24th European Conference on Artificial Intelligence - ECAI 2020
Santiago de Compostela, Spain

BTDE: Block Term Decomposition Embedding for Link Prediction in Knowledge Graph

Tao Luo[1,2,3] and **Yifan Wei**[1,2,3] and **Mei Yu**[1,2,3] and **Xuewei Li**[1,2,3] and **Mankun Zhao**[1,2,3] and **Tianyi Xu**[1,2,3] and **Jian Yu**[1,2,3] and **Jie Gao**[1,2,3] and **Ruiguo Yu**[1,2,3,✉]

Abstract. Link prediction is the main task of knowledge graph completion, predicting missing relations between entities based the existing links among the entities. The problem of knowledge graph completion can be framed as a third-order binary tensor completion problem. In this case, tensor decomposition seems like a natural solution. And many previous studies have shown that tensor decomposition methods are superior to Trans-based methods in link prediction experiments. Typical tensor decomposition methods are Canonical Polyadic (CP) decomposition and Tucker decomposition. In this paper, we propose Block term decomposition Embedding model (BTDE) for link prediction based on Block term decomposition (which can be seen as a combination of CP decomposition and Tucker decomposition) of the binary tensor representation of knowledge graph triples. The embeddings learned through BTDE is interpretable. In addition, we prove BTDE is fully expressive and derive the bound on its entity and relation embedding dimensionality for full expressivity which is the same as TuckER and smaller than the bound of previous start-of-the-art models ComplEx and SimplE. We show empirically that BTDE outperforms most previous state-of-the-art models across five standard link prediction datasets.

1 INTRODUCTION

Knowledge graph is a semantic network that contains numerous entities and relations in a structured way. It is a graph of nodes and edges in essence, in which nodes are entities and edges are relations. It usually use a triple $(subject, relation, object)$ to describe a fact in the real world. There are many well-established and successfully applied knowledge graphs, such as FreeBase [2], WordNet [20], DBpedia [16], YAGO [26], NELL [4], etc. These knowledge graphs are used in several fields, including search, question answering, natural language processing, recommendation systems, etc. Even a knowledge graph with billions of triples is still incomplete. In the completion of knowledge graph, learners must predict potential triples based on existing ones. This problem has received a huge amount of attention and researches.

Link prediction is the main task of knowledge graph completion, predicting missing links in knowledge graph, including search $(subject, relation, ?)$ or $(?, relation, object)$. Therefore, a large number of link prediction methods have been proposed, such as Trans-based TransE [3] and its extensions, including TransH [30], TransR [17], TransD [11], STransE [21], etc., and tensor decomposition based, such as RESCAL [23], DisMult [33], ComplEx [28], SimplE [12], TuckER [1] and so on. Recently, some scholars have proposed nonlinear convolution models to achieve state-of-the-art results, for example ConvE [6]. Although some deep learning methods have achieved good performance than before, they have the fundamental problem non-transparent and poorly understood.

In this paper, we introduce Block term decomposition embedding (BTDE), a more general tensor decomposition method for knowledge graph completion. Block term decomposition factorizes a tensor into the sum of R parts, and each term is a core tensor multiplied by a matrix along each mode. In our case, the knowledge graph third-order binary tensor will be decomposed the sum of two terms, forward term and reverse term. The forward term is the decomposition of the $(subject, relation, object)$ triple sets, and the reverse term is the decomposition of the corresponding $(object, relation^{-1}, subject)$ triple sets. The rows of three matrices contain subject entity, relation, and object entity embedding vectors, while the core tensor represents the interaction between the three. Further, the subject embedding of an entity is learned independently of its object embedding.

As relations in knowledge graph can be symmetrical, asymmetrical and transitive, link prediction models will have to accurately represent all of these relation types. In this paper, we show that BTDE is fully expressive, and the embedded boundaries of entities and relations can be limited to lower dimensions. We evaluate our model with the task of link prediction on public benchmark datasets: WN18RR, FB15k-237, YAGO3-10, WN18 and FB15k. Experimental results show that our approach outperforms most state-of-the-art models.

In summary, our contributions are as follows:

- We propose a novel model, BTDE, which outperforms most state-of-the-art link prediction models.
- We prove that BTDE is fully expressive and limits the embedding of entity and relation for full expressiveness to a lower dimension than ComplEx and SimplE.

The rest of the paper is organised as follows: Section "RELATED WORK" introduces the existing embedding approaches for link prediction. Section "BACKGROUND" introduces the link prediction and Block term decomposition in math. Section "BLOCK TERM DECOMPOSITION EMBEDDING" describes the proposed BTDE mode in detail. Section "EXPERIMENTS AND RESULTS" compares BTDE with several state-of-the-art embedding models. Section "ANALYSIS" explores the ablation study and the influence of

[1] College of Intelligence and Computing, Tianjin University, China, e-mail: {luotao, weiyifan, yumei, lixuewei, zmk, tianyi.xu, yujian, gaojie, rgyu}@tju.edu.cn; Ruiguo Yu, corresponding author
[2] Tianjin Key Laboratory of Cognitive Computing and Application
[3] Tianjin Key Laboratory of Advanced Networking (TANK Lab)

embedding dimensionality and then a conclusion in Section "CONCLUSION".

2 RELATED WORK

A large number of knowledge graph embedding models for link prediction have been proposed. There are three main types: Translational model, Multiplicative model and Deep learning model.

For ease of reference, the symbols used in the paper are briefly listed here. $\mathcal{E}$ and $\mathcal{R}$ represent assemblies of entities and relations respectively. The knowledge graph triplet set is represented as $\mathcal{G} = \{(s, r, o)\}$, where $s \in \mathcal{E}$ represents the subject entity, $r \in \mathcal{R}$ the relation, and $o \in \mathcal{E}$ the object entity. The $\mathbf{e}_s$ can be regarded as the embedding vector of subject entity, $\mathbf{r}_r$ the embedding vector of relations, $\mathbf{e}_o$ the embedding vector of object entity, and scoring function the $\varphi(s, r, o)$.

