Journal of Intelligent Information Systems
https://doi.org/10.1007/s10844-019-00592-7

Improve the translational distance models for knowledge graph embedding

Siheng Zhang[1,2] · Zhengya Sun[1] · Wensheng Zhang[1,3]

Received: 20 August 2019 / Revised: 23 December 2019 / Accepted: 27 December 2019 /
Published online: 27 January 2020

Abstract

Knowledge graph embedding techniques can be roughly divided into two mainstream, translational distance models and semantic matching models. Though intuitive, translational distance models fail to deal with the circle structure and hierarchical structure in knowledge graphs. In this paper, we propose a general learning framework named TransX-pa, which takes various models (TransE, TransR, TransH and TransD) into consideration. From this unified viewpoint, we analyse the learning bottlenecks are: (i) the common assumption that the inverse of a relation r is modelled as its opposite $-r$; and (ii) the failure to capture the rich interactions between entities and relations. Correspondingly, we introduce position-aware embeddings and self-attention blocks, and show that they can be adapted to various translational distance models. Experiments are conducted on different datasets extracted from real-world knowledge graphs *Freebase* and *WordNet* in the tasks of both triplet classification and link prediction. The results show that our approach makes a great improvement, showing a better, or comparable, performance with state-of-the-art methods.

Keywords Knowledge graph embedding · Translational distance model · Positional encoding · Self-attention

✉ Wensheng Zhang
wensheng.zhang@ia.ac.cn

Siheng Zhang
zhangsiheng2015@ia.ac.cn

Zhengya Sun
zhengya.sun@ia.ac.cn

[1] Research Center of Precision Sensing and Control, Institute of Automation, Chinese Academy of Sciences, Beijing, China

[2] University of Chinese Academy of Sciences, Beijing, China

[3] School of Mathematics and Big Data, Foshan University, Foshan, China

1 Introduction

A typical knowledge graph (KG) consists of a set of interconnected typed entities and their attributes. Usually, entities are modelled as nodes, and relations are modelled as different types of edges, linking from a head entity to a tail entity, denoted as (*head, relation, tail*) or (*h, r, t*). Although it is well defined and structured, KGs retain the underlying symbolic nature, which makes it difficult to automatically construct or inference on it. To tackle this issue, lots of work has been carried out on knowledge graph embedding. The key idea is to use distributed representation, i.e., embed entities and relations into continuous low-dimensional space, so that manipulation on KG can be simplified as algebraic operations (Nickel et al. 2016).

Roughly speaking, embedding techniques in this sort can be divided as two groups: *translational distance models* and *semantic matching models* (Wang et al. 2017). Our work follows the route of the first one, which measures the plausibility of a fact as the distance between the head and tail entities after a translation. Note that some other methods leverage addtional information, e.g., entity type, dependency path and etc. (Toutanova and Chen 2015), but these methods are out of our scope.

Among the translational distance models, TransE (Bordes et al. 2013) is the simplest but representative one. It models the relation as a translation and enforces the tranlated head entity to meet with the tail entity, i.e., $h + r = t$. Later on, TransH (Wang et al. 2014), TransR (Lin et al. 2015) and TransD (Ji et al. 2015) consider the different semantics of an entity linked with different relations. However, they all fail to handle some special structures of KG. We divide these structures into two classes: circle structure and hierarchical structure.

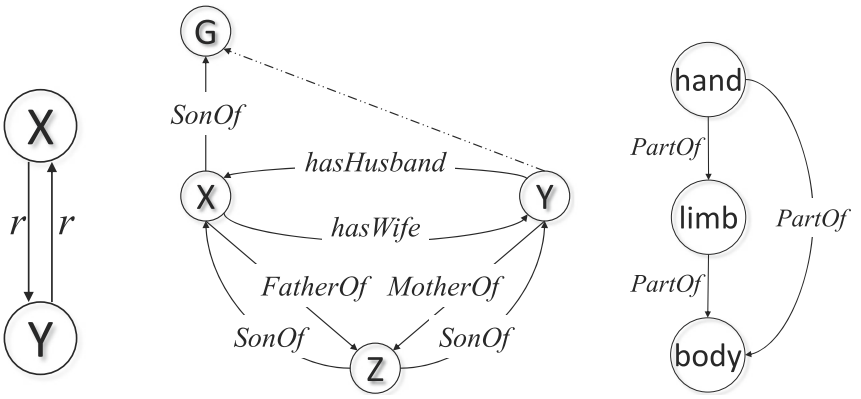
circle structure: Zhang (2017) noticed that 'One-Relation-Circle' (ORC) structure leads to the decline of performance. Take the knowledge graph in Fig. 1a as an example. Following the idea of existing translational distance models, $X+r = Y$ and $Y+r = X$ should hold, resulting a confusion of entities $X = Y$ and a degenerated representation of the relation $r = 0$.

In fact, however, the structure with multiple circles composed of different relations (we call it 'Multiple-Relation-Circles', or MRC) is also to blame. Consider the example in Fig. 1b, the facts $X + FatherOf = Z$, $Z + SonOf = X$ lead to $FatherOf = -SonOf$, and similarly, $MotherOf = -SonOf$. It comes to an unexpected confusion of relations that $MotherOf = FatherOf$. If taking all triplets into account, we will come to a result that $X = Y$, $hasHusband = hasWife = 0$, and even a false prediction that Y is also a son of G.

hierarchical structure: Hierarchical structures are common in the knowledge graph, especially in professional domains. Look into the example in Fig. 1c. Applying translational distance models, $hand + PartOf = limb$, $limb + PartOf = body$ and $hand + PartOf = body$, which leads to $v(PartOf) = 0$, and worse yet, we cannot distinguish all the entities $hand, limb$ and $body$.

The reasons that translational distance models fail in these cases are two folds. Firstly, an entity has the same representation regardless of its position, i.e., whether it acts as a head or a tail. According to the examples aforementioned, this implies an assumption that the inverse relation is modelled using an opposite vector. Secondly, the models only consider the effect that a relation acts on the head and tail entity, but neglect the opposite. However, there exists complex interactions between the relation and entities.

In this paper, we formulate a more general framework for large-scale knowledge graph representation. Corresponding to the two drawbacks aforementioned, we introduce two



(a) circle structure (b) multiple circles structure (c) hierarchical structure

Fig. 1 Three kinds of special structure of knowledge graph. **a**: in 'One-Relation-Circle' (ORC) structure, the enforcement of translational distance models leads to $r = 0$. Note that X can be the same with Y, forming a self-loop. **b**: an example of the 'Multiple-Relation-Circles' (MRC) structure. There are four circles, X-Y-X, X-Z-X, Y-Z-Y and X-Y-Z-X. The enforcement of translational distance models leads to confusion not only about entities but also about relations. **c** Hierarchical structure is the worst case for the translational distance models, which leads to a conclusion that $PartOf = 0$, and $hand = limb = body$

improvements in the framework: 1) **p**osition-aware entity embeddings: we enable an entity to have different semantic when acting as head and tail; 2) **a**ttention mechanism: we introduce a self-attention block to capture the rich interactions among relations and entities. For simplicity, we call it **TransX-pa**.

The main contributions of this paper are summarized as follows:

- We propose a general framework to unify the translational distance models. We investigate various models (TransE, TransH, TransR and TransD) and show that they are the special cases of our framework.
- Under the framework, we introduce positional-aware entity embeddings and self-attention block to deal with special structures (circle structure and hierarchical structure) in knowledge graphs. Besides, we introduce l_1-norm regularizer to guarantee the consistence of same entities.
- Extensive experiments have proved that with a little increase of computational cost, our proposed framework improves the existing models, achieving a better, or comparable, performance with state-of-the-art methods.

The rest of this paper is organized as follows. Section 2 briefly introduces translational distance models from a unified view, also it summarizes other state-of-the-art methods. Section 3 describes our proposed framework mathematically, applies it for different models and discusses the time and space complexity. Experimental study is presented in Section 4. We conclude the paper in Section 5.

2 Related work

Knowledge graph embedding (KGE) hires a low-dimensional vector to represent an entity or a relation in a knowledge graph. Early models for KGE include Structured Embedding

(SE) (Bordes et al. 2011), Semantic Matching Energy (SME) (Bordes et al. 2014) and so on. SE sets matrices to project head and tail entities for each relation. SME (linear and bilinear version) hires a neural network to capture the interactions between entities and relations via matrix operations. These models suffer from a high computational cost and low representation capacity.

In the following, we will introduce some representative works that are most relevant to our work.

