and derive a general form for them. Based on the general form, we further show the principles of model design to enable the model to learn the symmetric, anti-symmetric and inverse rules.

Theoretically, TDB models are fully expressive (Trouillon et al., 2017; Kazemi & Poole, 2018; Balažević et al., 2019a), which can represent any ground truth tensor. However, their performance usually suffers from the overfitting problem severely. In order to alleviate the overfitting problem of TDB models, various regularizations have been applied to KGC. The squared Frobenius norm regularization is commonly used in KGC (Nickel et al., 2011; Yang et al., 2014; Trouillon et al., 2017). Lacroix et al. (2018) proposed a regularization based on tensor nuclear p-norm and demonstrated better performance than the squared Frobenius norm. Zhang et al. (2020) proposed DURA, a

regularization based on the duality of TDB models and distance based models, achieving significant improvements on benchmark datasets. Moreover, they prove the regularization term is an upper bound of nuclear 2-norm.

In this paper, we propose a regularization term based on the n-rank of tensors. Motivated by the success of low-rank matrix completion (Nguyen et al., 2019), we aim to achieve low-rank KGC. However, the definition of tensor rank is not unique, $e.g.$, CP-rank (Gandy et al., 2011) which is based on CP decomposition and n-rank (Gandy et al., 2011) which is based on Tucker decomposition. Since Tucker decomposition is more general than CP decomposition, we use the tensor n-rank instead of the tensor CP-rank. The tensor n-rank is defined as a vector whose i-th entry is the rank of mode-i unfolding matrix (Kolda & Bader, 2009), which is based on the definition of the matrix rank. Thus, we can minimize the sum of the matrix ranks to realize low-rank KGC. Therefore, the task of low-rank tensor completion is transformed into the task of low-rank matrix completion. Since the matrix rank minimization problem is NP-hard for most problems (Recht et al., 2008), the matrix nuclear norm is widely used as a convex surrogate of the matrix rank. However, the matrix nuclear norm is a complicated non-differentiable function (Rennie & Srebro, 2005) and computing the matrix nuclear norm is expensive for large KG. Therefore, we further propose a computationally efficient upper bound of the sum of the matrix nuclear norm. Finally, we use the upper bound as our regularization term. By utilizing the regularization, we can preserve the expressiveness of TDB models and prevent them from the overfitting problem. It is worth noting that the squared Frobenius norm and DURA (Zhang et al., 2020) become parts of our regularization term. Our reglarization term is suitable for almost all TDB models. Experiments show that our tensor n-rank regularization (TNRR) gains significant improvements on benchmark datasets.

2 Related Work

Tensor Decomposition Based Models There has been extensive research for KGC, we focus here on tensor decomposition based (TDB) models. CP decomposition (Hitchcock, 1927) and Tucker decomposition (Tucker, 1966) are two common decompositions in tensor decomposition. Researchers mainly focus on these two decompositions. CP decomposition represents a tensor as a sum of n rank one tensors. Tucker decomposition (Tucker, 1966) decomposes a tensor into a core tensor and a set of matrices. Lacroix et al. (2018) apply the original CP decomposition to KGC. DisMult (Yang et al., 2014), a variant of CP, which makes the embedding matrices of the head entities and tail entities the same. SimplE (Kazemi & Poole, 2018), another variant of CP, addresses the independence problem among the embedding of head entities and tail entities in CP deompostion. ComplEx (Trouillon et al., 2017) extends DistMult to complex space because DistMult cannot handle asymmetric relations. QuatE (Zhang et al., 2019) further explore the hypercomplex space. Nickel et al. (2016) propose HOLE based on the circular correlation, while Liu et al. (2017) prove HOLE is equivalent to ComplEx. ANALOGY (Liu et al., 2017) explicitly exploits the analogical structures in KG. In addition, Balažević et al. (2019a) propose TuckER based on the Tucker decomposition.

Regularization Although TDB models are fully expressive (Trouillon et al., 2017; Kazemi & Poole, 2018; Balažević et al., 2019a), they usually suffer from the overfitting problem severely. Therefore, many regularization terms have been proposed. The squared Frobenius norm regularization has been used in KGC widely (Nickel et al., 2011; Yang et al., 2014; Trouillon et al., 2017). However, Lacroix et al. (2018) showed that the Frobenius norm regularization does not correspond to regularization with a tensor norm. Therefore, they proposed a N3 regularization term based on tensor nuclear 3-norm, which is an upper bound of tensor nuclear 3-norm. Zhang et al. (2020) proposed DURA, a regularization based on the duality of TDB models and distance based models, which is a upper bound of nuclear 2-norm.

3 Methods

In this section, we describe the technical details. In Section 3.1, we introduce the notations used throughout this paper. In Section 3.2, we summarize the proposed TDB models and derive a general form of TDB models. In Section 3.3, we show the principles of model design to enable the model to learn the symmetric, anti-symmetric and inverse rules. In Section 3.4, we propose our tensor n-rank regularization (TNRR).

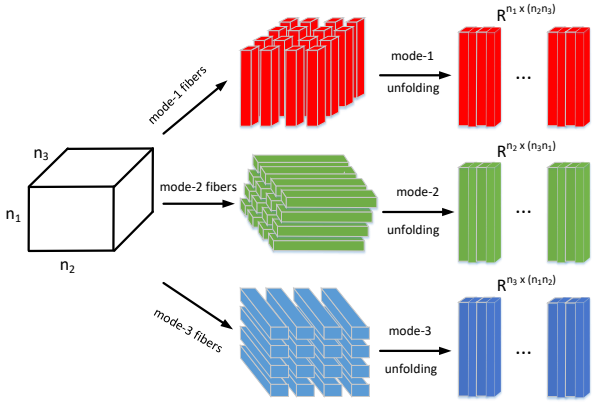

Figure 1: Left shows a 3rd order tensor. Middle describes the corresponding mode-i fibers of the tensor. Fibers are the higher-order analogue of matrix rows and columns. A fiber is defined by fixing every index but one. Right describes the corresponding mode-i unfolding of the tensor. The mode-i unfolding of a tensor arranges the mode-i fibers to be the columns of the resulting matrix.