2.1 Translational models

In 2013, Bordes et al. proposed TransE, the earliest translating embedding model, serving as the basis of a series of subsequent translating embedding models. Its basic idea is that when the triplet (s, r, o) is true, there will be $\mathbf{e}_s + \mathbf{r}_r \approx \mathbf{e}_o$. TransE is relatively simple, leading to a faster training speed which in turn results in the defects while dealing with the 1-N, N-1, and N-N relations. For example, two triples (s, r, o_1) and (s, r, o_2) have the same subject entity and relation, but different object entities. In this case, the embedding of o_1 and o_1 given by TransE may be very close, but in fact different.

To solve this problem, Wang et al. propose an improved model, TransH, which introduced relation-specific hyperplanes and projected entities s and o onto the hyperplanes for their relation r. It is because of the projection operation that the same entity may have completely different embedding on the hyperplane of different relations. Similar to TransH, TransR introduced spaces for relations, projecting entities s and o into the space for their relation r. Although TransR shows better performance than TransE and TransH, it introduced relational space for each relation, resulting in more parameters and training time. Drawing on TransR, Ji et al. have brought up TransD. In TransD, product of vectors is replacement of projection matrix used in TransR, greatly reducing the number of parameters required by the model. TransH, TransR, and TransD all project entities into the hyperplanes or spaces for specific relations to improve defects of TransE by enabling entities to have different embeddings in different relations.

In addition, some improvements of TransE are through relaxing conditions of $\mathbf{e}_s + \mathbf{r}_r \approx \mathbf{e}_o$. For example, Xiao, Huang, and Zhu have introduced ManifoldE [31], which relaxes $\mathbf{e}_s + \mathbf{r}_r \approx \mathbf{e}_o$ into $\|\mathbf{e}_s + \mathbf{r}_r \approx \mathbf{e}_o\|_2^2 \approx \theta_r^2$. o is no longer a certain point, but located at the hypersphere with $s + r$ as the center and a radius of θ_r. The same relaxation is also completed in TransF [8].

2.2 Multiplicative models

Multiplicative knowledge graph embedding models generally use a variety of multiplications to capture the latent interactions between entities.

2.2.1 RESCAL

In 2011, Nickel, Tresp, and Kriegel proposed RESCAL, in which each entity was represented as a vector, tensor product is used to capture similarities between entities, and each relation was represented as a matrix to capture interactions of entities. Since matrix in RESCAL represents relation, the number of parameters in it is tremendous.

2.2.2 DistMult

In 2015, based on the idea of the RESCAL, Yang et al. proposed DistMult, which simplified RESCAL by replacing the matrix representing relation with the diagonal matrix. This simplification has greatly reduced the number of parameters in DistMult. Although DistMult does not distinguish between subject and object entity, this model can only be used to deal with symmetrical relations. There is no doubt that there are certain limitations.

2.2.3 HolE

Nickel et al. propose Hole [22], which used circular correlations to capture the latent interactions between entities. Because of the asymmetry of circular correlation, HoLE can be used to handle asymmetrical relations.

2.2.4 ComplEx

In 2016, Trouillon et al. proposed ComplEx, which extended the embeddings of entities and relations from a real space to a complex space. Due to asymmetries of complex multiplication, ComplEx can achieve great performance in dealing with asymmetrical relations.

2.2.5 SimplE

Inspired by the idea of CP decomposition, so as to address independency of the subject and object vector of the same entity in CP decomposition, Seyed et al. proposed SimplE. They believe that when an entity embedding learns as subject should also be used in learning as object, and vice versa. So each entity is represented as two vectors, and each relation is represented as two vectors accordingly.

2.2.6 TuckER

Enlightened by Tucker's thoughts, Balažević et al. propose TuckER, a simple linear model for link predictions in 2019. TuckER decomposes knowledge graph third-order binary tensor into a core tensor multiplied by a matrix along each mode. The rows in three matrices represent vectors of subject entities, relations and object entities respectively. And it assumed that embedding vectors of subject and object entities are equivalent. They do not distinguish between the embedding of an entity, whether it appears as subject or as object in the triple.

2.3 Deep learning models

Neural networks of deep learning are adopted for interactions between subject entities and object entities in triples. Sereval convolutional models have been proposed in natural language processing for solving a variety of tasks. In 2018, Dettmers et al. have applied convolutional neural network in link predictions as knowledge embedding approaches for the first time–ConvE. In ConvE, subject entities and relation vectors are transformed into a matrix, then spliced and sent to neural network for 2D convolution operation. The obtained

Table 1. Scoring functions of state-of-the-art link prediction models. In this chart, $e_i, h_j, t_k \in \mathbb{R}^{d_e}$, d_e, d_r are regarded as embedding dimensions of entities and relations respectively. And n_e, n_r are represented as numbers of entities and relations. $\overline{e}_o \in \mathbb{C}^{d_e}$ denote the complex conjugate of e_o, $\overline{e_s}$ and $\overline{r_r}$ denote a 2D reshaping of e_s and r_r, $\langle v, w, x\rangle = (v \odot w) \cdot x$ where $\odot$ reprents element-wise (Hadamard) multiplication and $\cdot$ represents dot product, $*$ denotes the convolution operator, f denotes a non-linear function, $\mathcal{W} \in \mathbb{R}^{d_e \times d_r \times d_e}$ and $\times_n$ denotes the tensor product along the n-th mode.

