IEICE TRANS. INF. & SYST., VOL.E103–D, NO.5 MAY 2020

PAPER Special Section on Data Engineering and Information Management

Leveraging Entity-Type Properties in the Relational Context for Knowledge Graph Embedding

Md Mostafizur RAHMAN[†,††a)], *Nonmember and* **Atsuhiro TAKASU**[†,††], *Member*

SUMMARY Knowledge graph embedding aims to embed entities and relations of multi-relational data in low dimensional vector spaces. Knowledge graphs are useful for numerous artificial intelligence (AI) applications. However, they (KGs) are far from completeness and hence KG embedding models have quickly gained massive attention. Nevertheless, the state-of-the-art KG embedding models ignore the category specific projection of entities and the impact of entity types in relational aspect. For example, the entity *"Washington"* could belong to the person or location category depending on its appearance in a specific relation. In a KG, an entity usually holds many type properties. It leads us to a very interesting question: are all the type properties of an entity are meaningful for a specific relation? In this paper, we propose a KG embedding model TPRC that leverages entity-type properties in the relational context. To show the effectiveness of our model, we apply our idea to the TransE, TransR and TransD. Our approach outperforms other state-of-the-art approaches as TransE, TransD, DistMult and ComplEx. Another, important observation is: introducing entity type properties in the relational context can improve the performances of the original translation distance based models.

key words: *knowledge representations, entity types, embedding models*

1. Introduction

Knowledge graphs encode structured facts of real world entities and their rich relations. An increasingly large number of organizations such as Google, Microsoft, Facebook or IBM create and maintain large KGs, as they aid in the integration of heterogeneous data sources, complex query resolution, and structured knowledge exploration. Wikidata [1], DBpedia [2], YAGO [3], and Freebase [4] incorporate very large numbers of facts that facilitate many AI tasks e.g., entity recommendations [5], question answering [6], recommender systems [7], [8] etc. Although existing KGs contain billions of entities and relations, they still have gaps and may contain incorrect facts.

Traditionally, KGs represent the relations/facts between their various entities as triples. A triple can be represented as (h, r, t), where h and t are entities in the real world and r is a relation between h and t. For example, consider the triple (*Tokyo, CapitalOf, Japan*), where *Tokyo, Japan* are the head and tail entities, respectively, and *CapitalOf* is the relation. KG completion problem is similar to link prediction in social network. The purpose of link prediction is to detect unknown pairs of head and tail entities that are correlated via some relation. For example, if the KG contains facts like (*ShinzōAbe, PrimeMinisterOf, Japan*) and (*AkieAbe, SpouseOf, Shinzō Abe*) but the fact (*AkieAbe, HasNationality, Japan*) is not stored, then we would like the machines to complete the missing link between the entity *AkieAbe* and entity *Japan* automatically by link prediction.

The concept of "embedding" has also been widely used for representing words and texts [9], [10], with many embedding models having been proposed for KG completion. Most of these models fall into one of three categories: bilinear models, neural-network-based models, and translation-distance-based models.

The translation-distance-based models have gained popularity both for their simplicity and their effectiveness, where they have achieved state-of-the-art performance. Bordes et al. [11] proposed TransE, which is the simplest and smartest way of predicting the links in a KG. TransE was inspired by Mikolov's skip-gram model [10], [12]. It learns vector embeddings for entities and relations, with relations being represented as translations in the embedding space. The basic principle is that $\boldsymbol{h} + \boldsymbol{r} \approx \boldsymbol{t}$, where (h, r, t) holds. Here, $\boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t}$ are each embeddings of h, r, t, respectively. To solve the one-to-many/many-to-one/many-to-many issues in TransE, TransH [13] has been proposed. It involves a principle stating that entity representations will differ based on various relations. Similarly, TransR [14] assumes that each relation has its own embedding space. However, TransR proposes using separate spaces for entities and relations. TransD [15] proposed dynamic mapping matrix to improve the performance of TransR, which showed very significant results compared to other translation-distanced-based models.

In these models, the entities' type has been completely ignored. In the real world, entities can be categorized in terms of several types, such as person, movie, or organization. We can often assume that entities of the same type should share strong similarities and that, in their relation, their type also plays an important role. As an example, the *HasNationality* relation requires a person-type head entity and a location/country-type tail entity. On the other hand, the *CapitalOf* relation requires location-type entities for both head and tail. We can imagine the existence of two different triples for these two relations: (*Washington, HasNationality, U.S.*) and (*Washington, CityOf, U.S.*). Here, the entity "Washington" plays two completely different roles in these two relations, based on their type. However, an entity

Manuscript received June 26, 2019.
Manuscript revised November 4, 2019.
Manuscript publicized February 3, 2020.

[†]The authors are with The Graduate University for Advanced Studies (SOKENDAI), Tokyo, 101–8430 Japan.
[††]The authors are with National Institute of Informatics, Tokyo, 101–8430 Japan.
a) E-mail: rahman@nii.ac.jp

DOI: 10.1587/transinf.2019DAP0007

is usually not associated to a single generic type but rather to a set of more specific types in the context of a specific relation. As an example, in Freebase entity *DonaldJhonTrump* has 32 types information including "Person", "Organization founder", "Businessman", "Celebrity", "Politician", "Actor", "Architectural structure owner". Consider two triples: (*DonaldJohnTrump*, *PresidentOf*, *USA*) and (*DonaldJohnTrump*, *StarredIn*, *HomeAlone2*). In the first triple, the most expressive type of the head entity should be "Politician" in the context of relation *PresidentOf* and appropriate type of the head entity *DonaldJohnTrump* for second triple would be "Actor". Fortunately, Freebase provides very rich *rdfs#domain* and *rdfs#range* information of entity types for each relation, considering the relational context.

In our model, we explicitly define the role of entity type in a relation. We propose a model that leverages entity type properties in the relational context (TPRC), where, for each relation r, entities are mapped based on both type and relationship.