3.1 Notations

Given a set $\mathcal{E}$ of entities and a set $\mathcal{R}$ of relations, a KG contains a set of triplets $\mathcal{S} = \{(h_i, r_j, t_k)\} \subset \mathcal{E} \times \mathcal{R} \times \mathcal{E}$. Let the corresponding KG tensor be $\mathbf{X} \in \{0,1\}^{|\mathcal{E}|\times|\mathcal{R}|\times|\mathcal{E}|}$, with $\mathbf{X}_{i,j,k} = 1$ if $(h_i, r_j, t_k) \in \mathcal{S}$. Denote $|\mathcal{E}|$ and $|\mathcal{R}|$ as the number of entities and relations. Let $\boldsymbol{H} \in \mathbb{R}^{|\mathcal{E}|\times D}, \boldsymbol{R} \in \mathbb{R}^{|\mathcal{R}|\times D}, \boldsymbol{T} \in \mathbb{R}^{|\mathcal{E}|\times D}$ be the embedding matrices of head entities, relation and tail entities, where D is the embedding dimension. Let $\boldsymbol{H}_i, \boldsymbol{R}_i, \boldsymbol{T}_i$ be the ith head entity, relation and tail entity. Denote $< \boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c} >= \sum_{i=1}^{D} \boldsymbol{a}_i \boldsymbol{b}_i \boldsymbol{c}_i$ as the triple dot product of three vectors. Denote the mode-n unfolding (also called matricization) of a tensor $\mathbf{X}$ as $\mathbf{X}_{(n)}$, see Figure 1 for an example. Let $||\cdot||_1, ||\cdot||_2, ||\cdot||_F, ||\cdot||_*$ be the L_1 norm, the L_2 norm, the Frobenius norm and nuclear norm of matrices or vectors. Let $rank(\cdot)$ be the rank of a matrix, $\otimes$ be the Kronecker product.

3.2 General Form

In this section, we summarize the mainstream TDB models and derive a general form for them to facilitate the following theoretical analysis. And we analyze the number of parameters and computational complexity of the general form. For a vector $\boldsymbol{a} \in \mathbb{R}^D$, we split $\boldsymbol{a}$ into P parts each with d dimension and denote $\boldsymbol{a}$ as $[(\boldsymbol{a})_1, (\boldsymbol{a})_2, \ldots, (\boldsymbol{a})_P]$, where $(\boldsymbol{a})_i$ is the i-th part of vector $\boldsymbol{a}$.

CP/DistMult Let $P = 1$, CP (Lacroix et al., 2018) can be represented as

$$\mathbf{X}_{i,j,k} =< \boldsymbol{H}_i, \boldsymbol{R}_j, \boldsymbol{T}_k >$$

DistMult (Yang et al., 2014), a particular case of CP, which shares the embedding matrix of head entities and tail entities, $i.e.$, $\boldsymbol{H} = \boldsymbol{T}$.

ComplEx/HOLE Let $P = 2$, ComplEx (Trouillon et al., 2017) can be represented as

$$\begin{aligned}\mathbf{X}_{i,j,k} =& < (\boldsymbol{H}_i)_1, (\boldsymbol{R}_j)_1, (\boldsymbol{T}_k)_1 > + < (\boldsymbol{H}_i)_2, (\boldsymbol{R}_j)_1, (\boldsymbol{T}_k)_2 > \\ &+ < (\boldsymbol{H}_i)_1, (\boldsymbol{R}_j)_2, (\boldsymbol{T}_k)_2 > - < (\boldsymbol{H}_i)_2, (\boldsymbol{R}_j)_2, (\boldsymbol{T}_k)_1 >\end{aligned}$$

Liu et al. (2017) proved HOLE (Nickel et al., 2011) is equivalent to ComplEx.

SimplE Let $P = 2$, SimplE (Kazemi & Poole, 2018) can be represented as

$$\mathbf{X}_{i,j,k} =< (\boldsymbol{H}_i)_1, (\boldsymbol{R}_j)_1, (\boldsymbol{T}_k)_2 > + < (\boldsymbol{H}_i)_2, (\boldsymbol{R}_j)_2, (\boldsymbol{T}_k)_1 >$$

ANALOGY Let $P = 4$, ANALOGY (Liu et al., 2017) can be represented as

$$\begin{aligned}\mathbf{X}_{i,j,k} =& < (\boldsymbol{H}_i)_1, (\boldsymbol{R}_j)_1, (\boldsymbol{T}_k)_1 > + < (\boldsymbol{H}_i)_2, (\boldsymbol{R}_j)_2, (\boldsymbol{T}_k)_2 > \\ &+ < (\boldsymbol{H}_i)_3, (\boldsymbol{R}_j)_3, (\boldsymbol{T}_k)_3 > + < (\boldsymbol{H}_i)_3, (\boldsymbol{R}_j)_4, (\boldsymbol{T}_k)_4 > \\ &+ < (\boldsymbol{H}_i)_4, (\boldsymbol{R}_j)_3, (\boldsymbol{T}_k)_4 > - < (\boldsymbol{H}_i)_4, (\boldsymbol{R}_j)_4, (\boldsymbol{T}_k)_3 >\end{aligned}$$

QuatE Let $P = 4$, QuatE (Zhang et al., 2019) can be represented as

$$\begin{aligned}\mathbf{X}_{i,j,k} =& < (\boldsymbol{H}_i)_1, (\boldsymbol{R}_j)_1, (T_k)_1 > - < (\boldsymbol{H}_i)_2, (\boldsymbol{R}_j)_2, (\boldsymbol{T}_k)_1 > \\ &- < (\boldsymbol{H}_i)_3, (\boldsymbol{R}_j)_3, (\boldsymbol{T}_k)_1 > - < (\boldsymbol{H}_i)_4, (\boldsymbol{R}_j)_4, (\boldsymbol{T}_k)_1 > \\ &+ < (\boldsymbol{H}_i)_1, (\boldsymbol{R}_j)_2, (\boldsymbol{T}_k)_2 > + < (\boldsymbol{H}_i)_2, (\boldsymbol{R}_j)_1, (\boldsymbol{T}_k)_2 > \\ &+ < (\boldsymbol{H}_i)_3, (\boldsymbol{R}_j)_4, (\boldsymbol{T}_k)_2 > - < (\boldsymbol{H}_i)_4, (\boldsymbol{R}_j)_3, (\boldsymbol{T}_k)_2 > \\ &+ < (\boldsymbol{H}_i)_1, (\boldsymbol{R}_j)_3, (\boldsymbol{T}_k)_3 > + < (\boldsymbol{H}_i)_3, (\boldsymbol{R}_j)_1, (\boldsymbol{T}_k)_3 > \\ &+ < (\boldsymbol{H}_i)_4, (\boldsymbol{R}_j)_2, (\boldsymbol{T}_k)_3 > - < (\boldsymbol{H}_i)_2, (\boldsymbol{R}_j)_4, (\boldsymbol{T}_k)_3 > \\ &+ < (\boldsymbol{H}_i)_1, (\boldsymbol{R}_j)_4, (\boldsymbol{T}_k)_4 > + < (\boldsymbol{H}_i)_4, (\boldsymbol{R}_j)_1, (\boldsymbol{T}_k)_4 > \\ &+ < (\boldsymbol{H}_i)_2, (\boldsymbol{R}_j)_3, (\boldsymbol{T}_k)_4 > - < (\boldsymbol{H}_i)_3, (\boldsymbol{R}_j)_2, (\boldsymbol{T}_k)_4 >\end{aligned}$$