Model	Scoring Function	Relation Parameters	Space Complexity
TransE (Bordes et al. 2013)	$\|e_s + r_r - e_o\|_p$	$r_r \in \mathbb{R}^{d_e}$	$\mathcal{O}\left(n_e d_e + n_r d_e\right)$
RESCAL (Nickel et al. 2011)	$e_s^T W_r e_o$	$W_r \in \mathbb{R}^{{d_e}^2}$	$\mathcal{O}\left(n_e d_e + n_r d_r^2\right)$
DistMult (Yang et al. 2015)	$\langle e_s, r_r, e_o\rangle$	$r_r \in \mathbb{R}^{d_e}$	$\mathcal{O}\left(n_e d_e + n_r d_e\right)$
ComplEx (Trouillon et al. 2016)	$\mathrm{Re}\left(\langle e_s, r_r, \overline{e}_o\rangle\right)$	$r_r \in \mathbb{C}^{d_e}$	$\mathcal{O}\left(n_e d_e + n_r d_e\right)$
SimplE (Kazemi et al. 2018)	$\frac{1}{2}\left(\langle h_s, r_r, t_o\rangle + \langle h_o, r_{r^{-1}}, t_s\rangle\right)$	$r_r \in \mathbb{R}^{d_e}$	$\mathcal{O}\left(n_e d_e + n_r d_e\right)$
ConvE (Dettmers et al. 2018)	$f\left(\mathrm{vec}\left(f\left(\left[\overline{e_s}; \overline{r_r}\right] * w\right)\right) \mathrm{W}\right) e_o$	$r_r \in \mathbb{R}^{d_r}$	$\mathcal{O}\left(n_e d_e + n_r d_r\right)$
TuckER (Balažević et al. 2019)	$\mathcal{W} \times_1 e_s \times_2 r_r \times_3 e_o$	$r_r \in \mathbb{R}^{d_r}$	$\mathcal{O}\left(n_e d_e + n_r d_r\right)$
BTDE (ours)	$\mathcal{G}_1 \times_1 h_s \times_2 r_r \times_3 t_o + \mathcal{G}_2 \times_1 h_o \times_2 r_{r^{-1}} \times_3 t_s$	$r_r \in \mathbb{R}^{d_r}$	$\mathcal{O}\left(n_e d_e + n_r d_r\right)$

features then will be transformed into vectors. After being connected with fully connected layers, these vectors will be matched with object entities vector via an inner product. The performance of the model is significantly better than other models.

Table 1 shows scoring functions of some models described above.

3 BACKGROUND

3.1 Link prediction

Link prediction plays an essential part in evaluating performance of knowledge embedding models. It can predict the missing subject or object entity when given some other subject or object entity and relation in a triple. In link prediction, we use a subset of all true triples to learn the embedding vectors of entities and relations. And then score function $\varphi(s, r, o)$ are applied to determine whether the triplet (s, r, o) is true or false. It is the normal case that in some deep learning models or multiplicative models, score of triplet is positive so that the triplet is predicted to be true. And a negative score means that the triplet is false. For recent models, an activation function such as sigmod is applied for scoring function for probabilistic predictions to tell whether the triple is true or false.

3.2 Block term decomposition

In 2008, Lathauwer and Lieven have proposed Block term decomposition (BTD) [15], which is also called Block component decomposition (BCD) in other papers. In BTD, an N-order tensor is decomposed into R tensors. In a three-mode case, given a tensor $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$, it will be decomposed into R tensors just as shown in Figure 1. In mathematical expressions, it can be represented as:

$$\mathcal{X} \approx \sum_{r=1}^{R} \mathcal{G}_r \times_1 A_r \times_2 B_r \times_3 C_r \tag{1}$$

in which $\mathcal{G}_r \in \mathbb{R}^{L \times M \times N}$ are full rank-(L, M, N) and in which $A_r \in \mathbb{R}^{I \times L}$ (with $I \geq L$), $B_r \in \mathbb{R}^{J \times M}$ (with $J \geq M$), and $C_r \in \mathbb{R}^{K \times N}$ (with $K \geq N$) are full column rank, $1 \leq r \leq R$, $\times_n$ indicating the tensor product along the n-th mode.

In CP, the tensor $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$ will be decomposed as follow (Kolda and Bader 2009)[13]:

$$\mathcal{X} \approx \sum_{r=1}^{R} a_r \otimes b_r \otimes c_r \tag{2}$$

where R is a positive integer and $a_r \in \mathbb{R}^I$, $b_r \in \mathbb{R}^J$ and $c_r \in \mathbb{R}^K$, $1 \leq r \leq R$, $\otimes$ denotes the vector inner product.

In Tucker, the tensor $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$ will be decomposed as follow (Kolda and Bader 2009)[13]:

$$\begin{aligned} \mathcal{X} &\approx \mathcal{G} \times_1 A \times_2 B \times_3 C \\ &= \sum_{p=1}^{P} \sum_{q=1}^{Q} \sum_{r=1}^{R} g_{pqr} a_p \otimes b_q \otimes c_r \end{aligned} \tag{3}$$

in which $\mathcal{G} \in \mathbb{R}^{P \times Q \times R}$, $A \in \mathbb{R}^{I \times P}$, $B \in \mathbb{R}^{J \times Q}$, and $C \in \mathbb{R}^{K \times R}$.

Obviously, Block term decomposition can be seen as a combination of Tucker decomposition [29] and CP decomposition [10]. When $R = 1$, it is obvious that there is only one tensor involved, and this is a Tucker decomposition of rank-(L, M, N). When each tensor is a rank-1 decomposition, this will degenerate into a CP decomposition (CP decomposition is the decomposition of a tensor into R tensor with rank-1). This also reveals strong generalization ability of BTD.

4 BLOCK TERM DECOMPOSITION EMBEDDING IN KNOWLEDGE GRAPH

In this section, the knowledge embedded model BTDE based on Block term decomposition is displayed in detail. As the knowledge graph is essentially a set of triples, it can be transformed as a third-order binary tensor in math. The tensor coordinates correspond to the ids of entities and relations in triples. When triples are true, the corresponding coordinate value of third-order binary tensor will be 1, otherwise it will be 0. In this paper, knowledge graph is decomposed into the sum of two terms by BTDE. Meanwhile, $\mathrm{H} \in \mathbb{R}^{n_e \times d_e}$ and $\mathrm{T} \in \mathbb{R}^{n_e \times d_e}$ are subject and object entity embedded matrixes respectively where n_e denotes the number of entities and d_e denotes the dimensions of embedded vectors of each entity. An entity can be a subject entity or an object entity, and different roles mean different functions. But it does not mean that two embedded vectors of the same entity are completely independent. Just on the contrary, they are learned interactively during the training process. Each relation is represented by vectors of r_r and $r_{r^{-1}}$.