This paper is the extension of the previously published paper [16]. In [16], we exploited the basic entity type information of entities e.g., "Person", "Location" etc. The basic type information for real-world entities had been collected from the schema.org† vocabulary. According to the definitions in schema.org, there are 10 basic types of real-world entities. Although one entity may involve in various subcategories/subtypes, in [16], we focused only on basic entity types. So the representation for one entity was same for every relation. In this paper, we propose a more fine-grained model, which introduces entities' type mapping matrix considering relational context. TPRC defines the role of types in the head or tail entity more explicitly and clearly. For TPRC entity representation changes based on their role and relation. Our model can be easily combined with other state-of-the-art models to produce more accurate predictions. We evaluated our model using two tasks that involve link prediction and triple classification on the standard datasets FB15K, FB15k-237, YAGO3-10 and FB13.

2. Related Work

In its present state, KG technology is far from fully matured, although link prediction is an effective approach to completing a KG. Various models have been proposed to address the link-prediction issue. The models proposed to date differ in terms of their scoring function.

First, we describe the notation used in this paper. A knowledge graph $G = \{(h, r, t)\} \subseteq E \times R \times E$ can be formalized as a set of triples, where E is the set of all entities and R is the set of all relations. A triple is denoted by (h, r, t), where h is the head entity, r is the relation, and t is the tail entity. The bold letters $\boldsymbol{h}$, $\boldsymbol{r}$, and $\boldsymbol{t}$ denote embeddings of h, r, and t, respectively, in an embedding space $\mathbb{R}^n$. $f_r(\boldsymbol{h}, \boldsymbol{t})$ is the scoring function of the model under consideration.

†http://schema.org

2.1 Translation-Distanced Based Models

2.1.1 Unstructured Model (UM)

UM [17] is the preliminary image of TransE, considering only entities as embeddings. Because UM ignores relations, its scoring function is a simplification of that used in TransE. The scoring function is given as:

$$f_r(\boldsymbol{h}, \boldsymbol{t}) = \| \boldsymbol{h} - \boldsymbol{t} \|_{l_{1/2}}, \tag{1}$$

where $\boldsymbol{h}$ and $\boldsymbol{t}$ are the embeddings of head and tail, respectively.

2.1.2 Structure Embedding (SE)

Bordes proposed the SE model [19], which introduces two different matrices to project separately the head and tail entities for each relation. Its scoring function is defined as follows:

$$f_r(\boldsymbol{h}, \boldsymbol{t}) = \| \boldsymbol{M}_{rh}\boldsymbol{h} - \boldsymbol{M}_{rt}\boldsymbol{t} \|_{l_{1/2}}, \tag{2}$$

where $\boldsymbol{h}, \boldsymbol{t} \in \mathbb{R}^n$ and $\boldsymbol{M}_{rh}, \boldsymbol{M}_{rt} \in \mathbb{R}^{n \times n}$ are the role-specific projection matrices for the head entity and tail entity, respectively.

2.1.3 TransE, TransH, and TransR/CTransR

TransE [11] learns embedding as $\boldsymbol{h} + \boldsymbol{r} \approx \boldsymbol{t}$ where (h, r, t) holds. Therefore, $(\boldsymbol{h} + \boldsymbol{r})$ is very close to $\boldsymbol{t}$, when (h, r, t) holds. Here, the intuition is learning distributed word representations to capture linguistic regularities such as $Tokyo + CapitalOf \approx Japan$. TransE is the most popular translation-distance-based embedding model and is both very simple and fast.

Many researchers [13], [14] have claimed that TransE has problems in representing one-to-many, many-to-one, and many-to-many relations, with a number of models being proposed to address these issues.

The first such effort was TransH [13], which represents relations by hyperplanes. This model projects entities on the hyperplane corresponding to a relation. A single entity can have different representations on different hyperplanes. TransH models the relation r as $\boldsymbol{r}$ on a hyperplane with the normal vector $\boldsymbol{w_r}$. Given a triple (h, r, t), the entity representations $\boldsymbol{h}$ and $\boldsymbol{t}$ are projected on the hyperplane of $\boldsymbol{w_r}$ with the restriction that $\| \boldsymbol{w_r} \| = 1$. The calculation is expressed as:

$$\begin{aligned} \boldsymbol{h}_\perp &= \boldsymbol{h} - \boldsymbol{w}_r^\top \boldsymbol{h} \boldsymbol{w}_r, \\ \boldsymbol{t}_\perp &= \boldsymbol{t} - \boldsymbol{w}_r^\top \boldsymbol{t} \boldsymbol{w}_r. \end{aligned} \tag{3}$$

The scoring function is very similar to TransE:

$$f_r(\boldsymbol{h}, \boldsymbol{t}) = \| \boldsymbol{h}_\perp + \boldsymbol{r} - \boldsymbol{t}_\perp \|_{l_{1/2}}. \tag{4}$$

TransR [14] also addressed the flaws of TransE, but in

a slightly different way than did TransH. TransR considers separate spaces for entities and relations, but the main principle is that entities and relations are completely different types of objects, implying that they should not occupy the same vector space. Given a triple (h, r, t), TransR projects the entity representations $\boldsymbol{h}$ and $\boldsymbol{t}$ into the space specific to a relation r. That is:

$$\boldsymbol{h}_r = \boldsymbol{M}_r\boldsymbol{h}, \quad \boldsymbol{t}_r = \boldsymbol{M}_r\boldsymbol{t}, \tag{5}$$

where $\boldsymbol{h}, \boldsymbol{t} \in \mathbb{R}^n$, $\boldsymbol{r} \in \mathbb{R}^m$, and $\boldsymbol{M}_r \in \mathbb{R}^{n\times m}$ represents the projection matrix from the entity space to the relation space for relation r. The scoring function is:

$$f_r(\boldsymbol{h}, \boldsymbol{t}) = \| \boldsymbol{h}_r + \boldsymbol{r} - \boldsymbol{t}_r \|_{l_{1/2}}. \tag{6}$$

CTransR is an extension of TransR proposed by the same authors. In this model, entity pairs for a relation are clustered into different groups, and the pairs in the same group share the same unique relation vector.