2.1 Translational distance models

Translational distance models, which exploit distance-based scoring functions, significantly reduce the computational cost. Before proceeding, we define our notations. A column vector $\mathbf{e}$ is used to represent the embeddings of an entity e, and the column vectors $\mathbf{h} \in \mathcal{R}^m, \mathbf{r} \in \mathcal{R}^n, \mathbf{t} \in \mathcal{R}^m$ for a triplet (h, r, t) (m can be the same with n).

The residual of a triplet is:

$$\delta(h, r, t) = f_r(\mathbf{h}) + \mathbf{r} - f_r(\mathbf{t}) \tag{1}$$

in which the function $f_r(\cdot)$ is a relational-specific linear transformation. The scoring function is:

$$s(h, r, t) = \|\delta(h, r, t)\|_p \tag{2}$$

Usually, the p-norm can be l_1-norm or l_2-norm. Note that all the models share a common constraint, i.e., enforcing entities embeddings to have, at most, a unit l_2 norm.

TransE (Bordes et al. 2013) simplifies $f_r(\cdot)$ as an identity function, and $s(h, r, t) = \|\mathbf{h}+\mathbf{r}-\mathbf{t}\|_p$. Despite its simplicity and efficiency, TransE cannot handle with 1-to-N, N-to-1, and N-to-N relations ('N' represents for 'many').

To address this problem, an entity is allowed to have distinct representations when involved in different relations. Typical models include the following:

TransH (Wang et al. 2014) introduces relation-specific hyperplanes. It restricts the relation as a vector $\mathbf{r}$ on a hyperplane, which takes $\mathbf{w}_r$ as its normal vector. So the entities are projected onto the hyperplane, i.e., $f_r(\mathbf{e}) = \mathbf{e} - \mathbf{w}_r^\top \mathbf{e} \mathbf{w}_r$.

TransR (Lin et al. 2015) introduces relation-specific spaces, rather than hyperplanes. It associates each relation with a specific space and uses a projection matrix $\mathbf{M}_r$ to project the entities into the space, i.e., $f_r(\mathbf{e}) = \mathbf{M}_r \mathbf{e}$. Though representative, it suffers a large amount of parameters and so is hard to optimize. To address this problem, **TranSparse** (Ji et al. 2016) enforces sparseness on the projection matrix. It also varies the matrices for head and tail entities and find out to achieve a higher performance.

TransD (Ji et al. 2015) considers a more delicate projection method. The model assumes that the projection is associated not only with the relation but also with the entity itself. It then decompose the projection matrix in TransR as $\mathbf{M}_r = \mathbf{w}_r \mathbf{w}_e^\top + \mathbf{I}$.

Besides allowing distinct representations of an entity involved in different relations, some models improve TransE by relaxing the requirement that $\mathbf{h} + \mathbf{r} = \mathbf{t}$. **TransM** (Fan et al. 2014) assigns a relation-specific weight θ_r for each triplet, i.e., $f_r(h, t) = -\theta_r \|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_p$. With lower weight, it allows $\mathbf{t}$ to lie farther away from $\mathbf{h} + \mathbf{r}$. **ManifoldE** (Xiao et al. 2016) enlarges the point $\mathbf{h}+\mathbf{r}$ into a hyper-sphere with a relation-specific radius θ_r, and $\mathbf{t}$ is allowed to lie on this manifold. **KG2E** (He et al. 2015) takes uncertainty into consideration and hence represents an entity and relation with a Gaussian distribution.

Although the modified models show higher performance than TransE, they cannot deal with special structures in knowledge graphs appropriately. Zhang (2017) noticed the One-Relation-Circle (ORC) structure is to blame and suggest to decompose them. Ahead, we have pointed out that not only all circle structures but also hierarchical structures need to be handled carefully. Worse more, all the existing models still only take into account the effect from relation to entity, but neglect the effect from entity to relation.

2.2 Semantic matching models and the others

Semantic matching models exploit similarity-based scoring functions instead of distance-based scoring functions. Yang et al. (2015) argued that additive interactions between the relation and entities, which is hired in the translational distance models, are lack of rich representations. Instead, they proposed **DistMult**, which hires multiplicative interactions score a triplet. Later on, **HolE** (Holographic Embeddings) (Nickel et al. 2016) composes the entity representation into a compact feature space using circular correlation, which can be treated as a compression of pairwise interactions. **ComplEx** (Complex Embeddings) (Théo et al. 2016) extends DistMult by introducing complex-valued embeddings instead of real-valued. It has been proved that HolE is subsumed by ComplEx as a special case (Hayashi and Shimbo 2017). **ConvE** (Dettmers et al. 2018) hires a convolutional neural network to extract deep representative features for scoring the triplets. These models lead to our idea that translational distance models should allow bi-directional interactions between entities and relations.

There exists some other models hiring additional information from knowledge graphs. For example, **PTransE** (Path-based TransE) (Lin et al. 2015) composites each relation path and scores its reliability, **NTN** (Neural Tensor Network) (Socher et al. 2013) initializes entity embeddings from the textual information, **TEKE** (Text-Enhanced KG Embeddings) (Wang and Li 2016) constructs a co-occurrence network composed of entities and words to learn a more expressive representation. Additional information can help improve the final performance, but is out of the scope of this paper.

3 The proposed method

According to the analysis above, the improvements of our proposed framework contain two aspects. First, an entity is allowed to have different representations when acting as head and tail. This is expanded from the findings stated in Ji's work (2016) and Zhang's work (2017). Second, a self-attention block (Vaswani et al. 2017) is introduced to refine the score of a triplet. It enables a complicated interactions in a triplet, and hence can solve the problem that the existing translational distance models only capture additive interactions. To this end, we have done the following works to present the framework.

3.1 Models

The general framework of translational distance models can be extended as follow. We use $f_r^{(1)}(\cdot)$ to transform the head entities and $f_r^{(2)}(\cdot)$ to transform the tail (the subscript r indicates that they are relation-specific). Then, the residual of a triplet (h, r, t) is:

$$\delta(h, r, t) = f_r^{(1)}(\mathbf{h}) + \mathbf{r} - f_r^{(2)}(\mathbf{t}) \tag{3}$$

Then, before the final score, we employ a self-attention block (Vaswani et al. 2017) over the residual. The self-attention block is a special case of the multi-head attention mechanism, which is a collection of multiple attention blocks.

$$\begin{aligned} \text{MultiHead}(Q, K, V) &= \text{head}_1 \oplus \text{head}_2 \oplus \cdots \oplus \text{head}_k \\ \text{head}_i &= \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \end{aligned} \tag{4}$$

in which $\oplus$ represents a concatenation of vectors. If $Q = K = V$, (4) degenerates into a self-attention block.

Following the experience of Vaswani et al. (2017), each attention cell is scaled by $1/\sqrt{d}$ to avoid meaningless gradients during training:

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d}})V \tag{5}$$

After that, we come to the final score function:

$$s(h, r, t) = \|\text{SelfAttention}(\delta(h, r, t))\|_p \tag{6}$$

In this paper, we take $p = 1$.

In addition, although an entity now has different representations when acting as a head or a tail, we expect the difference is under a certain threshold. And following the disentangle characteristic of representation learning (Bengio et al. 2013), l_1-norm $\|f_r^{(1)}(\mathbf{e}) - f_r^{(2)}(\mathbf{e})\|_1$ is employed as a regularization term.

Summing up, the training objective is to minimize the following loss function, in which the first term is a margin-based ranking loss:

$$\sum_{(h,r,t)} \sum_{(h',r,t')} \left[\gamma + s(h, r, t) - s(h', s, t')\right]_+ + \lambda \sum_{e\in\{h,h',t,t'\}} \|f_r^{(1)}(\mathbf{e}) - f_r^{(2)}(\mathbf{e})\|_1 \tag{7}$$

in which, $[\cdot]_+$ denotes the positive part, $\gamma > 0$ is a margin separating positive and negative triplets, and λ controls the trade-off between the two terms. (h', r, t') is sampled by the randomly corrupting h or t in the positive triplet (h, r, t), but not both of them in the same time.