The scoring function of BTDE in this paper is as follows:

$$\begin{aligned} \varphi(s, r, o) = \mathcal{G}_1 \times_1 h_s \times_2 r_r \times_3 t_o \\ + \mathcal{G}_2 \times_1 h_o \times_2 r_{r^{-1}} \times_3 t_s \end{aligned} \tag{4}$$

$h_s, t_o \in \mathbb{R}^{d_e}$ are row vectors of matrix H and T, and $r_r, r_{r^{-1}} \in \mathbb{R}^{d_r}$ row vectors of relation matrixes. $\mathcal{G}_1 \in \mathbb{R}^{d_e \times d_r \times d_e}$ and $\mathcal{G}_2 \in$

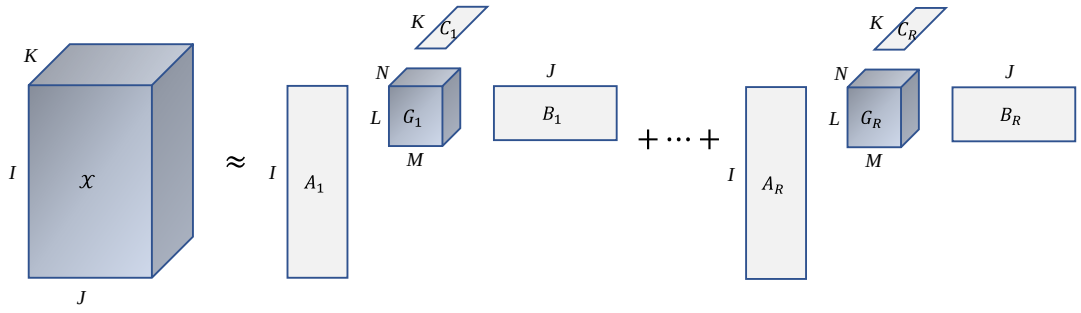

Figure 1. Visualization of the block term decomposition.

$\mathbb{R}^{d_e \times d_{r^{-1}} \times d_e}$ are core tensors of forward and reverse term respectively, and $\times_n$ is the tensor product along the n-th mode. Logistic sigmoid function $\sigma(\cdot)$ is applied to each score $\varphi(s, r, o)$ to obtain the predicted probability p of a triple being true.

4.1 Learning

Numerical methods are employed to fit subject entity matrix, relation matrix and object entity matrix in BTDE. We followed the training process of Dettmers et al. in this paper, training each triplet with 1-N. That is, for triples $(s, r, \cdot)$ and $(\cdot, r, o)$, the object or subject entities are replaced by all entities respectively. And if new generated triples in the training set, they will be marked as positive samples, otherwise negative samples. Inspired by Lacroix et al.[14], we designed the full binary cross-entropy loss, as shown in Equation 5. We use the Adam optimizer to train our model to minimize the loss:

$$L(p, y) = L(p, y)_{(s,r,o)} + L(p, y)_{(o,r^{-1},s)} \tag{5}$$

$$\begin{aligned} L(p, y)_{(s,r,o)} = -\frac{1}{n_e} \sum_{i=1}^{n_e} \Big[& y^{(i)} \log \left(p^{(i)}_{(s,r,o)} \right) \\ & + \left(1 - y^{(i)}\right) \log \left(1 - p^{(i)}_{(s,r,o)}\right) \Big] \end{aligned} \tag{6}$$

$$\begin{aligned} L(p, y)_{(o,r^{-1},s)} = -\frac{1}{n_e} \sum_{i=1}^{n_e} \Big[& y^{(i)} \log \left(p^{(i)}_{(o,r^{-1},s)} \right) \\ & + \left(1 - y^{(i)}\right) \log \left(1 - p^{(i)}_{(o,r^{-1},s)}\right) \Big] \end{aligned} \tag{7}$$

$$p_{(s,r,o)} = \sigma\left(\mathcal{G}_1 \times_1 h_s \times_2 r_r \times_3 t_o\right) \tag{8}$$

$$p_{(o,r^{-1},s)} = \sigma\left(\mathcal{G}_2 \times_1 h_o \times_2 r_{r^{-1}} \times_3 t_s\right) \tag{9}$$

where p is the vector of probabilities predicted by the model and y is the label vector of ones for true and zeros for false triples.

4.2 Full expressiveness

A tensor decomposition model can be said to be fully expressive if for any ground truth over all entities and relations, there exist entity and relation embeddings that correctly distinguish the true triple from the false ones. Trouillon et al. have demonstrated that ComplEx is fully expressive, while indicating that embedding dimensions of entities and relations are limited to $n_e * n_r$. Kazemi et al. have proved that SimplE is also fully expressive, and indicated that embedding dimensions of entities and relations are limited to $min\,(n_e * n_r, \gamma + 1)$, in which the number of all true triples is represented by γ. Balažević et al. have proved that TuckER is also fully expressive, while indicating that embedding dimensions of entities and relations are limited to n_e and n_r respectively. We show that BTDE is also fully expressive in following parts.

Theorem 1 *For any ground truth over entities $\mathcal{E}$ and relations $\mathcal{R}$, there exists a BTDE model with entity embedding vectors of size $d_e = n_e$ and relation embedding vectors of size $d_r = n_r$, where n_e is the number of entities and n_r is the number relations, that represents the ground truth.*

Proof 1 *In this paper, each entity e_i is represented by one-hot binary vector of n_e dimension, let the i-th element of $h_{e_i} = 1$, $t_{e_i} = 1$ and other elsements 0. As for each relation r, one-hot binary vector of n_r dimension, let the j-th element of $v_{r_j} = 1$, $v_{r_j^{-1}} = 1$ and other elements 0. If the triplet (e_s, r, e_o) is true, let the value of core tensor $\mathcal{G}_1 \in \mathbb{R}^{d_e \times d_r \times d_e}$, coordinated by (e_s, r, e_o), be 1 and other coordinations -1. And let the value of core tensor $\mathcal{G}_2 \in \mathbb{R}^{d_e \times d_{r^{-1}} \times d_e}$ coordinated by (e_o, r^{-1}, e_s) be 1 and other coordinations -1. Therefore, if the triplet is true during application of scoring function in this paper, the final score will be positive. Otherwise the final score will be negative. That is to say, BTDE can correctly represent ground truth.*

In the following part, symmetrical and asymmetrical relations will be employed to show that BTDE can correctly distinguish them.