2.1.4 TransD

TransD [15] can be considered as a special case of TransR. It replaces transfer matrix by the product of two projection vectors of an entity and relation pair. Specifically, for each triple (h, r, t), TransD introduces additional mapping vectors $\boldsymbol{h_d}, \boldsymbol{t_d} \in \mathbb{R}^n$ and $\boldsymbol{r_d} \in \mathbb{R}^m$, along with the entity or relation representations $\boldsymbol{h}, \boldsymbol{t} \in \mathbb{R}^n$ and $\boldsymbol{r} \in \mathbb{R}^m$. Projection matrices for head/tail are accordingly defined as:

$$\begin{aligned} \boldsymbol{M}_{rh} &= \boldsymbol{r}_d\boldsymbol{h}_d^\top + \boldsymbol{I}, \\ \boldsymbol{M}_{rt} &= \boldsymbol{r}_d\boldsymbol{t}_d^\top + \boldsymbol{I}. \end{aligned} \tag{7}$$

These two projection matrices are then applied on the head entity $\boldsymbol{h}$ and the tail entity $\boldsymbol{t}$ respectively to get their projections, i.e.,

$$\hat{\boldsymbol{h}} = \boldsymbol{M}_{rh}\boldsymbol{h}, \quad \hat{\boldsymbol{t}} = \boldsymbol{M}_{rt}\boldsymbol{t}, \tag{8}$$

TransD obtains state-of-the-art performance on triplet classification and link prediction tasks.

2.2 Bilinear and Neural-Network (NN) Based Models

For the link-prediction and triple-classification tasks, bilinear and neural-network-based models are also popular. RESCAL [20], [21] is a bilinear model, with each relation being represented by an *n-by-n* matrix in an embedding space $\mathbb{R}^n$ and the scores for the triples being calculated by a bilinear mapping.

DistMult [22] simplifies RESCAL by restricting the matrices to diagonal matrices but it has problem with the score of (h, r, t) and (t, r, h) are the same. ComplEx [23] addressed this issue of DistMult. It uses complex numbers instead of real numbers and takes the conjugate of the embedding of the tail entity before calculating the bilinear mapping.

The SLM model [24], proposed by Socher, concatenates head and tail entities as an input layer to the nonlinear hidden neural layer and has the scoring function:

$$f_r(\boldsymbol{h}, \boldsymbol{t}) = \boldsymbol{u}_r^\top f(\boldsymbol{T_{r1}}\boldsymbol{h} + \boldsymbol{T_{r2}}\boldsymbol{t} + \boldsymbol{b}_r), \tag{9}$$

where $\boldsymbol{T_{r1}}$ and $\boldsymbol{T_{r2}}$ are weighting matrices and $f(\cdot)$ is the *tanh* operation.

The NTN model [24] is an extension of the SLM model. It considers second-order correlations as inputs to nonlinear hidden neural networks. Its scoring function is:

$$f_r(\boldsymbol{h}, \boldsymbol{t}) = \boldsymbol{u}_r^\top f(\boldsymbol{h}^\top\boldsymbol{T}_r\boldsymbol{t} + \boldsymbol{T_{rh}}\boldsymbol{h} + \boldsymbol{T_{rt}}\boldsymbol{t} + \boldsymbol{b}_r), \tag{10}$$

where $\boldsymbol{T_r}$ represents a three-way tensor, $\boldsymbol{T_{rh}}$ and $\boldsymbol{T_{rt}}$ denote weighting matrices, $\boldsymbol{b_r}$ is the bias, and $f(\cdot)$ is the *tanh* operation. To date, NTN has proved computationally expensive and has scalability issues.

Neural Association Model (NAM) [25] conducts semantic matching with a Deep Neural Network (DNN) architecture. Given a fact (h, r, t), it first concatenates the vector embeddings of the head entity and the relation in the input layer, which gives $z^{(0)} = [\boldsymbol{h}; \boldsymbol{r}] \in \mathbb{R}^{2d}$. The input $z^{(0)}$ is then fed into a DNN consisting of L rectified linear hidden layers such that

$$\begin{aligned} \boldsymbol{a}^{(l)} &= \boldsymbol{M}^{(l)}\boldsymbol{z}^{(l-1)} + b^{(l)}, \ l = 1, \ldots, L, \\ \boldsymbol{z}^{(l)} &= ReLU(\boldsymbol{a}^{(l)}), \ l = 1, \ldots, L \end{aligned} \tag{11}$$

where $\boldsymbol{M}^{(l)}$ and $b^{(l)}$ represent the weight matrix and bias for the l-th layer respectively. After the feed-forward process, the score is given by matching the output of the last hidden layer and the embedding of the tail entity, $f_r(\boldsymbol{h}, \boldsymbol{t}) = \boldsymbol{t}^\top\boldsymbol{z}^{(L)}$.

In recent times, several convolutional models have been proposed for solving link prediction task. Dettmers et al. [26] proposed a multi-layer convolutional network model ConvE which is very efficient in terms of time and space complexity compare to other NN based models proposed earlier. Another convolutional model called ConvR [27] which enabled rich interactions between entities and relation representations and achieved the state-of-the-art performance in link prediction task. In this paper, we proposed a model based on translation distance based model but extending our model with convolutional models can be an interesting future work.

However, bilinear models add more redundancy than do translation-distance-based models. For this reason, they can have an overfitting problem. TPRC considers only entities' type constraints in relational context, aiming to retain simple model estimation. It exploits linear mapping and involves less parameter overhead than the latent-variable models. Although neural-network-based models also tend to encounter overfitting, the standard advantage of such models is that they can capture many kinds of relations. Other issues of NN based models are time and space complexity compare to translation distance based models and bilinear models though recently proposed NN based models e.g., ConvE [26] are highly parameters efficient.

2.3 Models Incorporating Entity Types

SSE [28], [29] is another model which incorporates the cat-

egory information into KG embeddings, which requires entities of the same type to stay close. It employs two manifold learning algorithm for representation learning based on semantic smoothness assumption i.e., Laplacian Eigenmaps (LE) [30] and Locally Linear Embedding (LLE) [31]. SSE introduces some hard constraints like each entity happens to belong to only one category, which is inconsistent with the KG. SSE can be extended with other translation distanced based models, Bilinear and Neural Network based models. SSE used LE and LLE regularization methods to measure the smoothness of the embedding space. The major limitation of SSE is: SSE introduces some hard constraints like each entity happens to belong to only one category, which is inconsistent with the KG. Another problem is each entity has same representation for all the relations where it appears.