3.1.1 Applied to TransE

Apply our idea to TransE, we get TransE-pa model. The model maintains two embeddings for each entity in head and tail position respectively. The residual of a triplet is:

$$\delta(h, r, t) = \mathbf{h}^{(1)} + \mathbf{r} - \mathbf{t}^{(2)} \tag{8}$$

Then, the score of a triplet is:

$$s(h, r, t) = \|\text{SelfAttention}(\delta(h, r, t))\|_p \tag{9}$$

And the loss function is:

$$\sum_{(h,r,t)} \sum_{(h',r,t')} [\gamma + s(h, r, t) - s(h', r, t')]_+ + \lambda \sum_{e\in\{h,h',t,t'\}} \|\mathbf{e}^{(1)} - \mathbf{e}^{(2)}\|_1 \tag{10}$$

3.1.2 Applied to TransR

In TransR-pa, the position information is no longer encoded by different entity representations. Instead, we use two relation-specific matrices, $\mathbf{M}_r^h$ to project the head entity into the relation-specific space, and $\mathbf{M}_r^t$ to project the tail. Then, the residual of a triplet is:

$$\delta(h, r, t) = \mathbf{M}_r^h\mathbf{h} + \mathbf{r} - \mathbf{M}_r^t\mathbf{t} \tag{11}$$

And the regularization term $\|f_r^{(1)}(\mathbf{e}) - f_r^{(2)}(\mathbf{e})\|_1 = \|\mathbf{M}_r^h\mathbf{e} - \mathbf{M}_r^t\mathbf{e}\|_1 \leq \|\mathbf{M}_r^h - \mathbf{M}_r^t\|_1\|\mathbf{e}\|_1$. So, for simplicity, we turn to restrict the term $\|\mathbf{M}_r^h - \mathbf{M}_r^t\|_1$. Now the loss function becomes:

$$\sum_{(h,r,t)}\sum_{(h',r,t')}[\gamma + s(h,r,t) - s(h',r,t')]_+ + \lambda\|\mathbf{M}_r^h - \mathbf{M}_r^t\|_1 \tag{12}$$

3.1.3 Applied to TransH

In TransH-pa, we assign two hyper-planes for each relation, which are determined by normal vectors $\mathbf{w}_h$ and $\mathbf{w}_t$ respectively. The former one is used to project the head entity $\mathbf{h}_\perp = \mathbf{h} - \mathbf{w}_h^\top\mathbf{h}\mathbf{w}_h$, while the latter one is used to project the tail entity $\mathbf{t}_\perp = \mathbf{t} - \mathbf{w}_t^\top\mathbf{t}\mathbf{w}_t$. Then, the residual of a triplet is:

$$\delta(h,r,t) = \mathbf{h}_\perp + \mathbf{r} - \mathbf{t}_\perp \tag{13}$$

As for the regularization term, We also derive its surrogate. Denote $\mathbf{w} = (\mathbf{w}_h + \mathbf{w}_t)/2$, $\mathbf{d} = (\mathbf{w}_h - \mathbf{w}_t)/2$, we have

$$\begin{aligned}\|f_r^{(1)}(\mathbf{e}) - f_r^{(2)}(\mathbf{e})\|_1 &= \|\mathbf{w}_h^\top\mathbf{e}\mathbf{w}_h - \mathbf{w}_t^\top\mathbf{e}\mathbf{w}_t\|_1 \\ &= 2\|\mathbf{d}^\top\mathbf{e}\mathbf{w} + \mathbf{w}^\top\mathbf{e}\mathbf{d}\|_1 \\ &\leq 2|\mathbf{d}^\top\mathbf{e}|\|\mathbf{w}\|_1 + 2|\mathbf{w}^\top\mathbf{e}|\|\mathbf{d}\|_1\end{aligned} \tag{14}$$

in which $\mathbf{d}^\top\mathbf{e}, \mathbf{w}^\top\mathbf{e}$ are scalars. As our model has restrict the l_2-norm of $\mathbf{e}, \mathbf{w}_h^r, \mathbf{w}_t^r$ (see Section 3.3 for detail), we turn to restrict the term $\|\mathbf{d}\|_1$, i.e., $\|\mathbf{w}_r^h - \mathbf{w}_r^t\|_1$.

The loss function becomes:

$$\sum_{(h,r,t)}\sum_{(h',r,t')}[\gamma + s(h,r,t) - s(h',r,t')]_+ + \lambda\|\mathbf{w}_r^h - \mathbf{w}_r^t\|_1 \tag{15}$$

3.1.4 Applied to TransD

In TransD-pa, the positional encoding parameters are column vectors: (i) $\mathbf{w}_h$, together with the projection vector of entity $\mathbf{w}_e$, determines the projection matrix of head entity, i.e., $\mathbf{M}_h = \mathbf{w}_h\cdot\mathbf{w}_e^\top + \mathbf{I}$; (ii) $\mathbf{w}_t$, together with the projection vector of entity $\mathbf{w}_e$, determines the projection matrix of head entity, i.e., $\mathbf{M}_t = \mathbf{w}_t\cdot\mathbf{w}_e^\top + \mathbf{I}$. Then, the residual of a triplet is:

$$\delta(h,r,t) = \mathbf{h}\mathbf{M}_h + \mathbf{r} - \mathbf{t}\mathbf{M}_t \tag{16}$$

And the regularization term $\|f_r^{(1)}(\mathbf{e}) - f_r^{(2)}(\mathbf{e})\|_1 = \|(\mathbf{w}_h - \mathbf{w}_t)\mathbf{w}_e^\top\mathbf{e}\|_1 = \|\mathbf{w}_h - \mathbf{w}_t\|_1|\mathbf{w}_e^\top\mathbf{e}|$ ($\mathbf{w}_e^\top\mathbf{e}$ is a scalar), so it is equivalent to use $\|\mathbf{w}_h - \mathbf{w}_t\|_1$ as the penalization. The loss function becomes:

$$\sum_{(h,r,t)}\sum_{(h',r,t')}[\gamma + s(h,r,t) - s(h',r,t')]_+ + \lambda\|\mathbf{w}_h - \mathbf{w}_t\|_1 \tag{17}$$

3.2 Model complexity

Table 1 lists the complexity of the baselines and our proposed framework. We choose to use a self-attention block with 4 heads (Section 4.3.2 for detail), and hence cost $12n^2$ extra parameters. Given that $n_e \gg n$ (usually, n_e is in ten thousand level, while n is chosen to be in hundred level), our framework will not significantly increase the model and time complexity, so can be applied on large-scale knowledge graphs.

We have also listed the complexity of state-of-the-art models, including HolE (Nickel et al. 2016) and ComplEx (Théo et al. 2016). Our model shows similar time and space

Table 1 Comparison of model and time complexity: the number of parameters and the number of multiplication operations in an epoch of several embedding models

Model	#Parameters	#Operations
TransE	$O(N_e m + N_r n)$ $(m = n)$	$O(nN_t)$
TransE-pa	$O(2N_e m + N_r n + 12n^2)$ $(m = n)$	$O(4nN_t)$
TransR	$O(N_e m + N_r(m+1)n)$	$O(2mnN_t)$
TransR-pa	$O(N_e m + N_r(2m+1)n + 12n^2)$	$O(2mnN_t + 3mN_t)$
TransH	$O(N_e m + 2N_r n)$ $(m = n)$	$O(2nN_t)$
TransH-pa	$O(N_e m + 3N_r n + 12n^2)$ $(m = n)$	$O(5nN_t)$
TransD	$O(2N_e m + 2N_r n)$	$O(2nN_t)$
TransD-pa	$O(2N_e m + 3N_r n + 12n^2)$ $(m = n)$	$O(5nN_t)$
HolE	$O(N_e m + N_r n)$ $(m = n)$	$O(n \log n)$
ComplEx	$O(2N_e m + 2N_r n)$ $(m = n)$	$O(4nN_t)$

N_e and N_r represent the number of entities and relations, respectively. N_t represents the number of triplets in a knowledge graph. m is the dimension of entity embedding space, and n is that of relation embedding space. In some models, $m = n$. Our model spends $12n^2$ parameters for a self-attention block with 4 heads

complexity compared with ComplEx, and HolE needs more computational cost. In KG2E model (He et al. 2015), it is necessary to calculate the determinant of matrices, resulting a huge computational cost. As for ConvE (Dettmers et al. 2018), it hires a deep neural network architecture, while our proposed model is shallow. As a conclusion, our proposed framework shows advantages in time and space complexity.

3.3 Training, initilization & constraints

The learning process of TransX-pa is carried out using Stochastic Gradient Descent (SGD). To accelerate training process, we initialize the entity and relation embeddings with results of TransE. Besides, projection matrices in TransR and TransR-pa are initialized as identity matrices, following what (Lin et al. 2015) did. Other parameters are initialized using Xavier's method (Glorot and Bengio 2010).

Last but not least, the translational distance models usually restrict the l_2-norm of the entity embeddings both before and after transformation. However, their concrete forms vary slightly, please see the Table 1 in Wang's review (2017) for detail. What's more, some of the them converted the l_2-norm constraints as a regularization term during optimization (Wang et al. 2014), but some unitize them explicitly (Lin et al. 2015; Ji et al. 2015).