TuckER Let $P = D$, TuckER (Balažević et al., 2019a) can be represented as

$$\mathbf{X}_{i,j,k} = \sum_{l=1}^{D}\sum_{m=1}^{D}\sum_{n=1}^{D}\mathbf{W}_{lmn}(\boldsymbol{H}_i)_l(\boldsymbol{R}_j)_m(\boldsymbol{T}_k)_n$$

where $\mathbf{W} \in \mathbb{R}^{D\times D\times D}$ is the weight tensor.

General Form We notice that all models are a linear combination of triple dot product. The only difference between these models are the choice of the part P and the weight tensor $\mathbf{W}$. And the weight tensor of TuckER is parameter tensor, while the weight tensor of other models are predetermined constant tensor. Therefore, we can write a general form of these models as

$$\mathbf{X}_{i,j,k} = \sum_{l=1}^{P}\sum_{m=1}^{P}\sum_{n=1}^{P}\mathbf{W}_{lmn} < (\boldsymbol{H}_i)_l, (\boldsymbol{R}_j)_m, (\boldsymbol{T}_k)_n > \tag{1}$$

where $\mathbf{W} \in \mathbb{R}^{P\times P\times P}$ is the weight tensor, which can be parameter tensor or predetermined constant tensor. This general form can also be seen as a special form of Tucker decomposition. For any weight tensor $\mathbf{W} \in \mathbb{R}^{P\times P\times P}$, we create a tensor $\mathbf{W}^* \in \mathbb{R}^{D\times D\times D}$ such that $\mathbf{W}^*_{dl+s,dm+s,dn+s} = \mathbf{W}_{lmn}(s = 1, 2, \ldots, d)$ and $\mathbf{W}^*_{i,j,k} = 0$ otherwise. Then the general form can also be represented as :

$$\mathbf{X} = \mathbf{W}^* \times_1 \boldsymbol{H} \times_2 \boldsymbol{R} \times_3 \boldsymbol{T} \tag{2}$$

where $\times_n$ is the tensor product along the n-th mode (Kolda & Bader, 2009).

The Number of Parameters and Computational Complexity The parameters of the model come from two part, the weight tensor $\mathbf{W}$ and the embedding matrix $\boldsymbol{H}, \boldsymbol{R}, \boldsymbol{T}$. The number of parameters of the weight tensor $\mathbf{W}$ is equal to P^3 if $\mathbf{W}$ is parameter tensor otherwise equal to 0. The number of parameters of the embedding matrix is equal to $|\mathcal{E}|D + |\mathcal{R}|D$ if $\boldsymbol{H} = \boldsymbol{T}$ otherwise equal to $2|\mathcal{E}|D + |\mathcal{R}|D$. The computational complexity is equal to $\mathcal{O}(DP^2|\mathcal{E}|^2|\mathcal{R}|)$. The larger the part P, the more expressive the model and the more the computation. Therefore, the choice of the part P is a trade-off between expressiveness and computation.

3.3 Logical Rules

In this section, we analyze how to design models such that the models can learn the symmetric rules, anti-symmetric rules and inverse rules. In the previous section, we derive a general form Eq.(1) of the TDB models. Thus, we study how to enable the general form model to learn logical rules. We first define the logical rules. We denote $\mathbf{X}_{i,j,k} = f(\boldsymbol{H}_i, \boldsymbol{R}_j, \boldsymbol{T}_k)$ as $f(\boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t})$ for simplicity.

A relation r is symmetry if $\forall h, t, r(h,t) \rightarrow r(t,h)$. A model is able to learn the symmetric rules if

$$\exists \boldsymbol{r} \in \mathbb{R}^D, \forall \boldsymbol{h}, \boldsymbol{t} \in \mathbb{R}^d, f(\boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t}) = f(\boldsymbol{t}, \boldsymbol{r}, \boldsymbol{h}) \wedge f \neq 0$$

A relation r is anti-symmetry if $\forall h, t, r(h,t) \rightarrow \neg r(t,h)$. A model is able to learn the anti-symmetric rules if

$$\exists \boldsymbol{r} \in \mathbb{R}^D, \forall \boldsymbol{h}, \boldsymbol{t} \in \mathbb{R}^d, f(\boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t}) = -f(\boldsymbol{t}, \boldsymbol{r}, \boldsymbol{h}) \wedge f \neq 0$$

A relation r_1 is inverse to a relation r_2 if $\forall h, t, r_1(h,t) \rightarrow r_2(t,h)$. A model is able to learn the inverse rules if

$$\forall \boldsymbol{r}_1 \in \mathbb{R}^D, \exists \boldsymbol{r}_2 \in \mathbb{R}^d, \forall \boldsymbol{h}, \boldsymbol{t} \in \mathbb{R}^d, f(\boldsymbol{h}, \boldsymbol{r}_1, \boldsymbol{t}) = f(\boldsymbol{t}, \boldsymbol{r}_2, \boldsymbol{h}) \wedge f \neq 0$$

Since the difference between TDB models is the part P and the weight tensor $\mathbf{W}$, the properties of TDB models are determined by P and $\mathbf{W}$. Therefore, we have the following proposition about logical rules and P and $\mathbf{W}$.