TKRL proposed by Xie et al. [32], which considers hierarchical entity categories. The scoring function is accordingly defined as:

$$f_r(\boldsymbol{h}, \boldsymbol{t}) = \| \boldsymbol{M}_{rh}\boldsymbol{h} - \boldsymbol{r} + \boldsymbol{M}_{rt}\boldsymbol{t} \|_{l_1}, \tag{12}$$

where $\boldsymbol{M}_{rh}$ and $\boldsymbol{M}_{rt}$ are projection matrices for h and t. $\boldsymbol{M}_{rh}/\boldsymbol{M}_{rt}$ can be presented as a weighted sum of all possible type matrices for each entity, i.e.,

$$\boldsymbol{M}_{rh} = \frac{\sum_{i=1}^{n_h} \alpha_i \boldsymbol{M}_{c_i}}{\sum_{i=1}^{n_h} \alpha_i}, \alpha_i = \begin{cases} 1, & c_i \in C_{rh} \\ 0, & c_i \notin C_{rh} \end{cases} \tag{13}$$

where n_h is the number of categories to which h belongs; c_i is the i-th category among the set of types of an entity; $\boldsymbol{M}_c$ is the projection matrix of c_i; α_i the corresponding weight; and C_{rh} the set of types of an entity in relation r. TKRL employed two types of encoders to compute the projection matrix $\boldsymbol{M}_c$: Recursive Hierarchy Encoder (RHE) and Weighted Hierarchy Encoder (WHE). They are defined in [32] as follows:

$$\begin{aligned} RHE &: \boldsymbol{M}_{c_i} = \boldsymbol{M}_{c_i^{(1)}} \boldsymbol{M}_{c_i^{(2)}} \dots \boldsymbol{M}_{c_i^{(l)}} \\ WHE &: \boldsymbol{M}_{c_i} = \beta_1 \boldsymbol{M}_{c_i^{(1)}} + \dots + \beta_l \boldsymbol{M}_{c_i^{(l)}} \end{aligned} \tag{14}$$

Here $c_i^{(1)}, \dots, c_i^{(l)}$ are sub-types of c_i in the hierarchy; $\boldsymbol{M}_{c_i^{(1)}}, \dots, \boldsymbol{M}_{c_i^{(l)}}$ their projection matrices; and $\beta_1, \dots, \beta_l$ the corresponding weights. They also applied a sampling method called Soft Type Constraint (STC) with their encoders. TKRL is an extended version of original TransE. It has very high time and space complexity since each entity may have multiple sub-types. On an average TKRL considered 8 sub-types for each entity for FB15k dataset. That means for each entity there are 8 type-embodied matrices. Another model proposed by Krompaß [33] which is a latent-variable model that also consider relation and entity type constraints.

2.4 Complexity Analysis

The parameters and the complexity of the related works are listed in Table 1. Here, N_e and N_r are the number of entities and relations respectively; n and m the dimensionality of entity and relation embedding space respectively (here, we assume $n = m$); $\bar{t}$ is the number of relations specific entity types; $\bar{n}$ the number of entities which have type information (for SSE model); $\bar{k}$ the number of sub-types for each entity; and L is the total number of hidden layers in the network in NAM model. SSE [28], [29] extended several models (e.g., TransE [11], RESCAL [20], SME [18] etc) using LE/LLE regularization as mentioned earlier in Section. In Table 1 *-LE/*-LEE denotes a model with the LE/LLE regularizer included. For *-LE/*-LEE: $\boldsymbol{E}$ is a matrix consisting of the entity embeddings; $\boldsymbol{D}, \boldsymbol{I} \in \mathcal{R}^{(n \times n)}$ are the diagonal and Identity matrices respectively. During the analysis we assume that $n, m \ll N_e$and all the models are trained under the open world assumption.

We can draw the following conclusions. First, models which represent entities and relations as vectors (e.g., TransE, TransH, TransD, and ComplEx) are more efficient. They usually have space and time complexity that scales linearly with n. Second, models which represent relations as matrices (e.g., TransR, $\text{TPRC}_{\text{TransD}}$, SE, and RESCAL) usually have higher complexity in both space and time, scaling quadratically or cubically with the dimensionality of embedding space. Third, models based on neural network (e.g., SME, NTN, and NAM) generally have higher complexity in time, if not in space, since matrix or even tensor computations are often required in these models (recent NN based models are very efficient in terms of time and space complexity e.g. ConvE [26]). Finally, TPRC incorporates relations specific types with the state-of-the-art translation based models which increase the complexity on updating embeddings. $\text{TPRC}_{\text{TransD}}$ achieves (see section) the better performance in our experiment and it's time complexity is $O(n^2)$, which is larger than TransD original model but considering the performance issue we think it's reasonable.

3. Our Method

Translation-distance-based embedding models mostly follow TransE. Both TransE and TransH assume embeddings of entities and relations within the same space $\mathbb{R}^n$. However, relations and entities are completely different objects and it may not be appropriate to represent them in a common semantic space. Although TransH extends modeling flexibility by employing relation hyperplanes, it does not fully address the restrictions of this assumption. In contrast, the entities in TransR are mapped to vectors in different relational spaces, according to their relations. TransD considers the diversity of relations and entities. However, none of these models consider the significance of entity type. Therefore, they cannot judge the exact role of each entity, based on its relation. In our model, we deliberately include the type information. The entity's type can be incorporated easily by introducing an entity type-mapping matrix. For the relations, the entity type information plays a significant role. For example, the "*CapitalOf*" relation would imply that both the head and the tail would be location type entities. If we think more precisely and look for more compact

Table 1 Parameters and complexity of related works.