In our framework, we unify them into two constraints: 1. constraints on the entities before transformation, $\|\mathbf{h}\|_2 \leq 1$, $\|\mathbf{t}\|_2 \leq 1$; 2. constraints on the entities after transformation: $\|f_r^{(1)}(\cdot)\|_2 \leq 1$, $\|f_r^{(2)}(\cdot)\|_2 \leq 1$. Note that we have removed any constraints on $\mathbf{r}$. As a remark, a norm vector of hyper-plane just indicates a direction, so in TransH and TransH-pa, the norm vectors are naturally to have unit length. For all restriction, we adopt the explicit way, i.e., letting $\mathbf{x} = \mathbf{x}/\|\mathbf{x}\|_2$.

4 Experimental study

In this section, we first describe the data sets. Then, we discuss how to determine the hyper-parameters. Finally, we evaluate our model on triplet classification and link prediction tasks,

for each task, we introduce the evaluation settings, present the experimental results and analysis them.

Worth mentioning, Akrami's work (Akrami et al. 2018) has shown a big difference among the results reported in the original papers and the open source tool-kits, even if the code is released by the authors (ANALOGY[1] (Liu et al. 2017), ComplEx[2] (Théo et al. 2016), ConvE[3] (Dettmers et al. 2018) and OpenKE[4] (Han et al. 2018)). In the following, this paper reports the results both in the original paper and our re-implementation.

Our code is based on the OpenKE. As a remark, OpenKE has two bugs. First, it neglects all l_2-norm constraints. Second, it makes error when training with several negative triplets per golden triplet.

4.1 Data sets

We test our framework on two tasks, triplet classification and link prediction, with several widely used knowledge graphs, which are extracted from WordNet (Miller 2005) and Freebase (Bollacker et al. 2008). WordNet is a large lexical knowledge graph, with entities expressing distinct concepts and relations expressing conceptual-semantic and lexical relations (like *type_of*). Freebase contains a large number of facts, such as (*barack_obama_sr, nationality, Kenya*).

For the task of link prediction, we use four datasets FB15k, FB15k-237, WN18 and WN18RR as the previous work does:

- FB15k and FB15k-237: FB15k, used by TransE (Bordes et al. 2013) and the following researchers, is a subset of Freebase. Unfortunately, 81% of the test triplets can be directly inferred by inversion (Toutanova et al. 2015), resulting in a test leakage. Therefore, FB15k-237 has been proposed with the inverse relations removed. Since the average number of linked triples for each entity is far smaller than that of FB15k (see Table 2), it is more challenging to predict unobserved fact on FB15k-237.
- WN18 and WN18RR : WN18, also adopted to be a standard benchmark (Bordes et al. 2013), is a subset of WordNet. WN18 also contains many inverse relations, and hence, a more challenging dataset, WN18RR, has been proposed for further study (Dettmers et al. 2018).

For the task of triplet classification, we use FB13 and WN11 (Socher et al. 2013), as well as the four datasets aforementioned. The former two have already been released with negative triplets,[5] in order to ensure a fair comparison. As for the other four datasets which do not contain negative triplets in its test set, we adopt the same settings following Socher's work (Socher et al. 2013) to generate negative triplets. Note that FB13 and WN11 contains plenty of triplets but few relations, so there is no need to test link prediction task on them.

The statistics of the extracted knowledge graphs are summarized into Table 2.

[1] https://github.com/quark0/ANALOGY
[2] https://github.com/ttrouill/complex
[3] https://github.com/TimDettmers/ConvE
[4] https://github.com/thunlp/OpenKE
[5] https://www.cs.princeton.edu/~danqic/data/nips13-dataset.tar.bz2

Table 2 Datasets used in the experiments: Avg. #Train/#Ent represents the average number of linked triplets regarding with each entity in the training set. With a small average linked triplets, it it is more challenging to learn the semantic of the entities

Dataset	#Rel	#Ent	#Train	#Valid	#Test	Avg. #Train/#Ent
FB15k	1,345	14,951	483,142	50,000	59,071	32.315
FB15k-237	237	14,541	272,115	17,535	20,466	18.714
FB13	13	75,043	316,232	5,908	23,733	4.214
WN18	18	40,943	141,442	5,000	5,000	3.455
WN18RR	11	40,943	86,835	3,034	3,134	2.121
WN11	11	38,696	112,581	2,609	10,544	2.909

4.2 Implementation

We train TransE for each dataset, using a grid search of hyper-parameters: the dimension of embeddings $n = m \in \{50, 100, 200\}$, margin $\gamma \in \{1, 2, 3, 4, 5\}$, and SGD learning rate $\in \{1e^{-4}, 5e^{-4}, 1e^{-3}, 5e^{-3}\}$, training epochs $\in \{1000, 2000, 3000, 4000, 5000\}$. The value corresponding to highest Hits@10 score will then be applied to all models, in order to ensure the fairness of comparison.

For FB15k, FB13 and WN11, $m = n = 100, \gamma = 1$, with SGD learning rate $1e^{-3}$ and training epochs of 1000; for FB15k-237, $m = n = 100, \gamma = 2$, with SGD learning rate $5e^{-4}$ and training epochs of 3000; for WN18, $m = n = 50, \gamma = 3$, with SGD learning rate $1e^{-3}$ and training epochs of 1000, for WN18RR, $m = n = 50, \gamma = 3$, with SGD learning rate $5e^{-4}$ and training epochs of 1000. Besides, the self-attention block uses a 4-head attention (we will discuss our choice for this hyper-parameter later in Section 4.3.2). And we set $\lambda = 0.001$ to control the trade-off in the loss function.

During training, SGD algorithm is used, with the number of batches is 100. For triplet classification task, we randomly generates only one negative triplet for each positive triplet, in order to build an unbiased classifier. As for link prediction task, in order to provide more guidance for the gradients, we randomly generates 10 negative triplets for each positive triplet.

As a remark, generating negative triplets involves two settings as follow:

unif: In TransE (Bordes et al. 2013), a pair of entities (h', t') is randomly sampled from all the entities, which introduces too many false negative labels during training.

bern: To address this problem, following Wang's opinion (Wang et al. 2014), works thereafter adopt a Bernoulli distribution to sample negative triplets. Specifically speaking, denote the average number of tail entities per head entity as tph, and the average number of head entities per tail entity as hpt, then with probability $\frac{tph}{tph+hpt}$, h is replaced by h', and with probability $\frac{hpt}{tph+hpt}$, t is replaced by t'.

4.3 Results and discussion

4.3.1 Triplet classification

Given a triplet (h, r, t), the task is to judge whether it is correct or not, so is a binary classification problem. In order to use the score $s(h, r, t)$ for triplet classification, we set a

relation-specific threshold δ_r. If $s(h, r, t) > \delta_r$, it will be classified as positive, and vice versa. δ_r is obtained by maximizing the accuracies on the validation set.

We compare our methods with the translational distance models on all datasets. Note that the recent models (HolE (Nickel et al. 2016), ComplEx (Théo et al. 2016), ConvE (Dettmers et al. 2018)) do not test on this task, so here we do not compare with them.

Table 3 shows the accuracy for triplet classification. Our re-implementation is nearly the same compared with results reported in the correspoinding original papers (see the accuracy in the parentheses). According to the table, we can conclude that our models outperform the corresponding existing models (TransE (Bordes et al. 2013), TransR (Lin et al. 2015), TransH (Wang et al. 2014), TransD (Ji et al. 2015)), and TransD-pa achieves the highest accuracy on all datasets.

To evaluate the significance of the results, we apply Wilcoxon's test (Demsar 2006) to compare differences statistically via pairwise comparisons. The Wilcoxon signed-rank test is applied to calculate the p-values and asymptotic p-values (p^*) corresponding to different pairs of comparisons for all datasets are obtained. Additionally, for each comparison, the sum of the ranks in favor of the first algorithm (R^+) and the sum of the ranks in favor of the second algorithm (R^-) are provided. In this paper, we consider a difference to be significant at $p < 0.05$. The results in Table 4 show that TransX-pa always performs better than TransX, they are significantly different for both settings.

Besides, we have two more interesting findings.