Proposition 1 *Assume a model can be represented as the form of Eq.(1), then a model is able to learn the symmetric rules iff* $rank(\mathbf{W}_{(2)}^T - \boldsymbol{S}\mathbf{W}_{(2)}^T) < P$. *A model is able to learn the anti-symmetric rule iff* $rank(\mathbf{W}_{(2)}^T + \boldsymbol{S}\mathbf{W}_{(2)}^T) < P$. *A model is able to learn the inverse rules iff* $rank(\mathbf{W}_{(2)}^T) = rank([\mathbf{W}_{(2)}^T, \boldsymbol{S}\mathbf{W}_{(2)}^T])$, *where* $\boldsymbol{S} \in \mathbb{R}^{P^2 \times P^2}$ *is a permutation matrix with* $\boldsymbol{S}_{(i-1)P+j,(j-1)P+i} = 1(i, j = 1, 2, \ldots, P)$ *else 0,* $[\mathbf{W}_{(2)}^T, \boldsymbol{S}\mathbf{W}_{(2)}^T]$ *is the concatenation of matrix* $\mathbf{W}_{(2)}^T$ *and matrix* $\boldsymbol{S}\mathbf{W}_{(2)}^T$.

See proof in Appendix. By Proposition 1, we only need to judge the relationship between the part P and the matrix rank about $\mathbf{W}_{(2)}$. ComplEx, SimplE, ANALOGYY and QuatE design a specific weight tensor to make the models enable to learn logical rules. We can verify that theses models satisfy the conditions in Proposition 1.

3.4 Tensor N-rank Regularization

ComplEx (Trouillon et al., 2017), SimplE (Kazemi & Poole, 2018) and TuckER (Balažević et al., 2019a) have proved their models are fully expressive, which can represent any ground truth tensor. Therefore, TDB models are fully expressive when $P \geq 2$. However, TDB models suffer from the overfitting problems in practice. In this section, we propose a regularization, tensor n-rank regularization (TNRR), to alleviate the overfitting problem. Motivated by the success of low- rank matrix completion (Nguyen et al., 2019), we aim to achieve low-rank KGC. The definition of tensor rank is not unique, $e.g.$, CP-rank (Gandy et al., 2011) which based on CP decomposition and n-rank (Gandy et al., 2011) which based on Tucker decomposition. Since the general form Eq.(1) can be seen as a special form of Tucker decomposition Eq.(2), we propose our regularization based on the n-rank of the tensor. The n-rank of a Nth order tensor $\mathbf{X} \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_N}$ is defined as the vector of the ranks of the mode-n unfoldings:

$$n - rank(\mathbf{X}) = (rank(\mathbf{X}_{(1)}), rank(\mathbf{X}_{(2)}), \ldots, rank(\mathbf{X}_{(N)}))$$

In order to achieve low-rank KGC, we can minimize $rank(\mathbf{X}_{(1)}) + rank(\mathbf{X}_{(2)}) + rank(\mathbf{X}_{(3)})$. Since the matrix rank minimization problem is NP-hard for most problems (Recht et al., 2008), the matrix nuclear norm is widely used as a convex surrogate of matrix rank. However, the matrix nuclear norm is a complicated non-differentiable function (Rennie & Srebro, 2005) and computing the nuclear norm is expensive for large KG, we do not use the matrix nuclear norm directly. Note that the matrix nuclear norm and squared Frobenius norm have the following relationship:

Lemma 1 (Srebro & Shraibman (2005)) *For any matrix* $\boldsymbol{Z}$*:*

$$||\boldsymbol{Z}||_* = \min_{\boldsymbol{Z}=\boldsymbol{U}\boldsymbol{V}^T} ||\boldsymbol{U}||_F^2 ||\boldsymbol{V}||_F^2 = \min_{\boldsymbol{z}=\boldsymbol{U}\boldsymbol{V}^T} \frac{1}{2}(||\boldsymbol{U}||_F^2 + ||\boldsymbol{V}||_F^2)$$

Since the general form of TDB models can be represented as:

$$\mathbf{X} = \mathbf{W}^* \times_1 \boldsymbol{H} \times_2 \boldsymbol{R} \times_3 \boldsymbol{T}$$

then we have (Kolda & Bader, 2009)

$$\begin{aligned}\mathbf{X}_{(1)} &= \boldsymbol{H}(\mathbf{W}^*_{(1)}(\boldsymbol{T}\otimes\boldsymbol{R})^T) = (\boldsymbol{H}\mathbf{W}^*_{(1)})(\boldsymbol{T}\otimes\boldsymbol{R})^T\\ \mathbf{X}_{(2)} &= \boldsymbol{R}(\mathbf{W}^*_{(2)}(\boldsymbol{T}\otimes\boldsymbol{H})^T) = (\boldsymbol{R}\mathbf{W}^*_{(2)})(\boldsymbol{T}\otimes\boldsymbol{H})^T\\ \mathbf{X}_{(3)} &= \boldsymbol{T}(\mathbf{W}^*_{(3)}(\boldsymbol{R}\otimes\boldsymbol{H})^T) = (\boldsymbol{T}\mathbf{W}^*_{(3)})(\boldsymbol{R}\otimes\boldsymbol{H})^T\end{aligned}$$

By applying Lemma 1 to $\mathbf{X}_{(1)}$, $\mathbf{X}_{(2)}$ and $\mathbf{X}_{(3)}$, we have the following proposition:

Proposition 2 *Let* $L(X) = ||\mathbf{X}_{(1)}||_* + ||\mathbf{X}_{(2)}||_* + ||\mathbf{X}_{(3)}||_*$. *We have that*

$$2L(X) \le ||\boldsymbol{H}||_F^2+||\mathbf{W}^*_{(1)}(\boldsymbol{T}\otimes\boldsymbol{R})^T||_F^2+||\boldsymbol{R}||_F^2+||\mathbf{W}^*_{(2)}(\boldsymbol{T}\otimes\boldsymbol{H})^T||_F^2+||\boldsymbol{T}||_F^2+||\mathbf{W}^*_{(3)}(\boldsymbol{R}\otimes\boldsymbol{H})^T||_F^2$$

$$2L(X) \le ||\boldsymbol{H}\mathbf{W}^*_{(1)}||_F^2 + ||\boldsymbol{T}\otimes\boldsymbol{R}||_F^2 + ||\boldsymbol{R}\mathbf{W}^*_{(2)}||_F^2 + ||\boldsymbol{T}\otimes\boldsymbol{H}||_F^2 + ||\boldsymbol{T}\mathbf{W}^*_{(3)}||_F^2 + ||\boldsymbol{R}\otimes\boldsymbol{H}||_F^2$$