Models	Parameters	Space Complexity	Time Complexity
RESCAL [20]	$\boldsymbol{h}, \boldsymbol{t} \in \mathbb{R}^n, \boldsymbol{W}_r \in \mathbb{R}^{n \times n}$	$O(N_e n + N_r n^2)$	$O(n^2)$
RESCAL-LE [29]	$\boldsymbol{h}, \boldsymbol{t} \in \mathbb{R}^n, \boldsymbol{E}, \boldsymbol{R}_k, \boldsymbol{W}_r \in \mathbb{R}^{n \times n}$	$O(N_e n + \bar{n}^2 n + N_r n^2)$	$O(n^2)$
RESCAL-LLE [29]	$\boldsymbol{h}, \boldsymbol{t} \in \mathbb{R}^n, \boldsymbol{E}, \boldsymbol{W}_r \in \mathbb{R}^{n \times n}$	$O(N_e n + \bar{n}^2 K n + N_r n^2)$	$O(n^2)$
BILINEAR [34]	$\boldsymbol{h}\boldsymbol{W}_r t, \boldsymbol{W}_r, \boldsymbol{E}, \boldsymbol{D} \in \mathbb{R}^{n \times n}$	$O(N_e n + N_r n^2)$	$O(n^2)$
BILINEAR-LE [29]	$\boldsymbol{h}\boldsymbol{W}_r t, \boldsymbol{W}_r, \boldsymbol{E}, \boldsymbol{D} \in \mathbb{R}^{n \times n}$	$O(N_e n + \bar{n}^2 n + N_r n^2)$	$O(n^2)$
BILINEAR-LLE [29]	$\boldsymbol{h}\boldsymbol{W}_r \boldsymbol{t}, \boldsymbol{E}, \boldsymbol{I} \in \mathbb{R}^{n \times n}$	$O(N_e n + \bar{n} K n + N_r n^2)$	$O(n^2)$
DistMult [22]	$\boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t} \in \mathbb{R}^n$	$O(N_e n + M_r n)$	$O(n)$
ComplEx [23]	$\boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t} \in \mathbb{C}^n$	$O(N_e n + M_r n)$	$O(n)$
UM [17]	$\boldsymbol{h}, \boldsymbol{t} \in \mathbb{R}^n$	$O(N_e n)$	$O(n)$
SE [19]	$\boldsymbol{h}, \boldsymbol{t} \in \mathbb{R}^n, \boldsymbol{M}_{rh}, \boldsymbol{M}_{rt} \in \mathbb{R}^{n \times n}$	$O(N_e n + N_r n^2)$	$O(n^2)$
SE-LE [29]	$\boldsymbol{h}, \boldsymbol{t} \in \mathbb{R}^n, \boldsymbol{E}, \boldsymbol{D} \in \mathbb{R}^{n \times n}$	$O(N_e n + \bar{n}^2 n + N_r n^2)$	$O(n^2)$
SE-LLE [29]	$\boldsymbol{h}, \boldsymbol{t} \in \mathbb{R}^n, \boldsymbol{I}, \boldsymbol{D} \in \mathbb{R}^{n \times n}$	$O(N_e n + \bar{n} K d + N_r n^2)$	$O(n^2)$
SME (lin) [18]	$\boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t} \in \mathbb{R}^n$	$O(N_e n + N_r d)$	$O(n^2)$
SME (lin)-LE [29]	$\boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t} \in \mathbb{R}^n, \boldsymbol{E}, \boldsymbol{D} \in \mathbb{R}^{n \times n}$	$O(N_e n + \bar{n}^2 n + N_r n)$	$O(n^2)$
SME (lin)-LLE [29]	$\boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t} \in \mathbb{R}^n, \boldsymbol{I}, \boldsymbol{D} \in \mathbb{R}^{n \times n}$	$O(N_e n + \bar{n} K n + N_r n)$	$O(n^2)$
SME (bilin) [18]	$\boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t} \in \mathbb{R}^n$	$O(N_e d + N_r n)$	$O(n^3)$
SME (bilin)-LE [29]	$\boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t} \in \mathbb{R}^n, \boldsymbol{E}, \boldsymbol{D} \in \mathbb{R}^{n \times n}$	$O(N_e n + \bar{n}^2 n + N_r n)$	$O(n^3)$
SME (bilin)-LLE [29]	$\boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t} \in \mathbb{R}^n, \boldsymbol{I}, \boldsymbol{D} \in \mathbb{R}^{n \times n}$	$O(N_e n + \bar{n} K n + N_r n)$	$O(n^3)$
TKRL [32]	$\boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t} \in \mathbb{R}^n, \boldsymbol{M}_{rh}, \boldsymbol{M}_{rt} \in \mathbb{R}^{n \times n}$	$O(N_e n + N_r n^{\bar{k}})$	$O(n^{\bar{k}})$
SLM [24]	$\boldsymbol{h}, \boldsymbol{t} \in \mathbb{R}^n, \boldsymbol{r} \in \mathbb{R}^n, \boldsymbol{M}_r^1, \boldsymbol{M}_r^2 \in \mathbb{R}^{n \times n}$	$O(N_e n + N_r n^2)$	$O(n^2)$
NTN [24]	$\boldsymbol{h}, \boldsymbol{t} \in \mathbb{R}^n, \boldsymbol{u}_r, \boldsymbol{b}_r \in \mathbb{R}^n, \boldsymbol{T}_r \in \mathbb{R}^{n \times n \times n}, \boldsymbol{T}_{r,h}, \boldsymbol{T}_{r,t} \in \mathbb{R}^{n \times n}$	$O(N_e n + N_r n^3)$	$O(n^3)$
NAM [25]	$\boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t} \in \mathbb{R}^n$	$O(N_e n + N_r n)$	$O(Ln^2)$
ConvE [26]	$\boldsymbol{h}, \boldsymbol{t} \in \mathbb{R}^n, \boldsymbol{r} \in \mathbb{R}^n$	$O(N_e n + N_r n)$	$O(n)$
TransE [11]	$\boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t} \in \mathbb{R}^n$	$O(N_e n + N_r n)$	$O(n)$
TransE-LE [29]	$\boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t} \in \mathbb{R}^n, \boldsymbol{E}, \boldsymbol{D} \in \mathbb{R}^{n \times n}$	$O(N_e n + \bar{n}^2 n + N_r n)$	$O(n)$
TransE-LLE [29]	$\boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t} \in \mathbb{R}^n, \boldsymbol{E}, \boldsymbol{I} \in \mathbb{R}^{n \times n}$	$O(N_e n + \bar{n} K n + N_r d)$	$O(n)$
TransH [13]	$\boldsymbol{h}, \boldsymbol{t} \in \mathbb{R}^n, \boldsymbol{r}, \boldsymbol{W}_r \in \mathbb{R}^n$	$O(N_e n + N_r n)$	$O(n)$
TransR [14]	$\boldsymbol{h}, \boldsymbol{t} \in \mathbb{R}^n, \boldsymbol{r} \in \mathbb{R}^n, \boldsymbol{M}_r \in \mathbb{R}^{n \times n}$	$O(N_e n + N_r n^2)$	$O(n^2)$
TransD [15]	$\boldsymbol{h}_d, \boldsymbol{t}_d, \boldsymbol{h}, \boldsymbol{t} \in \mathbb{R}^n, \boldsymbol{r}_d, \boldsymbol{r} \in \mathbb{R}^n$	$O(N_e n + N_r n)$	$O(n)$
$\text{TPRC}_{\text{TransE}}$ (this paper)	$\boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t} \in \mathbb{R}^n, \hat{\boldsymbol{M}}_{rh}, \hat{\boldsymbol{M}}_{rt} \in \mathbb{R}^{n \times n}$	$O(N_e n + \bar{t} n^2 + N_r n)$	$O(n^2)$
$\text{TPRC}_{\text{TransR}}$ (this paper)	$\boldsymbol{h}, \boldsymbol{t} \in \mathbb{R}^n, \boldsymbol{r} \in \mathbb{R}^n, \hat{\boldsymbol{M}}_{rh}, \hat{\boldsymbol{M}}_{rt} \in \mathbb{R}^{n \times n}$	$O(N_e n + \bar{t} n^2 + N_r n^2)$	$O(n^2)$
$\text{TPRC}_{\text{TransD}}$ (this paper)	$\boldsymbol{h}_d, \boldsymbol{t}_d, \boldsymbol{h}, \boldsymbol{t} \in \mathbb{R}^n, \boldsymbol{r}_d, \boldsymbol{r} \in \mathbb{R}^n, \hat{\boldsymbol{M}}_{rh}, \hat{\boldsymbol{M}}_{rt} \in \mathbb{R}^{n \times n}$	$O(N_e n + \bar{t} n^2 + N_r n)$	$O(n^2)$