1. If (e_s, r, e_o) and (e_o, r, e_s) are both true, then $\mathcal{G}_1(e_s, r, e_o) = 1$, $\mathcal{G}_2(e_o, r^{-1}, e_s) = 1$, $\mathcal{G}_1(e_o, r, e_s) = 1$, $\mathcal{G}_2(e_s, r^{-1}, e_o) = 1$,
   $\varphi\,(e_s, r, e_o) = \mathcal{G}_1 \times_1 h_{e_s} \times_2 r_r \times_3 t_{e_o} + \mathcal{G}_2 \times_1 h_{e_o} \times_2 r_{r^{-1}} \times_3 t_{e_s} = 2 > 0$,
   $\varphi\,(e_o, r, e_s) = \mathcal{G}_1 \times_1 h_{e_o} \times_2 r_r \times_3 t_{e_s} + \mathcal{G}_2 \times_1 h_{e_s} \times_2 r_{r^{-1}} \times_3 t_{e_o} = 2 > 0$.
2. If (e_s, r, e_o) is true while (e_o, r, e_s) is false, then $\mathcal{G}_1(e_s, r, e_o) = 1$, $\mathcal{G}_2(e_o, r^{-1}, e_s) = 1$, $\mathcal{G}_1(e_o, r, e_s) = -1$, $\mathcal{G}_2(e_s, r^{-1}, e_o) = -1$,
   $\varphi\,(e_s, r, e_o) = \mathcal{G}_1 \times_1 h_{e_s} \times_2 r_r \times_3 t_{e_o} + \mathcal{G}_2 \times_1 h_{e_o} \times_2 r_{r^{-1}} \times_3 t_{e_s} = 2 > 0$,
   $\varphi\,(e_o, r, e_s) = \mathcal{G}_1 \times_1 h_{e_o} \times_2 r_r \times_3 t_{e_s} + \mathcal{G}_2 \times_1 h_{e_s} \times_2 r_{r^{-1}} \times_3 t_{e_o} = -2 < 0$.
3. If (e_s, r, e_o) is false while (e_o, r, e_s) is true, then $\mathcal{G}_1(e_s, r, e_o) = -1$, $\mathcal{G}_2(e_o, r^{-1}, e_s) = -1$, $\mathcal{G}_1(e_o, r, e_s) = 1$, $\mathcal{G}_2(e_s, r^{-1}, e_o) =$

1,
$\varphi(e_s, r, e_o) = \mathcal{G}_1 \times_1 h_{e_s} \times_2 r_r \times_3 t_{e_o} + \mathcal{G}_2 \times_1 h_{e_o} \times_2 r_{r^{-1}} \times_3 t_{e_s} = -2 < 0$,
$\varphi(e_o, r, e_s) = \mathcal{G}_1 \times_1 h_{e_o} \times_2 r_r \times_3 t_{e_s} + \mathcal{G}_2 \times_1 h_{e_s} \times_2 r_{r^{-1}} \times_3 t_{e_o} = 2 > 0$.

4. If (e_s, r, e_o) and (e_o, r, e_s) are both false, then $\mathcal{G}_1(e_s, r, e_o) = -1$, $\mathcal{G}_2(e_o, r^{-1}, e_s) = -1$, $\mathcal{G}_1(e_o, r, e_s) = -1$, $\mathcal{G}_2(e_s, r^{-1}, e_o) = -1$,
$\varphi(e_s, r, e_o) = \mathcal{G}_1 \times_1 h_{e_s} \times_2 r_r \times_3 t_{e_o} + \mathcal{G}_2 \times_1 h_{e_o} \times_2 r_{r^{-1}} \times_3 t_{e_s} = -2 < 0$,
$\varphi(e_o, r, e_s) = \mathcal{G}_1 \times_1 h_{e_o} \times_2 r_r \times_3 t_{e_s} + \mathcal{G}_2 \times_1 h_{e_s} \times_2 r_{r^{-1}} \times_3 t_{e_o} = -2 < 0$.

5 EXPERIMENTS AND RESULTS

5.1 Datasets

Our experiments are conducted on four public knowledge graph datasets retrieved from WordNet and Freebase corpus.

FB15k[3] is the subset of Freebase. Most part of its facts are about movies, actors, awards, sports and sports teams.

FB15k-237[27] is the subset of FB15k, and reverse relations are removed from it. A host of testing triples will be acquired through reverse trainings from FB15k. In order to create a data set without such attributes, Toutanova et al. have proposed FB15k-237.

WN18[3] is the subset of WordNet. Entities in WordNet are synonymous with different concepts, and relations indicate lexical relationships between synonyms.

WN18RR[6] is the subset of WN18. Dettmers et al. have discovered the same deficiency of FB15k in WN18, and constructed the dataset from removing inverse relations between validation and test sets from WN18.

YAGO3-10[19] is a subset of YAGO3, which consists of entities with at least 10 relations per entity. Most of the triples involve human description attributes such as citizenship, gender, and occupation.

The statistics of the four datasets are listed in Table 2.