By adding the two inequalities in Proposition 2, we get

$$\begin{aligned}4L(\mathbf{X}) \le & ||\boldsymbol{H}||_F^2 + ||\boldsymbol{R}||_F^2 + ||\boldsymbol{T}||_F^2\\ &+||\boldsymbol{T}\otimes\boldsymbol{R}||_F^2 + ||\boldsymbol{T}\otimes\boldsymbol{H}||_F^2 + ||\boldsymbol{R}\otimes\boldsymbol{H}||_F^2\\ &+||\boldsymbol{H}\mathbf{W}^*_{(1)}||_F^2 + ||\boldsymbol{R}\mathbf{W}^*_{(2)}||_F^2 + ||\boldsymbol{T}\mathbf{W}^*_{(3)}||_F^2\\ &+||\mathbf{W}^*_{(1)}(\boldsymbol{T}\otimes\boldsymbol{R})^T||_F^2 + ||\mathbf{W}^*_{(2)}(\boldsymbol{T}\otimes\boldsymbol{H})^T||_F^2 + ||\mathbf{W}^*_{(3)}(\boldsymbol{R}\otimes\boldsymbol{H})^T||_F^2\end{aligned} \tag{3}$$

The $r.h.s.$ of Eq.(3) is an upper bound of the sum of the nuclear norms. We utilize this upper bound as our regularization term. We further use the weighted versions of the Eq.(3), in which the regularization term corresponding to the sampled valid triplets, as in (Lacroix et al., 2018; Zhang et al., 2020). The weighted version of regularization usually outperforms the unweighted regularization when entries of the matrix or tensor are sampled non-uniformly. And we use different regularization coefficients for different parts of Eq.(3). This leads to our regularization term (TNRR) as follows:

$$\begin{aligned}reg(\mathbf{X}) =& \lambda_1(||\boldsymbol{H}_i||_2^2 + ||\boldsymbol{R}_j||_2^2 + ||\boldsymbol{T}_k||_2^2)\\ &+\lambda_2(||\boldsymbol{T}_k||_2^2||\boldsymbol{R}_j||_2^2 + ||\boldsymbol{T}_k||_2^2||\boldsymbol{H}_i||_2^2 + ||\boldsymbol{R}_j||_2^2||\boldsymbol{H}_i||_2^2)\\ &+\lambda_3(||\boldsymbol{H}_i\mathbf{W}^*_{(1)}||_2^2 + ||\boldsymbol{R}_j\mathbf{W}^*_{(2)}||_2^2 + ||\boldsymbol{T}_k\mathbf{W}^*_{(3)}||_2^2)\\ &+\lambda_4(||\mathbf{W}^*_{(1)}(\boldsymbol{T}_k\otimes\boldsymbol{R}_j)^T||_2^2 + ||\mathbf{W}^*_{(2)}(\boldsymbol{T}_k\otimes\boldsymbol{H}_i)^T||_2^2 + ||\mathbf{W}^*_{(3)}(\boldsymbol{R}_j\otimes\boldsymbol{H}_i)^T||_2^2)\end{aligned} \tag{4}$$

where $\lambda_i (i = 1, 2, 3, 4)$ are the regularization coefficients. The intuition behind our regularization is that it can control the norm of $\boldsymbol{H}, \boldsymbol{R}, \boldsymbol{T}$ and the norm of the intermediate variables in the process of computing the tensor $\mathbf{X}$. Since the tensor $\mathbf{X}$ in Eq.(2) can be computed in different orders, the intermediate variables will be different. Therefore, the regularization can control the norm of all intermediate variables in the different processes of computing $\mathbf{X}$. The computational complexity of TNRR is equal to $\mathcal{O}(DP^2)$.

We use the same loss function, multiclass log-loss function, as in (Lacroix et al., 2018; Zhang et al., 2020). For a training triplet (i, j, k), our loss function is

$$\begin{aligned}\ell_{i,j,k}(\mathbf{X}) = &- \mathbf{X}_{i,j,k} + log(\sum_{k'=1}^{|\mathcal{E}|} exp(\mathbf{X}_{i,j,k'}))\\ &- \mathbf{X}_{i,j+|\mathcal{R}|,k} + log(\sum_{i'=1}^{|\mathcal{E}|} exp(\mathbf{X}_{k,j+|\mathcal{R}|,i'}))\\ &+ reg(\mathbf{X})\end{aligned} \tag{5}$$

At test time, we use $\mathbf{X}_{i,j,:}$ to rank possible right hand sides for query $(i, j, ?)$ and $\mathbf{X}_{:,j,k}$ to rank possible left hand sides for query $(?, j, k)$.

Remark Zhang et al. (2020) prove DURA is an upper bound of tensor nuclear 2-norm based on CP decomposition. Since the CP decomposition is a special case of Tucker decomposition, our regularization based on Tucker decomposition is more general. We can write DURA as follows:

$$\begin{aligned}DURA(\mathbf{X}) =& ||\boldsymbol{H}_i||_2^2 + ||\boldsymbol{T}_k||_2^2\\ &+||\mathbf{W}^*_{(1)}(\boldsymbol{T}_k\otimes\boldsymbol{R}_j)^T||_2^2 + ||\mathbf{W}^*_{(3)}(\boldsymbol{R}_j\otimes\boldsymbol{H}_i)^T||_2^2\end{aligned}$$

which is a part of our regularization term.

Table 1: Knowledger graph completion results on WN18 and FB15k datasets.

	WN18			**FB15k**		
	MRR	H@1	H@10	MRR	H@1	H@10
TransE	0.495	0.113	0.943	0.463	0.297	0.749
RotatE	0.949	0.944	0.959	0.797	0.746	0.0.884
ConvE	0.943	0.935	0.956	0.657	0.558	0.831
HypER	0.951	0.947	0.958	0.790	0.734	0.885
DisMult	0.822	0.728	0.936	0.654	0.546	0.824
ComplEx	0.941	0.936	0.947	0.692	0.599	0.840
HOLEX	0.938	0.930	0.949	0.800	0.750	0.886
SimplE	0.942	0.939	0.947	0.727	0.660	0.838
ANALOGY	0.942	0.939	0.947	0.725	0.646	0.854
QuatE	0.950	0.944	**0.962**	**0.833**	**0.800**	0.900
TuckER	**0.953**	**0.949**	0.958	0.795	0.741	0.892
TNRR($\lambda_i = 0$)	0.948	0.945	0.954	0.815	0.767	0.895
TNRR	**0.953**	0.948	**0.962**	0.828	0.784	**0.902**

Table 2: Knowledger graph completion results on WN18RR and FB15k-237 datasets.