types, then we can understand that, the type of the head entity would be country and the type of the tail entity would be city/town, both of them are sub-types of location. As we mentioned earlier we can obtain such kind of relation-wise rich information from Freebase. Therefore, we propose a more fine-grained model to map the entities with their corresponding types considering relational context. Using type information in a translation model can improve the efficiency of any such model.

An entity may belong to multiple sub-types and it's also possible to obtain hierarchical types of an entity. If we consider the subcategories in the hierarchy when modeling, it will make the model complex, thereby the time and space complexity will increase significantly. To address this issue in our model, we consider only the domain and range of the types for a specific relation.

Considering the diversity of entity types in different relations we propose TPRC. For each relation r, we introduce type-embodied mapping matrices $\hat{\boldsymbol{M}}_{rh}, \hat{\boldsymbol{M}}_{rt} \in \mathbb{R}^{n \times n}$ for the head and tail entity based on relational context. It means the type mapping matrix for an entity can be different based on the relation it appears. With the mapping matrices, we defined the projected vectors of entities as:

$$\boldsymbol{h}_p = \hat{\boldsymbol{M}}_{rh} \boldsymbol{h}, \quad \boldsymbol{r}_p = \hat{\boldsymbol{M}}_{rt} \boldsymbol{t}, \tag{15}$$

where $(\boldsymbol{h}, \boldsymbol{t}) \in \mathbb{R}^n$. In our model, we enforce the constraints $\| \boldsymbol{h} \|_2 \leq 1$, $\| \boldsymbol{t} \|_2 \leq 1$, $\| \hat{\boldsymbol{M}}_{rh} \boldsymbol{h} \|_2 \leq 1$, and $\| \hat{\boldsymbol{M}}_{rt} \boldsymbol{t} \|_2 \leq 1$. It is not mandatory to have the same dimensionality for entity embeddings and entities' type embeddings. However, in our experiments to learn vectors and matrices, we keep the same dimensionality. The scoring function for each specific relation r (where $\boldsymbol{r} \in \mathbb{R}^n$) is correspondingly defined as:

$$f_r(\boldsymbol{h}, \boldsymbol{t}) = \| \boldsymbol{h}_p + \boldsymbol{r} - \boldsymbol{t}_p \|_{l_{1/2}}. \tag{16}$$

TPRC can be easily combined with other state-of-the-art translation-distance-based models due to its simplicity. In this paper we apply it to TransE, TransR and TransD.

TransE

TransE is the most representative translational distance based model. The score function of TransE is defined as:

$$f(\boldsymbol{h}, \boldsymbol{t}) = \| \boldsymbol{h} + \boldsymbol{r} - \boldsymbol{t} \|_{l_{1/2}}, \tag{17}$$

where $\boldsymbol{h}, \boldsymbol{t} \in \mathbb{R}^n$ and $\boldsymbol{r} \in \mathbb{R}^n$ are the vectors of a head entity, a tail entity, and a relation on a single embedding space.

If we incorporate TransE into the entities' type-mapping matrix model, denoted by $\text{TPRC}_{\text{TransE}}$ then the model's scoring function will be given by Eq. (12). TransE has the same vector representation for each entity. In $\text{TPRC}_{\text{TransE}}$, the head and tail entities are mapped according

to their type based on the relation they appear.