Table 2. Dataset statistics

Datatset	#Entities	#Relations	#Trains	#Valids	#Tests
FB15k	14,951	1,345	483,142	50,000	59,071
FB15k-237	14,541	237	272,115	17,535	20,466
WN18	40,943	18	141,442	5,000	5,000
WN18RR	40,943	11	86,835	3,034	3,134
YAGO3-10	123,182	37	1,079,040	5,000	5,000

5.2 Experiments

In order to evaluate different models, for each test triple (e_i, r, e_j), we replace its source entity e_i with every entity $e_i' \in \mathcal{E}$, calculate the scores of test triple (e_i, r, e_j) and every corrupted triple (e_i', r, e_j), and compute the rank of test triple among these corrupted triples. Because of the existence of one-to-many, many-to-one and many-to-many structures in knowledge graph, there may be a phenomenon that other correct fake triples rank ahead of the ground truth. So, we follow Bordes et al. to report the filtered results. During ranking, we remove the corrupted triples that already exist in either the train, valid and test sets. Similarly, the object entity e_j of the test triple will be replaces as well. For measuring the performances of the models, we use four main evaluation protocols: MRR, Hits@1, Hits@3 and Hits@10. MRR is Mean Reciprocal Rank, which means the average of reciprocal ranks. Compared with MR(Mean Rank), MRR is more stable and less susceptible to a single bad rank. And Hits@K represents the percentage of the top K in all ranks. Higer MRR and higer Hits@K mean better performance.

We implemented BTDE in PyTorch and did the experiments by using a single GPU (NVIDIA Titan Xp). We tuned our hyperparameters over the validation set. For FB15k and FB15k-237, we set entity and relation dimensions to 150. For WN18 and WN18RR, as the number of relations in these two datasets is obviously less than the amount in FB15k and FB15k-237, we set the entity dimension to 200 and the relation dimension to 30. Since the entity and relation ratios in YAGO3-10 are roughly the same as WN18RR, we follow the embedding setting 200 and 30 in WN18RR. We use batch normalization, dropout and label smoothing to lessen overfitting. In particular, we use dropout on the embeddings and on the hidden units after the head embedding and relation embedding interact with the core tensor, respectively. We set the batch size to 1024. We select the learning rate λ among $\{0.0003, 0.003, 0.005, 0.01\}$, the learning decay γ among $\{0.99, 0.995, 1.0\}$, embedding dropout $\{0.0, 0.1, 0.2\}$, hidden dropout $\{0.0, 0.1, 0.2, 0.3, 0.4, 0.5\}$ and label smoothing $\{0.0, 0.1, 0.2, 0.3\}$.

We found that the optimal configurations of BTDE are as follows: $\lambda = 0.0005, \gamma = 1.0$ on WN18, $\lambda = 0.005, \gamma = 0.995$ on FB15k, $\lambda = 0.01, \gamma = 1.0$ on WN18RR, $\lambda = 0.003, \gamma = 1.0$ on FB15k-237 and $\lambda = 0.01, \gamma = 1.0$ on YAGO3-10. For all datasets, we set the embedding dropout 0.2 and label smoothing 0.1 except label smoothing 0.0 on FB15k. Lower hidden dropout values work well on WN18(0.1, 0.2), WN18RR(0.2, 0.3) and YAGO3-10(0.2, 0.3). For FB15k-237, the higher hidden dropout values (0.4, 0.5) are needed to control overfitting.

5.3 Link prediction results

Link prediction results for all four datasets are listed in Tables 3 and 4. It can be seen that BTDE has achieved the most desired results compared to other existing baselines on the four datasets (excluding Hits@10 on WN18). In the series of tensor decomposition approaches, it is as plain as daylight that BTDE is optimal compared with DistMult, HolE, ComplEx, SimplE and TuckER. Especially when compared with WN18RR, BTDE increased by 1.5% on MRR and 2.5% on Hits@10 than TuckER, and 4.5% on MRR and 4.1% on Hits@10 than ComplEx. Compared to deep learning based methods such as ConvE, BTDE also shows better performance. These results have also verified that tensor decomposition based model can better represent complexities in knowledge representation and achieve state-of-the-art results.

In 2015, Toutanova and Chen[27] first indicated that WN18 and FB15k suffered test leaks through an inverse relations. That is, one only need to invert the triples in the training set can get a large number of test triples. Just like the case, a triple in test set such as (s, r, o) while the training set contains its inverse (o, r, s). Such cases occur frequently in these two datasets. Therefore, the experimental results on the two datasets FB15k and WN18 are not very convincing. To further validate the superiority of our model, we performed experiments on the dataset YAGO3-10 recommended by Dettmeter et al. The experimental results are shown in Table 5. Obviously our model is better than the others. Compared to ConvE, BTDE increased by 2.7% on MRR and 3.6% on Hits@10. In contrast with D4-STE, BTDE increased by 7.5% on MRR and 5.3% on Hits@10. Our proposed model, BTDE, achieves the state-of-the-art performance for

Table 3. Results on WN18 and FB15k. The best results are in bold, while second best results are underlined.

	WN18				FB15k			
	MRR	Hits@10	Hits@3	Hits@1	MRR	Hits@10	Hits@3	Hits@1
TransE (Bordes et al. 2013)	-	.892	-	-	.380	.471	-	-
DistMult (Yang et al. 2015)	.822	.936	.914	.728	.654	.824	.733	.546
HolE (Nickel et al. 2016)	.938	.949	.945	.930	.524	.739	.613	.402
ComplEx (Trouillon et al. 2016)	.941	.947	.936	.936	.692	.840	.756	.599
Neural LP (Yang et al. 2017)[34]	.940	.945	-	-	.760	.837	-	-
ANALOGY (Liu et al. 2017)[18]	.942	.947	.944	.939	.725	.854	.785	.646
RUGE (Guo et al. 2018)[9]	-	-	-	-	.768	.865	.815	.703
R-GCN (Schlichtkrull et al. 2018)[24]	.819	**.964**	.929	.697	.696	.842	.760	.601
TorusE (Ebisu and Ichise 2018)[7]	.947	.954	.950	.943	.733	.832	.771	.674
ConvE (Dettmers et al. 2018)	.943	.956	.946	.935	.657	.931	.723	.558
SimplE (Kazemi et al. 2018)	.942	.947	.944	.939	.727	.838	.773	.660
CrossE (Zhang et al. 2019)[35]	.830	.950	.931	.741	.728	.875	.802	.634
D4-STE (Xu et al. 2019)[32]	.946	.952	.948	.942	.733	.877	.803	.641
TuckER (Balažević et al. 2019)	.953	.958	.955	.949	.795	.892	.833	.741
BTDE (ours)	**.954**	.961	**.956**	**.950**	**.800**	**.897**	**.840**	**.742**

Table 4. Results on WN18RR and FB15k-237. The best results are in bold, while second best results are underlined.