	WN18RR			**FB15k-237**		
	MRR	H@1	H@10	MRR	H@1	H@10
TransE	0.226	—	0.501	0.294	—	0.465
RotatE	0.476	0.428	0.571	0.338	0.241	0.533
ConvE	0.43	0.40	0.52	0.325	0.237	0.501
HypER	0.465	0.436	0.522	0.341	0.252	0.520
CP	0.438	0.414	0.485	0.333	0.247	0.508
DistMult	0.430	0.390	0.490	0.241	0.155	0.419
ComplEx	0.440	0.410	0.510	0.247	0.158	0.428
QuatE	0.482	0.436	0.572	0.366	0.271	**0.556**
TuckER	0.470	0.443	0.526	0.358	0.266	0.358
TNRR($\lambda_i = 0$)	0.448	0.422	0.496	0.308	0.220	0.485
TNRR	**0.501**	**0.460**	**0.579**	**0.368**	**0.273**	0.555

4 EXPERIMENTS

In this section, we introduce the experimental settings in Section 4.1 and show the results in Section 4.2. We study the impact of the part P in Section 4.3. Finally, we analyze the impact of the regularization and compare our regularization to other regularization in Section 4.4.

4.1 EXPERIMENTAL SETTINGS

Datasets We evaluate our model on four popular benchmark datasets, WN18, WN18RR, FB15k and FB15k-237. WN18 (Bordes et al., 2013) is extracted from WordNet, a database containing lexical relations between words. WN18RR (Dettmers et al., 2018) is a subset of WN18, with inverse relations removed. FB15k (Bordes et al., 2013) is extracted from Freebase, a large database of real world facts. FB15k-237 (Toutanova et al., 2015) is a subset of FB15k, with inverse relations removed.

Table 3: The results on WN18RR dataset with different part P.

Part P	1	2	4	10	20	50	100	200
MRR	0.450	0.447	0.461	0.460	0.459	0.465	0.493	0.501
H@1	0.411	0.407	0.425	0.422	0.422	0.427	0.452	0.460
H@10	0.526	0.524	0.538	0.533	0.534	0.541	0.573	0.579
$rank(\mathbf{W}_{(2)}^T - \boldsymbol{S}\mathbf{W}_{(2)}^T)$	0	1	2	7	15	44	50	200
$rank(\mathbf{W}_{(2)}^T + \boldsymbol{S}\mathbf{W}_{(2)}^T)$	1	1	4	7	17	45	50	200
$rank(\mathbf{W}_{(2)}^T)$	1	1	3	7	15	40	47	200
$rank([\mathbf{W}_{(2)}^T, \boldsymbol{S}\mathbf{W}_{(2)}^T])$	1	1	3	8	18	45	50	200

Table 4: The results on FB15k-237 dataset with different part P.

Part	1	2	4	10	20	50	100	200
MRR	0.346	0.340	0.345	0.345	0.347	0.345	0.359	0.368
H@1	0.254	0.250	0.253	0.253	0.255	0.252	0.264	0.273
H@10	0.529	0.522	0.529	0.530	0.533	0.532	0.549	0.555
$rank(\mathbf{W}_{(2)}^T - \boldsymbol{S}\mathbf{W}_{(2)}^T)$	0	1	4	10	20	50	100	200
$rank(\mathbf{W}_{(2)}^T + \boldsymbol{S}\mathbf{W}_{(2)}^T)$	1	2	4	10	20	50	100	200
$rank(\mathbf{W}_{(2)}^T)$	1	2	4	10	20	50	100	200
$rank([\mathbf{W}_{(2)}^T, \boldsymbol{S}\mathbf{W}_{(2)}^T])$	1	2	4	10	20	50	100	200

Hyper-parameters We use the filtered MRR and Hits@N (H@N) (Bordes et al., 2013) as evaluation metrics and choose the hyper-parameters with the best filtered MRR on the validation set. We used Adagrad (Duchi et al., 2011) as the optimizer. We set the batch size to 512 and learning rate to 0.1, total embedding dimension D to 200, part P to 200 and embedding dimension d to 1 for all models. We search the regularization coefficients $\lambda_i (i = 1, 2, 3, 4)$ in $\{0, 10^{-4}, 3 \times 10^{-4}, 10^{-3}, 3 \times 10^{-3}, 10^{-2}, 3 \times 10^{-2}, 10^{-1}\}$. We tune hyper-parameters with the help of Hyperopt, a hyper-parameter optimization framework based on TPE (Bergstra et al., 2011). In the hyper-parameters search process, we train the model for 20 epochs. And we train the model for 500 epochs with the best hyper-parameters. Please refer to the supplementary material for more experimental details.

4.2 Results

In this section, we compare the performance of TNRR with several translational models, including TransE (Bordes et al., 2013; Nickel et al., 2016), RotatE (Sun et al., 2018), neural networks models, including ConvE (Dettmers et al., 2018), HypER (Balažević et al., 2019b) and TDB models, including CP (Zhang et al., 2020), DisMult (Yang et al., 2014), ComplEx (Toutanova et al., 2015), HOLEX (Xue et al., 2018), SimplE (Kazemi & Poole, 2018), ANALOGY (Liu et al., 2017), QuatE (Zhang et al., 2019) TuckER (Balažević et al., 2019a). We also train our model without the tensor n-rank regularization, $i.e.$, $\lambda_i = 0 (i = 1, 2, 3, 4)$. See Table 1 and Table 2 for the results. Our model achieves state-of-the-art performance on benchmark datasets.

4.3 The Part

In Section 3.2, we show that the choice of the part P will affect the expressiveness and computation, we study the impact of the part P on the model performance in this section. In Section 3.3, we show that proper part P and weight tensor $\mathbf{W}$ can make the model enable to learn logical rules. we analyze whether the weight tensor learned by the models can automatically satisfy the conditions in Proposition 1. We evaluate the model on WN18RR and FB15k-237 datasets. We set the total

Table 5: The results on WN18RR and FB15k-237 datasets with different regularization coefficient and regularization term.