TransR

In TransR, entities are mapped separately to a relation space. Equations (5) and (6) denote the embeddings and the scoring function of the TransR method. If we incorporate TransR into the entities' type-mapping matrix model ($\text{TPRC}_{\text{TransR}}$), then the embeddings of head and tail entities become (we keep the same dimension for entity and relation vectors):

$$\boldsymbol{h}_p^r = (\hat{\boldsymbol{M}}_{rh}\boldsymbol{h})\boldsymbol{M}_r, \quad \boldsymbol{t}_p^r = (\hat{\boldsymbol{M}}_{rt}\boldsymbol{t})\boldsymbol{M}_r \tag{18}$$

and the scoring function is defined as:

$$f_r(\boldsymbol{h}, \boldsymbol{t}) = \| \boldsymbol{h}_p^r + \boldsymbol{r} - \boldsymbol{t}_p^r \|_{l_{1/2}}. \tag{19}$$

The TransR mapping matrix $\boldsymbol{M}_r$ is the same for both head and tail entities. The key difference between our proposed model ($\text{TPRC}_{\text{TransR}}$) and the traditional TransR is that, before projecting to a relation space, entities are mapped via the entities' type-mapping matrix. In the relation space, entities therefore have vector representations based on their type and a specific relation property.

TransD

Different types of entities have different attributes and functions. It's the principle of TransD. The concept of TransD is very close to TPRC. This model defines the interaction between the vectors of different types of entities and relations (see Eq. (7)) implicitly. On the other hand, TPRC explicitly care about the entity types based on the relation. By applying, TPRC to TransD ($\text{TPRC}_{\text{TransD}}$), we define more fine-graned representations of entities and relations than TransD itself. The score function for TransD after projecting the entities into relational space is (Eqs. (7) and (8) define the projection procedure):

$$f_r(\boldsymbol{h}, \boldsymbol{t}) = \| \hat{\boldsymbol{h}} + \boldsymbol{r} - \hat{\boldsymbol{t}} \|_{l_{1/2}}. \tag{20}$$

For $\text{TPRC}_{\text{TransD}}$ the embeddings of the head h and tail t are:

$$\boldsymbol{h}_p^d = (\hat{\boldsymbol{M}}_{rh}\boldsymbol{h})\boldsymbol{M}_{rh}, \quad \boldsymbol{t}_p^d = (\hat{\boldsymbol{M}}_{rt}\boldsymbol{t})\boldsymbol{M}_{rt} \tag{21}$$

and the scoring function is defined as:

$$f_r(\boldsymbol{h}, \boldsymbol{t}) = \| \boldsymbol{h}_p^d + \boldsymbol{r} - \boldsymbol{t}_p^d \|_{l_{1/2}}. \tag{22}$$

The design of the model TransR and TransD is identical except the mapping matrices. So the main difference between $\text{TPRC}_{\text{TransR}}$ and TransD is similar to the difference of $\text{TPRC}_{\text{TransD}}$ and TransD, before projecting to a relation space, entities are mapped via the entities' type-mapping matrices $\hat{\boldsymbol{M}}_{rh}\boldsymbol{h}, \hat{\boldsymbol{M}}_{rt}\boldsymbol{t}$ for head and tail accordingly.

In the above discussion, we have shown how to combine TPRC with the state-of-the-art translational distance based models. In the same way, we could combine TPRC with other existing translation-distance-based models.

4. Training

We use a margin-based loss function for training:

$$L = \sum_{(h,r,t)\in S} \sum_{(h',r,t')\in S'} max(0, f_r(h,t) + \gamma - f_r(h',t')). \tag{23}$$

Here, γ is the margin, S is the set of correct triples, and S' is the set of incorrect triples. The ratio of the correct and incorrect samples is same, as we can guess it from the objective function. Existing KGs should only contain correct triples. An S' is constructed by replacing a head or a tail entity in an existing triple as follow:

$$\begin{aligned} S' = &\{(h', r, t) | h' \in E \wedge e' \neq h \wedge (h, r, t) \in S\} \\ &\cup \{(h, r, t') | t' \in E \wedge t' \neq t \wedge (h, r, t) \in S\} \end{aligned}$$

We also employ two strategies "unif" and "bern" reported in [13] to replace head or tail entity. We use a stochastic gradient descent (SGD) method to minimize L. TransR and TransD initialize the embeddings of entities and relations obtained from TransE. To avoid overfitting of $\text{TPRC}_{\text{TransR}}$ and $\text{TPRC}_{\text{TransD}}$, we also initialize entity and relation embeddings with the results of $\text{TPRC}_{\text{TransE}}$.

5. Experiment

Our proposed model was evaluated via two tasks: link prediction [11] and triple classification [24]. This section discusses the experimental procedures for our model.

5.1 Datasets

In this paper, the experiments are conducted on four benchmark datasets: FB15k [11], FB15k-237 [35], YAGO3-10 [3] and FB13 [24] are respectively extracted from real knowledge graphs Freebase[†] and YAGO[††]. The link prediction task has been conducted with FB15k, FB15k-237 and YAGO3-10 datasets. In FB15k dataset has redundancy entries as it also includes inverse relations. It contains 14,951 entities, 1,345 relations and 592,213 triples with 541 relation specific types of head/tail entities. On the other hand, FB15k-237 was prepared from FB15k by removing relations that were considered as inverse relations of other relations. FB15k-237 dataset includes 14,541 entities, 237 relations and total 272,115 triples. It has 141 relation specific entity types. Recently, YAGO3-10 becomes a popular benchmark dataset with very large number of triples (1,079,040 triples) comparing to other datasets. Though it contains vast number of triples but it has only 37 relations. We collected the relation specific types of head/tail entities for all the mentioned datasets. TPRC didn't exploit hierarchical types and

[†]https://developers.google.com/freebase/

[††]https://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/yago-naga/yago/

Table 2 Datasets used in the experiments.

Dataset	#E	#R	#T	#Train	#Valid	#Test
FB15k	14,951	1,345	571	483,142	50,000	59,071
FB15k-237	14,541	237	181	272,115	17,535	20,466
FB13	75,043	13	10	316,232	5,908	23,733
YAGO3-10	123,182	37	21	1,079,040	5,000	5,000

we already mentioned earlier how hierarchical types can effect the KG embeddings.