Comparison under different negative sampling settings For triplets classification task, TransX-pa performs similarly under the 'unif ' and 'bern' sampling settings, especially

Table 3 Accuracy of TransX-pa for triplet classification

		WN11	FB13	FB15k	FB15k-237	WN18	WN18RR
TransE	TransE (unif)	77.89 (75.9)	72.98 (70.9)	84.71 (77.3)	78.93	95.49	85.33
	TransE-pa (unif)	82.14	83.35	87.68	80.04	96.79	85.69
	TransE (bern)	77.33 (75.9)	79.83 (81.5)	85.97 (79.8)	80.28	95.65	85.06
	TransE-pa (bern)	82.01	83.96	88.61	81.82	96.80	85.70
TransR	TransR (unif)	84.64 (85.5)	78.63 (74.7)	85.51 (81.7)	79.92	95.81	83.72
	TransR-pa (unif)	85.89	81.94	87.95	80.74	96.43	85.46
	TransR (bern)	84.39 (85.9)	80.17 (82.5)	85.88 (83.9)	81.03	95.77	84.06
	TransR-pa (bern)	*86.90*	83.01	88.77	81.82	96.80	85.68
TransH	TransH (unif)	80.82 (77.7)	80.75 (76.5)	86.00 (80.2)	80.33	94.99	84.97
	TransH-pa (unif)	84.39	88.21	89.74	83.19	**96.93**	85.97
	TransH (bern)	81.03 (78.8)	84.52 (83.3)	87.78 (87.7)	80.50	95.71	85.11
	TransH-pa (bern)	84.98	88.58	*89.95*	**83.64**	*96.91*	85.27
TransD	TransD (unif)	86.12 (85.6)	86.01 (85.9)	87.36 (86.4)	80.04	95.88	85.37
	TransD-pa (unif)	**86.93**	**89.63**	89.82	83.41	96.81	**86.57**
	TransD (bern)	86.31 (86.4)	87.78 (89.1)	88.16 (88.0)	81.43	95.78	85.55
	TransD-pa (bern)	86.76	*89.51*	**90.27**	*83.54*	96.80	*86.24*

[a] The best results are marked in bold, and the second best results in italic

[b] Results in the parentheses are from the corresponding original papers

Table 4 Wilcoxon's Test after comparing TransX and TransX-pa

	Unif				Bern			
	R^+	R^-	p	p^*	R^+	R^-	p	p^*
TransE-pa / TransE	21.0	0.0	0.03126	0.021098	21.0	0.0	0.03126	0.021098
TransR-pa / TransR	21.0	0.0	0.03126	0.021098	21.0	0.0	0.03126	0.01787
TransH-pa / TransH	21.0	0.0	0.03126	0.021098	21.0	0.0	0.03126	0.021098
TransD-pa / TransD	21.0	0.0	0.03126	0.021098	21.0	0.0	0.03126	0.01787

on WN11 and FB13 datasets, which is different to the findings in the link prediction task (see Section 4.3.2 for detail). The reason is that the validation and testing set of the three datasets used here are balanced, i.e., the number of positive triplets is equal to that of negative triplets. And the number of relations of FB13 and WN11 datasets are only 13 and 11 respectively, so both negative sampling settings will not introduce so many 'false negative' triplets. However, on FB15k dataset, now that the number of relations is much bigger, TransX-pa performs a litter better under the 'bern' sampling settings.

Closer look to TransD and TransD-pa The difference of TransD-pa and TransD is not so obvious on WN11 dataset (only an increment of 0.94% (unif), 0.52% (bern)), compared to the differences on FB13 (an increment of 4.21% (unif), 1.97% (bern)) and FB15k (an increment of 2.82% (unif), 2.39% (bern)). This phenomenon indicates there exists a learning bottleneck for translational distance models on WN11 dataset. In fact, there are 470 entities appearing in the validation and testing sets but not appearing in the training set of WN11. And the triplets that contains these 'unseen' entities accounts 6.4% in the validation and testing sets. Some previous work, like (Wang et al. 2014), introduced textural resources to address this 'out-of-kb' problem. So, though little improvements compared with TransD on WN11 dataset, we can still declare that TransX-pa outperforms the existing translational distance models.

4.3.2 Link prediction

Link prediction is a rank problem. In testing phase, for each test triple, we replace the head or tail entity by all entities in the knowledge graph, then rank them in descending order according to their scores. There are three acknowledged metrics:

- Mean Reciprocal Rank of correct entities (MRR): the reciprocal rank of a query response is the multiplicative inverse of the rank of the first correct answer, then MRR is the mean of it over the test set, saying that MRR $= \frac{1}{N}\sum_{i=1}^{N}\frac{1}{rank_i}$;
- Mean Rank of correct entities (MR): MR $= \frac{1}{N}\sum_{i=1}^{N} rank_i$, MR is sensitive to a few bad results.
- Proportion of correct entities in top-n ranked entities (Hits@n). We show the results of $n = 1, 3, 10$.

A good link predictor should achieve higher MRR, lower MR or higher Hits@n.

In fact, a corrupted triplet may exist in knowledge graph, and of course is correct. The evaluation above may under-estimate the models that rank these corrupted but correct triples high. To enable a fair evaluation, a 'Filter' evaluation setting filters out these triples are filtered out. Accordingly, the original setting is called 'Raw'.

Table 5 Comparison with the translational distance models on the FB15k dataset

		MRR ↑		MR ↓		Hits@10 ↑		Hits@3 ↑		Hits@1 ↑	
		Raw	Filter	Raw	Filter	Raw	Filter	Raw	Filter	Raw	Filter
TransE	TransE (unif)	.207	.394	237 (243)	83 (125)	47.6 (34.9)	70.6 (47.1)	25.0	55.7	11.9	32.5
	TransE-pa (unif)	.218	.473	191	75	47.3	74.2	24.9	58.9	12.4	35.8
	TransE (bern)	.220	.457	236	101	49.2	71.1	27.3	56.1	12.1	33.8
	TransE-pa (bern)	.250	.499	**178**	77	51.3	72.9	28.2	59.1	13.7	36.7
TransR	TransR (unif)	.202	.394	228.5 (226)	77.7 (78)	45.0 (43.8)	67.3 (65.5)	23.0	52.3	13.9	33.9
	TransR-pa (unif)	.241	.435	199.4	74.4	48.5	74.5	22.9	58.2	15.3	35.9
	TransR (bern)	.250	.450	206.5 (198)	78.9 (77)	49.6 (48.2)	69.8 (68.7)	28.6	53.6	12.5	32.1
	TransR-pa (bern)	.241	.468	194.0	75.6	51.1	73.8	28.7	63.8	**16.0**	40.4
TransH	TransH (unif)	.242	.519	230 (211)	75 (84)	48.6 (42.5)	75.9 (58.5)	26.9	60.6	13.2	38.6
	TransH-pa (unif)	.255	.548	191	70	50.7	79.7	28.5	66.5	14.2	41.2
	TransH (bern)	.254	.530	217 (212)	92 (87)	50.7 (45.7)	75.9 (64.4)	29.1	62.6	13.7	39.5
	TransH-pa (bern)	*.270*	*.589*	197.3	72	*52.0*	80.3	*30.4*	*67.8*	15.4	*46.2*
TransD	TransD (unif)	.226	.511	203 (211)	72 (67)	49.3 (49.4)	76.8 (74.2)	26.6	62.2	12.1	35.9
	TransD-pa (unif)	.259	.583	*188*	**63**	51.7	**81.1**	29.4	67.5	*15.6*	45.3
	TransD (bern)	.257	.539	209 (194)	95 (91)	51.1 (53.4)	76.2 (77.3)	29.4	63.3	14.0	40.6
	TransD-pa (bern)	**.276**	**.591**	193	*67*	**53.0**	*80.7*	**31.3**	**68.1**	**16.0**	**46.5**