	WN18RR				FB15k-237			
	MRR	Hits@10	Hits@3	Hits@1	MRR	Hits@10	Hits@3	Hits@1
DisMult (Yang et al. 2015)	.430	.490	.440	.390	.241	.419	.263	.155
ComplEx (Trouillon et al. 2016)	.440	.510	.460	.410	.247	.428	.275	.158
Neural LP (Yang et al. 2017)	-	-	-	-	.250	.408	-	-
ANALOGY (Liu et al. 2017)	-	-	-	-	.219	.405	.240	.131
R-GCN (Schlichtkrull et al. 2018)	-	-	-	-	.248	.417	.262	.151
MINERVA (Das et al. 2018)[5]	-	-	-	-	-	.456	-	-
ConvE (Dettmers et al. 2018)	.430	.520	.440	.400	.325	.501	.356	.237
M-Walk (Shen et al. 2018)[25]	.437	-	.445	.414	-	-	-	-
CrossE (Zhang et al. 2019)	-	-	-	-	.299	.474	.331	.211
D4-STE (Xu et al. 2019)	.480	.536	.491	**.452**	.320	.502	.353	.230
TuckER (Balažević et al. 2019)	.470	.526	.482	.443	.358	.544	.394	.266
BTDE (ours)	**.485**	**.551**	**.501**	**.452**	**.367**	**.553**	**.402**	**.274**

Table 5. Results on YAGO3-10. The best results are in bold, while second best results are underlined.

	YAGO3-10			
	MRR	Hits@10	Hits@3	Hits@1
DisMult (Yang et al. 2015)	.340	.540	.380	.240
ComplEx (Trouillon et al. 2016)	.360	.550	.400	.260
Neural LP (Yang et al. 2017)	-	-	-	-
ANALOGY (Liu et al. 2017)	-	-	-	-
R-GCN (Schlichtkrull et al. 2018)	-	-	-	-
MINERVA (Das et al. 2018)	-	-	-	-
ConvE (Dettmers et al. 2018)	.520	.660	.560	.450
M-Walk (Shen et al. 2018)	-	-	-	-
CrossE (Zhang et al. 2019)	-	-	-	-
D4-STE (Xu et al. 2019)	.472	.643	.523	.381
TuckER (Balažević et al. 2019)	-	-	-	-
BTDE (ours)	**.547**	**.696**	**.596**	**.465**

all metrics on YAGO3-10.

6 ANALYSIS

6.1 Ablation study

Table 6 shows the results of our ablation study, in which we evaluated different parameter initializations (n = 3) to calculate confidence intervals. We see that hidden dropout 2 is by far the most important component, which is not surprising, as it is our main regularization technique. Embedding dropout , hidden dropout 1 and label smoothing improve performance but seem to be negligible, and we can get good results without these components.

Table 6. Ablation study for FB15k-237.

Ablation	Hits@10
Full BTDE	0.553
Hidden dropout 2	-0.036 ± 0.000
Embedding dropout	-0.004 ± 0.000
Hidden dropout 1	-0.003 ± 0.000
Label smoothing	-0.001 ± 0.000

6.2 Influence of embedding dimensionality

The full expressiveness of BTDE is demonstrated in previous section of this paper. At the same time, the embedding dimensions of entity and relation of BTDE are the same as TuckER, which are much lower than ComplEx and SimplE. In order to safely land on this conclusion, contrast experiments on FB15k-237 are conducted and evaluation indexes, MRR and Hits@10 are observed in different dimensions {50, 100, 150, 200}. As is shown in Figure 2 and Figure 3, MRR and Hits@10 of BTDE excel other three models in both low and high dimensions, further highlighting its superior performance.

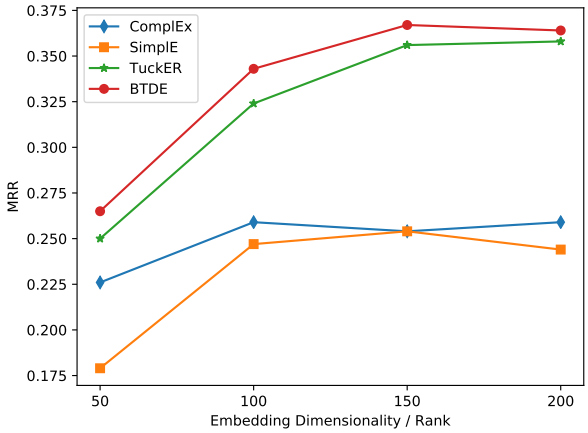

Figure 2. MRR for ComplEx, SimplE, TuckER and BTDE for different embedding sizes {50,100,150,200} on FB15k-237.

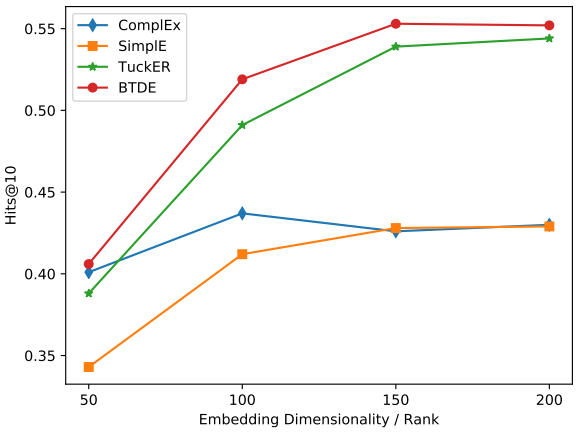

Figure 3. Hits@10 for ComplEx, SimplE, TuckER and BTDE for different embedding sizes {50,100,150,200} on FB15k-237.

7 CONCLUSION

In this paper, we propose BTDE based on Block component decomposition of third-order binary tensors for link predictions. Block term decomposition is a more generalized decomposition in terms of mathematical tensor decomposition, and it can better decompose the formed factors more accurately. Based on this, We can get more accurate embedded vectors of entity and relation and our experimental results also prove this. We also prove that BTDE is fully expressive as ComplEx, SimplE and TuckER, while BTDE also obtains lower embedding dimensions of entity and relation. Besides, BTDE has achieved excellent results on all five standard datasets.