	WN18RR			**FB15k-237**		
	MRR	H@1	H@10	MRR	H@1	H@10
$\lambda_i = 0$	0.448	0.422	0.496	0.308	0.220	0.485
$\lambda_1 \neq 0$	0.453	0.424	0.508	0.325	0.237	0.502
$\lambda_2 \neq 0$	0.455	0.426	0.509	0.321	0.235	0.493
$\lambda_3 \neq 0$	0.449	0.423	0.496	0.319	0.228	0.505
$\lambda_4 \neq 0$	0.465	0.429	0.534	0.337	0.247	0.518
F2	0.453	0.424	0.508	0.325	0.237	0.502
N3	0.457	0.426	0.515	0.323	0.236	0.497
DURA	0.484	0.448	0.555	0.363	0.269	0.550
TNRR	0.501	0.460	0.579	0.368	0.273	0.555

embedding dimension to 200, and set the part P to 1, 2, 4, 10, 20, 50, 100, 200. See Table 3 and Table 4 for the results. The matrix rank is computed by counting the number of singular values which are greater than 10^{-3}. The results show that the performance generally improves as P increases. And the learned model can approximately satisfy the conditions in Proposition 1. And we find that the models learned on WN18RR dataset basically satisfy the conditions in the Proposition 1. While the models learned on FB15k-237 dataset only satisfy the inverse rule in general. The conditions in Proposition 1 may be too strong in practice. The models may not need to strictly satisfy the logical rules.

4.4 Regularization

In this section, we study which part of our regularization term Eq.(4) is more important and compare our regularization with squared Frobenius norm (F2) regularization and N3 regularization (Lacroix et al., 2018) and DURA Zhang et al. (2020) regularization. We use the model with regularization coefficient $\lambda_i = 0 (i = 1, 2, 3, 4)$ as the baseline. And we compare the baseline with the models with one of the regularization coefficient not equal to 0. Meanwhile, we compare our regularization with squared Frobenius norm (F2) regularization and N3 regularization (Lacroix et al., 2018) and DURA Zhang et al. (2020) regularization. We evaluate the models on WN18RR and FB15k-237 datasets. All models have the same hyper-parameters setting except the regularization coefficients. See Table 5 for the result. The first and the second part of the regularization term Eq.(4) contribute the model nearly because both of them control the scale of the embedding vectors. The third part has the least contribution and the fourth part has the most contribution. The reason may be that the elements in the fourth part are closer to the output of the model. The F2 and N3 regularization improve the models almost equally. DURA is better than N2 and F3, which achieves significant improvement. TNRR is the regularization that improves the model most. This result is not surprising because F2 and DURA are parts of TNRR.

5 Conclusion

In this paper, we analyze the TDB models in KGC. We first summarize the TDB models and derive a general form for them. Based on the general form, we show the principles of model design to satisfy logical rules. TDB models often suffer from the overfitting problem, we propose a regularization term based on the n-rank of the tensor to alleviate the problem by enforcing the low-rankness of the unfolding matrices. We first relax the tensor n-rank to the sum of the nuclear norm and propose a computationally efficient upper bound of the sum of the nuclear norm as our regularization term. Experiments show that our regularization achieves significant improvement on benchmark datasets.

REFERENCES

Ivana Balažević, Carl Allen, and Timothy Hospedales. Tucker: Tensor factorization for knowledge graph completion. 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)*, pp. 5185–5194, 2019a.

Ivana Balažević, Carl Allen, and Timothy M Hospedales. Hypernetwork knowledge graph embeddings. In *International Conference on Artificial Neural Networks*, pp. 553–565. Springer, 2019b.

James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. *Advances in neural information processing systems*, 24, 2011.

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

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. Convolutional 2d knowledge graph embeddings. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

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

Silvia Gandy, Benjamin Recht, and Isao Yamada. Tensor completion and low-n-rank tensor recovery via convex optimization. *Inverse problems*, 27(2):025010, 2011.

Frank L Hitchcock. The expression of a tensor or a polyadic as a sum of products. *Journal of Mathematics and Physics*, 6(1-4):164–189, 1927.

Seyed Mehran Kazemi and David Poole. Simple embedding for link prediction in knowledge graphs. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pp. 4289–4300, 2018.

Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3): 455–500, 2009.

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

Hanxiao Liu, Yuexin Wu, and Yiming Yang. Analogical inference for multi-relational embeddings. In *International conference on machine learning*, pp. 2168–2178. PMLR, 2017.

Luong Trung Nguyen, Junhan Kim, and Byonghyo Shim. Low-rank matrix completion: A contemporary survey. *IEEE Access*, 7:94215–94237, 2019.

Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, pp. 809–816, 2011.

Maximilian Nickel, Lorenzo Rosasco, and Tomaso Poggio. Holographic embeddings of knowledge graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

Benjamin Recht, Weiyu Xu, and Babak Hassibi. Necessary and sufficient conditions for success of the nuclear norm heuristic for rank minimization. In *2008 47th IEEE Conference on Decision and Control*, pp. 3065–3070. IEEE, 2008.

Jasson DM Rennie and Nathan Srebro. Fast maximum margin matrix factorization for collaborative prediction. In *Proceedings of the 22nd international conference on Machine learning*, pp. 713–719, 2005.

Nathan Srebro and Adi Shraibman. Rank, trace-norm and max-norm. In *International Conference on Computational Learning Theory*, pp. 545–560. Springer, 2005.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. Rotate: Knowledge graph embedding by relational rotation in complex space. In *International Conference on Learning Representations*, 2018.

Kristina Toutanova, Danqi Chen, Patrick Pantel, Hoifung Poon, Pallavi Choudhury, and Michael Gamon. Representing text for joint embedding of text and knowledge bases. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pp. 1499–1509, 2015.

Théo Trouillon, Christopher R Dance, Éric Gaussier, Johannes Welbl, Sebastian Riedel, and Guillaume Bouchard. Knowledge graph completion via complex tensor factorization. *Journal of Machine Learning Research*, 18:1–38, 2017.

Ledyard R Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3): 279–311, 1966.

Yexiang Xue, Yang Yuan, Zhitian Xu, and Ashish Sabharwal. Expanding holographic embeddings for knowledge completion. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pp. 4496–4506, 2018.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. *arXiv e-prints*, pp. arXiv–1412, 2014.

Jing Zhang, Bo Chen, Lingxi Zhang, Xirui Ke, and Haipeng Ding. Neural, symbolic and neural-symbolic reasoning on knowledge graphs. *AI Open*, 2:14–35, 2021.

Shuai Zhang, Yi Tay, Lina Yao, and Qi Liu. Quaternion knowledge graph embeddings. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pp. 2735–2745, 2019.