[a] The best results are marked in bold, and the second best results in italic

[b] Results in the parentheses are from the corresponding original papers

[c] '↑' means higher is better, while '↓' means lower is better

Table 6 Performance on the WN18 dataset

		MRR ↑		MR ↓		Hits@10 ↑		Hits@3 ↑		Hits@1 ↑	
		Raw	Filter	Raw	Filter	Raw	Filter	Raw	Filter	Raw	Filter
TransE	TransE(unif)	.423	.591	334 (263)	319 (251)	79.7 (75.4)	93.5 (89.2)	57.7	87.0	5.3	10.3
	TransE-pa(unif)	.457	.625	329	313	80.2	**94.2**	59.9	91.1	26.2	35.1
	TransE(bern)	.448	.589	329	315	77.6	93.7	57.8	88.7	5.3	11.0
	TransE-pa(bern)	.454	.611	321	306	79.9	93.8	58.9	90.5	26.3	34.2
TransR	TransR(unif)	.438	.601	328 (232)	312 (219)	79.6 (78.3)	93.3 (91.7)	58.4	88.8	8.9	22.0
	TransR-pa(unif)	.460	.627	315	299	80.3	94.0	**61.0**	*91.6*	26.2	34.8
	TransR(bern)	.434	.617	319 (238)	304 (225)	79.7 (79.8)	93.4 (92)	58.7	87.6	10.6	21.5
	TransR-pa(bern)	*.465*	.629	311	**286**	80.2	*94.1*	**61.0**	91.3	**27.0**	*35.6*
TransH	TransH(unif)	.447	.617	321 (318)	305 (303)	79.7 (75.4)	93.5 (86.7)	58.4	88.3	8.5	18.2
	TransH-pa(unif)	.460	.630	*309*	298	80.0	94.0	**61.0**	**92.1**	26.1	34.6
	TransH(bern)	.433	.601	316 (401)	300 (388)	78.8 (73)	92.5 (82.3)	57.0	84.2	7.1	17.8
	TransH-pa(bern)	.454	.632	**302**	*287*	*80.2*	*94.1*	59.4	90.0	25.8	33.5
TransD	TransD(unif)	.447	.623	328 (242)	312 (229)	79.6 (79.2)	93.5 (92.5)	57.8	87.9	10.5	21.3
	TransD-pa(unif)	.459	*.667*	325	312	**80.4**	93.9	60.2	90.9	*26.5*	35.4
	TransD(bern)	.463	.617	321 (224)	306 (212)	78.6 (79.6)	92.4 (92.2)	56.7	84.8	11.1	24.1
	TransD-pa(bern)	**.480**	**.675**	310	295	**80.4**	**94.2**	*60.6*	90.8	**27.0**	**37.4**

[a] The best results are marked in bold, and the second best results in italic

[b] Results in the parentheses are from the corresponding original papers

[c] '↑' means higher is better, while '↓' means lower is better

Table 7 Performance on the FB15k-237 dataset

Model		MRR ↑		MR ↓		Hits@10 ↑		Hits@3 ↑		Hits@1 ↑	
		Raw	Filter	Raw	Filter	Raw	Filter	Raw	Filter	Raw	Filter
TransE	TransE(unif)	.161	.287	**436**	**237**	30.9	47.2	16.7	31.7	9.1	19.5
	TransE-pa(unif)	.170	.324	*445*	*243*	31.8	52.2	17.3	35.4	9.8	23.1
	TransE(bern)	.179	.305 (.294)	451	311 (347)	32.3	48.4 (46.5)	16.3	31.8	9.0	19.8
	TransE-pa(bern)	*.185*	*.337*	459	323	**33.2**	*52.2*	**19.2**	36.9	**11.4**	24.6
TransR	TransR(unif)	.163	.285	458	253	29.8	46.5	16.2	31.0	8.4	19.0
	TransR-pa(unif)	.173	.314	466	258	30.9	51.7	18.5	36.0	10.9	23.1
	TransR(bern)	.172	.299	467	312	30.6	48.8	17.6	32.3	10.0	20.1
	TransR-pa(bern)	.180	.322	454	309	32.1	51.9	18.7	36.9	10.9	23.8
TransH	TransH(unif)	.156	.286	474	268	30.0	47.3	16.0	31.8	8.9	19.3
	TransH-pa(unif)	.165	.323	478	272	31.2	51.4	18.9	*37.4*	11.3	24.8
	TransH(bern)	.177	.309	455	310	31.8	48.9	18.3	34.1	11.0	22.0
	TransH-pa(bern)	.183	**.343**	453	308	*32.8*	**53.0**	*19.0*	**37.5**	**11.4**	*25.0*
TransD	TransD(unif)	.157	.288	449	*243*	30.2	47.5	16.0	31.8	8.9	19.5
	TransD-pa(unif)	.175	.323	451	245	31.2	51.4	16.7	35.5	10.8	24.7
	TransD(bern)	.171	.306	461	306	31.0	48.6	17.2	33.7	10.4	21.8
	TransD-pa(bern)	**.188**	**.343**	449	305	*32.8*	**53.0**	18.9	**37.5**	*11.3*	**25.1**

[a] The best results are marked in bold, and the second best results in italic

[b] Results of the TransE in the parentheses are from Nguyen's work (Nguyen et al. 2018)

[c] '↑' means higher is better, while '↓' means lower is better

Table 8 Performance on the WN18RR dataset

		MRR ↑		MR ↓		Hits@10 ↑		Hits@3 ↑		Hits@1 ↑	
		Raw	Filter	Raw	Filter	Raw	Filter	Raw	Filter	Raw	Filter
TransE	TransE(unif)	.153	.201	3794	3780	43.9	47.4	24.4	36.2	1.72	3.10
	TransE-pa(unif)	.191	.252	*2999*	2984	49.6	52.3	29.4	43.7	7.59	10.1
	TransE(bern)	.167	.217 (.226)	3664	3650 (3384)	46.3	49.8 (50.1)	26.1	38.8	3.38	4.98
	TransE-pa(bern)	**.199**	*.261*	**2970**	**2956**	**50.6**	*53.2*	30.2	44.8	**8.70**	*11.4*
TransR	TransR(unif)	.154	.207	4099	4084	44.5	46.6	25.1	38.1	1.47	2.52
	TransR-pa(unif)	.187	.247	3339	3325	48.8	51.0	29.8	43.6	6.29	8.55
	TransR(bern)	.165	.220	3955	3940	46.4	49.5	26.8	40.2	2.55	4.24
	TransR-pa(bern)	.194	.255	3250	3236	49.8	52.1	*30.6*	44.4	7.91	10.7
TransH	TransH(unif)	.149	.200	3752	3737	44.1	48.4	23.5	37.1	1.21	2.04
	TransH-pa(unif)	.185	.243	2983	*2959*	48.6	52.1	29.0	43.4	6.00	8.04
	TransH(bern)	.168	.219	3622	3598	47.1	49.8	23.9	37.8	3.25	4.95
	TransH-pa(bern)	.193	.254	3006	2971	*50.0*	52.5	29.9	*45.2*	7.47	10.1
TransD	TransD(unif)	.149	.200	3320	3206	43.8	48.8	24.6	37.9	2.57	4.53
	TransD-pa(unif)	.178	.245	3033	3009	47.0	50.9	28.4	43.0	5.62	7.48
	TransD(bern)	.165	.211	3531	3498	47.1	50.5	26.8	39.5	5.42	8.67
	TransD-pa(bern)	*.196*	**.267**	3114	3096	*50.0*	**54.3**	**31.3**	**46.4**	*8.33*	**12.9**

[a] The best results are marked in bold, and the second best results in italic

[b] Results of the TransE in the parentheses are from Nguyen's work (2018)

[c] '↑' means higher is better, while '↓' means lower is better

We first compare TransX-pa to TransE (Bordes et al. 2013), TransR (Lin et al. 2015), TransH (Wang et al. 2014), and TransD (Ji et al. 2015). Tables 5, 6, 7 and 8 summary the self-comparison results. Because the corresponding original papers only report MR and Hits@10 for FB15k and WN18 (see the result in the parentheses in Tables 5 and 6), we re-implement and statistic the performances of all metrics. As for FB15k-237 and WN18RR, Nguyen (2018) has reported the filtered MR, MRR and Hits@10 of TransE under 'bern' negative sample setting, which are collected as a reference of our re-implementation (see results in the parentheses in Tables 7 and 8). According to the tables, it can be seen that our re-implementation is comparable to, or even much better than (see Hits@10 of TransE, TransH on FB15k and WN18), the original results, indicating the fairness of the comparison.

From these tables, we can conclude that: (1) TransX-pa performs better when adapting to different models, under different negative sample settings and different metrics; (2) under most circumstances, the 'bern' negative sampling setting is better for link prediction task than the 'unif' setting; (3) TransD-pa (bern) ranks the first under most evaluation metrics, and in the rest metrics, TransD-pa (bern) also ranks the top. Hence, we choose TransD-pa (bern) to be the best models and compare it with state-of-the-art methods.

Again, in order to demonstrate the effectiveness of our improvement, we apply Wilcoxon's test (Demsar 2006) via pairwise comparisons (Table 9). Now that there are only four datasets, to ensure a suitable test, we view a dataset under 'unif' and 'bern' setting to be two datasets. For simplicity, we choose filtered MRR and Hits@10 for Wilcoxon's test.