In the future, we aim to explore whether the third-order binary tensor can be decomposed the sum of more terms and not just the forward and the reverse parts. We will also explore a good way to add background knowledge into BTDE.

ACKNOWLEDGEMENTS

This work is supported in part by National Natural Science Foundation of China (No.61877043) and the National Natural Science Foundation of China (No.61877044).

REFERENCES

[1] Ivana Balažević, Carl Allen, and Timothy M Hospedales, 'Tucker: Tensor factorization for knowledge graph completion', in *Empirical Methods in Natural Language Processing*, (2019).

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

[3] Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko, 'Translating embeddings for modeling multi-relational data', in *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013*, pp. 2787–2795, (2013).

[4] Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka Jr., and Tom M. Mitchell, 'Toward an architecture for never-ending language learning', in *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010*, (2010).

[5] 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 *6th International Conference on Learning Representations, ICLR*, (2018).

[6] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel, 'Convolutional 2d knowledge graph embeddings', in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18)*, pp. 1811–1818, (2018).

[7] Takuma Ebisu and Ryutaro Ichise, 'Toruse: Knowledge graph embedding on a lie group', in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18)*, pp. 1819–1826, (2018).

[8] Jun Feng, Minlie Huang, Mingdong Wang, Mantong Zhou, Yu Hao, and Xiaoyan Zhu, 'Knowledge graph embedding by flexible translation', in *Fifteenth International Conference on the Principles of Knowledge Representation and Reasoning*, (2016).

[9] Shu Guo, Quan Wang, Lihong Wang, Bin Wang, and Li Guo, 'Knowledge graph embedding with iterative guidance from soft rules', in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18)*, pp. 4816–4823, (2018).

[10] 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).

[11] Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao, 'Knowledge graph embedding via dynamic mapping matrix', in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics, ACL*, pp. 687–696, (2015).

[12] Seyed Mehran Kazemi and David Poole, 'Simple embedding for link prediction in knowledge graphs', in *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS*, pp. 4289–4300, (2018).

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

[14] Timothée Lacroix, Nicolas Usunier, and Guillaume Obozinski, 'Canonical tensor decomposition for knowledge base completion', in *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pp. 2869–2878, (2018).

[15] De Lathauwer and Lieven, 'Decompositions of a higher-order tensor in block termsłpart ii: Definitions and uniqueness', *SIAM Journal on Matrix Analysis and Applications*, **30**(3), 1033–1066, (2008).

[16] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al., 'Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia', *Semantic Web*, **6**(2), 167–195, (2015).

[17] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu, 'Learning entity and relation embeddings for knowledge graph completion', in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pp. 2181–2187, (2015).

[18] Hanxiao Liu, Yuexin Wu, and Yiming Yang, 'Analogical inference for multi-relational embeddings', in *Proceedings of the 34th International Conference on Machine Learning, ICML*, pp. 2168–2178, (2017).

[19] Farzaneh Mahdisoltani, Joanna Biega, and Fabian M. Suchanek, 'YAGO3: A knowledge base from multilingual wikipedias', in *CIDR 2015, Seventh Biennial Conference on Innovative Data Systems Research, Asilomar, CA, USA, January 4-7, 2015, Online Proceedings*, (2015).

[20] George A. Miller, 'Wordnet: A lexical database for english', *Commun. ACM*, **38**(11), 39–41, (1995).

[21] Dat Quoc Nguyen, Kairit Sirts, Lizhen Qu, and Mark Johnson, 'Stranse: a novel embedding model of entities and relationships in knowledge bases', in *NAACL HLT, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 460–466, (2016).

[22] Maximilian Nickel, Lorenzo Rosasco, and Tomaso A. Poggio, 'Holographic embeddings of knowledge graphs', in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pp. 1955–1961, (2016).

[23] 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 Machine Learning, ICML*, pp. 809–816, (2011).

[24] Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling, 'Modeling relational data with graph convolutional networks', in *The Semantic Web - 15th International Conference, ESWC*, pp. 593–607, (2018).

[25] Yelong Shen, Jianshu Chen, Po-Sen Huang, Yuqing Guo, and Jianfeng Gao, 'M-walk: Learning to walk over graphs using monte carlo tree search', in *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS*, pp. 6787–6798, (2018).

[26] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum, 'Yago: a core of semantic knowledge', in *Proceedings of the 16th International Conference on World Wide Web, WWW*, pp. 697–706, (2007).

[27] Kristina Toutanova and Danqi Chen, 'Observed versus latent features for knowledge base and text inference', in *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*, pp. 57–66, (2015).

[28] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard, 'Complex embeddings for simple link prediction', in *Proceedings of the 33nd International Conference on Machine Learning, ICML*, pp. 2071–2080, (2016).

[29] L. R. Tucker, 'The extension of factor analysis to three-dimensional matrices', *Contributions to mathematical psychology*, **110119**, (1964).

[30] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen, 'Knowledge graph embedding by translating on hyperplanes', in *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, pp. 1112–1119, (2014).

[31] Han Xiao, Minlie Huang, and Xiaoyan Zhu, 'From one point to a manifold: Knowledge graph embedding for precise link prediction', in *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI*, pp. 1315–1321, (2016).

[32] Canran Xu and Ruijiang Li, 'Relation embedding with dihedral group in knowledge graph', in *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL*, pp. 263–272, (2019).

[33] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng, 'Embedding entities and relations for learning and inference in knowledge bases', in *3rd International Conference on Learning Representations, ICLR*, (2015).

[34] Fan Yang, Zhilin Yang, and William W. Cohen, 'Differentiable learning of logical rules for knowledge base reasoning', in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems*, pp. 2319–2328, (2017).

[35] Wen Zhang, Bibek Paudel, Wei Zhang, Abraham Bernstein, and Huajun Chen, 'Interaction embeddings for prediction and explanation in knowledge graphs', in *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, WSDM*, pp. 96–104, (2019).