Zhanqiu Zhang, Jianyu Cai, and Jie Wang. Duality-induced regularizer for tensor factorization based knowledge graph completion. *Advances in Neural Information Processing Systems*, 33, 2020.

A Appendix

A.1 Proofs

Proposition 1 Proof: According to the symmetric rule, we have that

$$
\begin{aligned}
&\sum_{l=1}^{P}\sum_{m=1}^{P}\sum_{n=1}^{P}\mathbf{W}_{lmn}<(\boldsymbol{h})_l,(\boldsymbol{r})_m,(\boldsymbol{t})_n>-\sum_{l=1}^{P}\sum_{m=1}^{P}\sum_{n=1}^{P}\mathbf{W}_{lmn}<(\boldsymbol{t})_l,(\boldsymbol{r})_m,(\boldsymbol{h})_n>\\
=&\sum_{l=1}^{P}\sum_{m=1}^{P}\sum_{n=1}^{P}\mathbf{W}_{lmn}<(\boldsymbol{h})_l,(\boldsymbol{r})_m,(\boldsymbol{t})_n>-\sum_{l=1}^{P}\sum_{m=1}^{P}\sum_{n=1}^{P}\mathbf{W}_{nml}<(\boldsymbol{h})_l,(\boldsymbol{r})_m,(\boldsymbol{t})_n>\\
=&\sum_{l=1}^{P}\sum_{m=1}^{P}\sum_{n=1}^{P}(\mathbf{W}_{lmn}-\mathbf{W}_{nml})<(\boldsymbol{h})_l,(\boldsymbol{r})_m,(\boldsymbol{t})_n>\\
=&\sum_{l=1}^{P}\sum_{m=1}^{P}\sum_{n=1}^{P}(\mathbf{W}_{lmn}-\mathbf{W}_{nml})(\boldsymbol{r})_m^T((\boldsymbol{h})_l\odot(\boldsymbol{t})_n)\\
=&\sum_{l=1}^{P}\sum_{n=1}^{P}(\sum_{m=1}^{P}(\mathbf{W}_{lmn}-\mathbf{W}_{nml})(\boldsymbol{r})_m^T)((\boldsymbol{h})_l\odot(\boldsymbol{t})_n)=0
\end{aligned}
$$

where $\odot$ is the Hadamard product. Since the above equation holds for any $\boldsymbol{h},\boldsymbol{t}\in\mathbb{R}^D$, we can get

$$
\sum_{m=1}^{P}(\mathbf{W}_{lmn}-\mathbf{W}_{nml})(\boldsymbol{r})_m^T)=\mathbf{0}
$$

Table 6: The hyper-parameters of the experiments in Section 4.2.

Part	λ_1	λ_2	λ_3	λ_4
WN18	0.0003	0.03	0	0.003
FB15k	0.003	0.01	0	0.001
WN18RR	0.01	0.03	0	0.03
FB15k-237	0.0003	0.03	0	0.01

Table 7: The results of our models on WN18, FB15k, WN18RR, FB15k-237 datasets.

Part	MR	MRR	H@1	H@3	H@10
WN18	184.21	0.953	0.948	0.956	0.962
FB15k	43.61	0.828	0.784	0.859	0.902
WN18RR	2923.69	0.501	0.460	0.515	0.579
FB15k-237	179.45	0.368	0.273	0.404	0.555

Therefore, the symmetric rule is transformed into a system of linear equations. This system of linear equations should have non-zero solution, otherwise it will result in $f = 0$. Thus, we have $rank(\mathbf{W}_{(2)}^T - \boldsymbol{S}\mathbf{W}_{(2)}^T) < P$.

For the anti-symmetric rule, we can also get $rank(\mathbf{W}_{(2)}^T + \boldsymbol{S}\mathbf{W}_{(2)}^T) < P$.

For the inverse rule, we have the following equation:

$$\mathbf{W}_{(2)}^T \boldsymbol{r}_1 = \boldsymbol{S}\mathbf{W}_{(2)}^T \boldsymbol{r}_2$$

For any $\boldsymbol{r}_1 \in \mathbb{R}^D$, there exists $\boldsymbol{r}_2 \in \mathbb{R}^D$ such that the above equation holds. Therefore, the column vectors of $\mathbf{W}_{(2)}^T$ can be expressed linearly by the column vectors of $\boldsymbol{S}\mathbf{W}_{(2)}^T$. Since $\boldsymbol{S}$ is a permutation matrix, we have that $\boldsymbol{S}^2 = \boldsymbol{I}$, thus

$$\boldsymbol{S}\mathbf{W}_{(2)}^T \boldsymbol{r}_1 = \boldsymbol{S}^2\mathbf{W}_{(2)}^T \boldsymbol{r}_1 = \mathbf{W}_{(2)}^T \boldsymbol{r}_2$$

For any $\boldsymbol{r}_1 \in \mathbb{R}^D$, there exists $\boldsymbol{r}_2 \in \mathbb{R}^D$ such that the above equation holds. Thus, the column vectors of $\boldsymbol{S}\mathbf{W}_{(2)}^T$ can be expressed linearly by the column vectors of $\mathbf{W}_{(2)}^T$. Therefore, the column space of $\mathbf{W}_{(2)}^T$ is equivalent to the column space of $\boldsymbol{S}\mathbf{W}_{(2)}^T$, thus we have

$$rank(\mathbf{W}_{(2)}^T) = rank(\boldsymbol{S}\mathbf{W}_{(2)}^T) = rank([\mathbf{W}_{(2)}^T, \boldsymbol{S}\mathbf{W}_{(2)}^T])$$

A.2 Experimental details

Hyper-parameters We set the batch size to 512 and learning rate to 0.1, total embedding dimension D to 200, part P to 200 and embedding dimension d to 1 for all models. We search the regularization coefficients $\lambda_i (i = 1, 2, 3, 4)$ in $\{0, 10^{-4}, 3 \times 10^{-4}, 10^{-3}, 3 \times 10^{-3}, 10^{-2}, 3 \times 10^{-2}, 10^{-1}\}$. See Table 6 for the best hyper-parameters we searched. And we show more detailed result of our model in Table 7.

Initialization of the model Since the model has the general form Eq.(1), we initialize the parameters of the model from a uniform distribution with 0 mean and $\sqrt[4]{\frac{1}{p^3 d}}$ variance such that the output of the model has 0 mean and 1 variance.