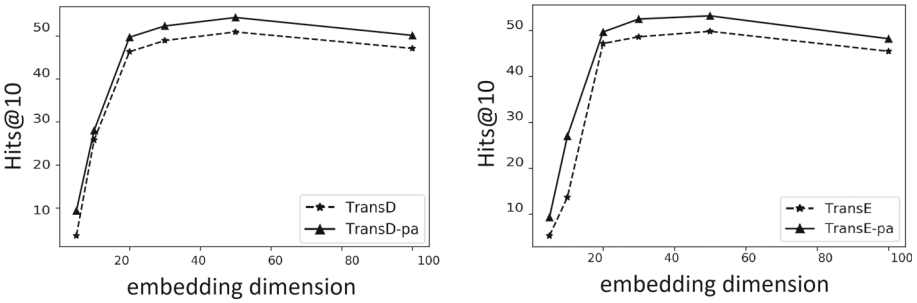
With a significant level at 0.05, we find out that except for TransH, TransX-pa always performs better than TransX. We suggest that it is because that TransH has extra constraints over the norm vectors of hyper-planes (see Section 3.1.3), which causes the optimization to be more difficult.

To further illustrate our improvements, we select two pairs, TransE v.s. TransE-pa (see Fig. 2a), TransD v.s. TransD-pa (see Fig. 2b), to see what the performance will be if we vary the embedding dimension in the range {5, 10, 20, 30, 50, 100}. According to the figures, our proposed methods show higher performance for all dimensions, indicating the robustness of our framework.

Also, to investigate the effect of the number of attention heads, we retrain the TransD and TransD-pa from scratch. That is to say, we do not use the pre-trained TransE model to initialize them. We find out that the number of attention heads not only affect the performance but also affect the convergence rate. According to Fig. 3a, TransD-pa with 4-head attention block shows highest Hits@10, while TransD-pa with 8-head attention block is hard to train, obtaining a little worse performance. Taking a closer look to the early training epochs (see

Table 9 Wilcoxon's Test after comparing MRR and Hits@10 of TransX and TransX-pa

	Filter MRR				Filter Hits@10			
	R^+	R^-	p	p^*	R^+	R^-	p	p^*
TransE-pa / TransE	35.0	1.0	0.015626	0.012626	36.0	0.0	0.007812	0.009583
TransR-pa / TransR	36.0	0.0	0.007812	0.009583	36.0	0.0	0.007812	0.009583
TransH-pa / TransH	28.0	8.0	0.19532	0.141482	30.0	6.0	0.10938	0.080058
TransD-pa / TransD	36.0	0.0	0.007812	0.009583	36.0	0.0	0.007812	0.009583



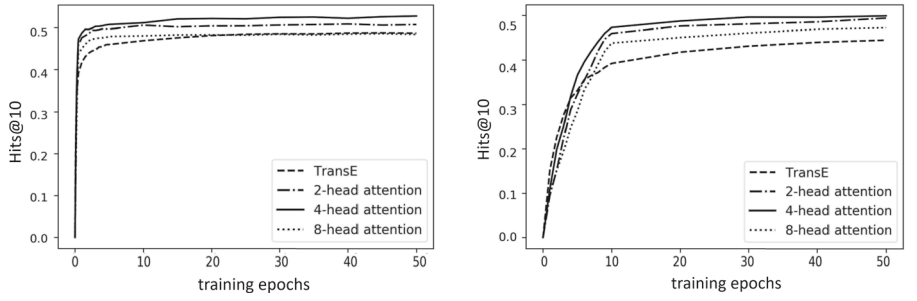
(a) TransE v.s. TransE-pa on WN18RR (b) TransD v.s. TransD-pa on WN18RR

Fig. 2 Translational distance models with different embedding dimensions on WN18RR dataset. As the embedding dimension grows, Hits@10 increases quickly, and tends to over-fit when the dimension is larger than 50, so we choose $m = n = 50$ for WN18RR

Fig. 3b), TransE shows the fastest convergences rate, following by TransD-pa with 4-head attention block. It suggests that the self-attention block with too few or too much heads is unsatisfactory.

Then, we compare TransX-pa to the state-or-the-art models, including KG2E (He et al. 2015), HolE (Nickel et al. 2016), ComplEx (Théo et al. 2016) and ConvE (Dettmers et al. 2018). Tables 10 and 11 summary the comparison with state-of-the-art methods. We choose KG2E (He et al. 2015), HolE (Nickel et al. 2016), ComplEx (Théo et al. 2016) and ConvE (Dettmers et al. 2018) to be the competitors, for they are proposed in recent years and show great improvements for link prediction. The corresponding original papers did not test the models on all the datasets with full metrics (the reported results are shown with parentheses in the tables), so here we re-implement them to enable a fare comparison.

From these tables, our proposed model shows a better, or at least comparable performance, than state-of-the-art models. Specifically, on FB15k-237 dataset, our model ranks top under MRR and Hits@10 metrics. And on WN18RR dataset, our model ranks top under MR and Hits@10 metrics. Given the advantage on running time compared to them (HolE and ComplEx require to be trained using SGD with AdaGrad (Duchi et al. 2011) for tuning



(a) TransD v.s. TransD-pa on WN18RR (b) A closer look of the early training

Fig. 3 TransD-pa with different number of attention heads v.s. TransD on WN18RR dataset. Figure 3b takes a closer look before 50th training epochs of Fig. 3a. We can see that, the convergence rates of TransE and TransE-pa with different attention heads are almost the same, but they convergence to different value of Hits@10, showing different learning capacity. As a result, we choose $k = 4$

Table 10 Comparison with the other models on the FB15k and WN18 datasets

	FB15k						WN18					
	MRR ↑		MR ↓		Hits@10 ↑		MRR ↑		MR ↓		Hits@10 ↑	
	Raw	Filter	Raw	Filter	Raw	Filter	Raw	Filter	Raw	Filter	Raw	Filter
TransD-pa	**.276**	.591	*193*	**67**	**53.0**	80.1	.480	.675	**310**	**295**	*80.4*	*94.2*
KG2E	.268	.539	242 (174)	129 (59)	52.3 (48.9)	77.4 (74.0)	.410	.510	395 (342)	379 (331)	79.4 (80.2)	90.9 (92.8)
HolE	.270	.468 (.524)	288	160	51.0	70.6 (73.9)	.605	.935 (.938)	320	307	**81.3**	94.0 (94.9)
ComplEx	.263	*.634* (.689)	**192**	89	52.0	*83.5* (85.1)	.580	*.936* (.941)	318	302	80.2	**94.3** (94.7)
ConvE	*.272*	**.679** (.745)	198	*71* (64)	*52.5*	**84.1** (87.3)	.590	**.940** (.942)	522	498 (504)	80.0	**94.3** (95.5)

[a] The best results are marked in bold, and the second best results in italic

[b] The state-of-the-art models only report partial metrics (shown in parentheses), so we also report our re-implementation

[c] '↑' means higher is better, while '↓' means lower is better

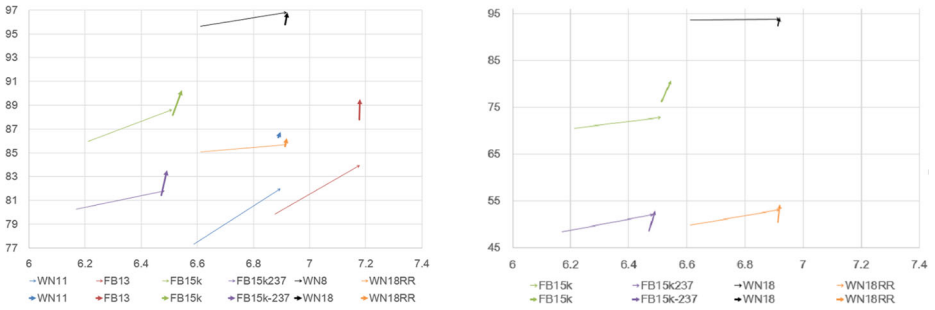
Table 11 Comparison with the other models on the FB15k-237 and WN18RR datasets

	FB15k-237						WN18RR					
	MRR ↑		MR ↓		Hits@10 ↑		MRR ↑		MR ↓		Hits@10 ↑	
	Raw	Filter	Raw	Filter	Raw	Filter	Raw	Filter	Raw	Filter	Raw	Filter
TransD-pa	**.188**	**.343**	**449**	*305*	**32.8**	**53.0**	.196	.267	**3114**	**3096**	**50.0**	**54.3**
KG2E	*.174*	.292	559	401	*30.9*	46.1	.193	.238	4410	4395	*46.0*	47.8
HolE	.138	.264	552	342	26.7	43.5	.233	.364	*4076*	*4061*	38.2	39.2
ComplEx	.143	.266 (.247)	525	387 (339)	29.1	46.1 (42.8)	*.266*	*.418* (.44)	5703	5688 (5261)	45.7	*50.4* (51.0)
ConvE	.154	*.310* (.316)	*489*	**277** (246)	28.4	*49.5* (49.1)	**.287**	**.448** (.46)	5342	5215 (5277)	45.0	48.7 (48.0)

[a] The best results are marked in bold, and the second best results in italic

[b] The state-of-the-art models only report partial metrics (shown in parentheses), so we also report our re-implementation

[c] '↑' means higher is better, while '↓' means lower is better



(a) Improvement on triplet classification (b) Improvement on link prediction

Fig. 4 Improved performance v.s. increased costs between two pairs of models: TransE and TransE-pa, TransD and TransD-pa. Arrows link from TransX to TransX-pa, with different colors represents for different datasets. For better visualization, we set the x-axis to be the logarithm of the total amount of parameters. Note that we compare the models both under the 'bern' setting

the learning rate), our proposed framework improves the translational distance models to be competitive in the field of large-scale knowledge graph learning.

4.3.3 Discussion

As a remark, we would like to justify that the increased cost in relation to the improvements. Since it is hard to perform quantitative assessment, we investigate it via comparison with the previous improvement of translational distance models, i.e., from TransE to TransD.

In Fig. 4a and b, we plot the performance with regard to the total number of parameters of the two pairs of models: TransE and TransE-pa (arrow from TransE to TransE-pa, in thin line), TransD and TransD-pa (arrow from TransD to TransD-pa, in heavy line), with different colors representing for different datasets. Note that the number of parameters of TransD and TransE-pa is nearly the same. According to the figure, from TransD to TransD-pa, a relatively higher performance improvement is obtained with a smaller increased cost, while from TransE to TransD or TransE-pa, the corresponding improvement will result in a higher increased cost.

5 Conclusion

In this paper, we propose a framework named TransX-pa. This framework takes the existing translational distance models for knowledge graph embedding (TransE, TransR, TransH and TransD) into consideration from a unified viewpoint. Under this framework, we have used positional-aware embeddings and self-attention blocks to deal with circle and hierarchical structures in knowledge graphs. Compared with previous models, TransX-pa only requires a small extra computational cost. A large number of experiments on triplet classification and link prediction tasks have shown the effectiveness of TransX-pa, demonstrating that it can be used to overcome the learning problem caused by the special structures.

Acknowledgments The authors are thankful for the financial support from the National Key Research and Development Program of China (2016QY03D0500), as well as the National Natural Science Foundation of China (U1636220, 61876183, 61976212).

References

Nickel, M., Murphy, K., Tresp, V., Gabrilovich, E. (2016). A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*, *104*, 11–33.

Wang, Q., Mao, Z., Wang, B., Guo, L. (2017). Knowledge graph embedding: a survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering*, *29*, 2724–2743.

Toutanova, K., & Chen, D. (2015). 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).

Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O. (2013). Translating embeddings for modeling multi-relational data. *Advances in Neural Information Processing Systems*, *26*, 2787–2795.

Wang, Z., Zhang, J., Feng, J., Chen, Z. (2014). Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the twenty-eighth AAAI conference on artificial intelligence* (pp. 1112–1119).

Lin, Y., Liu, Z., Zhu, X., Zhu, X., Zhu, X. (2015). Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the twenty-ninth AAAI conference on artificial intelligence* (pp. 2181–2187).

Ji, G., He, S., Xu, L., Liu, K., Zhao, J. (2015). Knowledge graph embedding via dynamic mapping matrix. In *Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing*, (Vol. 1: Long Papers pp. 687–696).

Ji, G., Liu, K., He, S., Zhao, J. (2016). Knowledge graph completion with adaptive sparse transfer matrix. In *Proceedings of the thirtieth AAAI conference on artificial intelligence* (pp. 985–991).

Fan, M., Zhou, Q., Chang, E., Zheng, T.F. (2014). Transition-based knowledge graph embedding with relational mapping properties. In *Proceedings of the 28th Pacific Asia Conference on Language, Information and Computing* (pp. 328–337).

Xiao, H., Huang, M., Zhu, X. (2016). From one point to a manifold: knowledge graph embedding for precise link prediction. In *Proceedings of the 25th international joint conference on artificial intelligence* (pp. 1315–1321).

He, S., Liu, K., Ji, G., Zhao, J. (2015). Learning to represent knowledge graphs with Gaussian embedding. In *Proceedings of the 24th ACM international on conference on information and knowledge management* (pp. 623–632).

Zhang, W. (2017). Knowledge graph embedding with diversity of structures. In *Proceedings of the 26th international conference on world wide web companion* (pp. 747–753).

Bordes, A., Weston, J., Collobert, R., Bengio, Y. (2011). Learning structured embeddings of knowledge bases. In *Proceedings of the twenty-fifth AAAI conference on artificial intelligence* (pp. 301–306).

Bordes, A., Glorot, X., Weston, J., Bengio, Y. (2014). A semantic matching energy function for learning with multi-relational data: application to word-sense disambiguation. *Machine Learning*, *94*(2), 233–259.

Yang, B., Yih, W.T., He, X., Gao, J., Deng, L. (2015). Embedding entities and relations for learning and inference in knowledge bases. In *Proceedings of the International Conference on Learning Representations*.

Nickel, M., Rosasco, L., Poggio, T. (2016). Holographic embeddings of knowledge graphs. In *Proceedings of the thirtieth AAAI conference on artificial intelligence* (pp. 1955–1961).

Théo, T., Johannes, W., Sebastian, R., Eric, G., Guillaume, B. (2016). Complex embeddings for simple link prediction. In *International Conference on Machine Learning* (pp. 2071–2080).

Hayashi, K., & Shimbo, M. (2017). On the equivalence of holographic and complex embeddings for link prediction. In *Proceedings of the 55th annual meeting of the association for computational linguistics* (pp. 554–559).

Dettmers, T., Minervini, P., Stenetorp, P., Ridedel, S. (2018). Convolutional 2D knowledge graph embeddings. In *Proceedings of the thirty-second AAAI conference on artificial intelligence*.

Lin, Y., Liu, Z., Luan, H., Sun, M., Rao, S., Liu, S. (2015). Modeling relation paths for representation learning of knowledge bases. In *Proceedings of the conference on empirical methods in natural language processing* (pp. 705–714).

Socher, R., Chen, D., Manning, C., Chen, D., Ng, A. (2013). Reasoning with neural tensor networks for knowledge base completion. In *Proceedings of the 26th international conference on neural information processing systems* (pp. 926–934).

Wang, Z., & Li, J. (2016). Text-enhanced representation learning for knowledge graph. In *Proceedings of the 25th international joint conferences on artificial intelligence* (pp. 1293–1299).

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I. (2017). Attention is all you need. In *Proceedings of the 30th international conference on neural information processing systems* (pp. 5998–6008).

Bengio, Y., Courville, A., Vincent, P. (2013). Representation learning: a review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*, 1798–1828.

Glorot, X., & Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics* (pp. 249–256).

Miller, G.A. (2005). Wordnet: a lexical database for english. *Communications of the Association for Computing Machinery*, *38*, 39–41.

Bollacker, K., Evans, C., Paritosh, P., Sturge, T., Taylor, J. (2008). 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).

Liu, H., Wu, Y., Yang, Y. (2017). Analogical inference for multi-relational embeddings. In *Proceedings of the 34th international conference on machine learning* (pp. 2168–2178).

Han, X., Cao, S., Lv, X., Lin, Y., Liu, Z., Sun, M., Li, J. (2018). OpenKE: an open toolkit for knowledge embedding. In *Proceedings of the 2018 conference on empirical methods in natural language processing: system demonstrations* (pp. 139–144).

Akrami, F., Guo, L., Hu, W., Li, C. (2018). Re-evaluating embedding-based knowledge graph completion methods. In *Proceedings of the 27th ACM international conference on information and knowledge management* (pp. 1779–1782).

Toutanova, K., Chen, D., Pantel, P., Poon, H., Choudhury, P., Gamon, M. (2015). 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).

Nguyen, D.Q., Nguyen, T.D., Nguyen, D.Q., Phung, D. (2018). A novel embedding model for knowledge base completion based on convolutional neural network. In *Proceedings of Annual Conference of the North American Chapter of the Association for Computational Linguistics*.

Duchi, J., Hazan, E., Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, *12*, 2121–2159.

Demsar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, *7*, 1–30.

Wang, Z., Zhang J., Feng J., Chen Z. (2014). Knowledge graph and text jointly embedding. In *Proceedings of the 2014 conference on empirical methods in natural language processing* (pp. 1591–1601).