In this paper, the "FB13" dataset has been employed to evaluate the triple classification task. These datasets contain negative triples, which is helpful for this particular task. Moreover, it has only 13 relations and we also collected the relation specific type information for this dataset. Table 2 shows the statistics for the datasets used in this paper, where #E/#R/#T/#Train/#Valid/#Test denotes the number of entities/relations/ entity types/training triples/validation triples and test triples.

5.2 Link Prediction

We follow [11] and formalize link prediction task as a pointwise learning to rank problem, where the objective is learning a scoring function $f_r : E \times R \times E \rightarrow \mathbb{R}$. For a triple (h, r, t), the link-prediction task predicts the missing h or t, given the relation and the other entity. The results were evaluated by ranking the predicted head or tail entity, as calculated by the scoring function $f_r(h, t)$ for test triples.

Experimental Protocol: For our experiments, we adopted the same protocol proposed by Bordes et al. [11]. For each testing triple (h, r, t), we corrupted it by replacing the tail t or head h with every entity e in the KG or the current dictionary and calculated a probabilistic score for the corrupted triple (h, r, e) or (e, r, t), respectively, in terms of the scoring function $f_r(h, e)$. It's the "Raw" setting protocol. Because we have corrupted the triples randomly, this same triple may already exist in the actual KG and would be considered correct. During the ranking, it is logically possible that such triples may appear before the original triple. To eliminate this issue, we intentionally remove those corrupt triples that are created by replacing h or t randomly but that already exist in the KG before computing the rank of each testing triple. They may exist in any of the training, valid, or testing sets. This revised setting protocol is called the"Filter" setting. In addition, we employ the same two sampling methods, "bern" and "unif," that were used in the previous studies.

Three evaluation metrics were used: (1) Mean Rank (MR, the mean of all predicted ranks); (2) Mean Reciprocal Rank (MRR, the mean of all the reciprocals of predicted ranks); and (3) Hits@10 (the proportion of testing triples whose rank did not exceed those of the top 10 predictions). For both settings, a lower MR and, higher MRR and Hits@10 imply a better performance.

Baselines: Though the Goal of TPRC is to increase the effectiveness of translation distance based models but we address the results of the state-of-the-art- bilinear models. To demonstrate the effectiveness of our models, we compare results with the following baselines.
Translation-Distance Based Models: SE [19], UM [17], TKRL [32] TransE [11], TransR [14], TransD [15].
Bilinear Models: RESCAL [20], DistMult [22], ComplEx [23].

Optimization and Implementation: We conducted a grid search to tune suitable parameters of TransE, TransR, TransD, $\text{TPRC}_{\text{TransE}}$, $\text{TPRC}_{\text{TransR}}$ and $\text{TPRC}_{\text{TransD}}$. For bilinear models (DistMult and ComplEx) we followed the instructions in their original paper. We have used the FB15k and FB15k-237 datasets to compare our models with the baseline models. To obtain the best settings for our models, we fine-tuned five parameters. We selected the margin γ from the set {0.5, 1, 2}, the dimensionality of entity and relation vectors from {50, 100, 200, 500, 1000}, the learning rate α from {0.01, 0.05, 001, 0.005, 0.0001, 0.0005}, the number of training triples in each mini-batch from {20, 50, 200, 300, 1440, 2000, 4000, 4800, 5000}, and the dissimilarity measure in the embedding scoring function from $\{L_1, L_2\}$. The parameters of $\text{TPRC}_{\text{TransE}}$, $\text{TPRC}_{\text{TransR}}$ and $\text{TPRC}_{\text{TransD}}$ are initialized as described in [11], [14] and [15] respectively.

The optimal configurations for $\text{TPRC}_{\text{TransE}}$ were $\gamma = 1$, $d = 50$, $\alpha = 0.0001$, $B = 50$, and using L_1 as the dissimilarity function for FB15k dataset; $\gamma = 1$, $d = 50$, $\alpha = 0.0001$, $B = 50$, and using L_1 as the dissimilarity function for FB15k-237 dataset; $\gamma = 1$, $d = 150$, $\alpha = 0.0005$, $B = 100$, and using L_1 as the dissimilarity function for YAGO3-10 dataset.

The optimal configurations for $\text{TPRC}_{\text{TransR}}$ were $\gamma = 1$, $d = 100$, $\alpha = 0.05$, $B = 4800$, and using L_1 as the dissimilarity function for FB15k dataset; $\gamma = 1$, $d = 100$, $\alpha = 0.01$, $B = 1440$, and using L_1 as the dissimilarity function for FB15k-237 dataset; $\gamma = 1$, $d = 200$, $\alpha = 0.05$, $B = 1440$, and using L_1 as the dissimilarity function for YAGO3-10 dataset.

The optimal configurations for $\text{TPRC}_{\text{TransD}}$ were $\gamma = 1$, $d = 100$, $\alpha = 0.05$, $B = 1440$, and using L_2 as the dissimilarity function for FB15k dataset; $\gamma = 1$, $d = 100$, $\alpha = 0.01$, $B = 200$, and using L_2 as the dissimilarity function for FB15k-237 dataset; $\gamma = 1$, $d = 200$, $\alpha = 0.01$, $B = 300$, and using L_2 as the dissimilarity function for YAGO3-10 dataset.

We also had to find optimal parameter settings for original TransE, TransR and TransD. We achieved better results than the original paper by tuning the parameters for these three models. For TransE, they were $\gamma = 1$, $d = 50$, $\alpha = 0.0001$, $B = 50$, and using L_1 as the dissimilarity function for FB15k and FB15k-237 and $\gamma = 1$, $d = 100$, $\alpha = 0.0001$, $B = 100$, and using L_1 as the dissimilarity function for YAGO3-10 dataset. For TransR, they were $\gamma = 1$, $d = 50$, $k = 50$ (dimensionality for relation vectors), $\alpha = 0.001$, $B = 1440$, and using L_1 as the dissimilarity function for FB15k dataset; $\gamma = 1$, $d = 100$, $\alpha = 0.0001$,

Table 3 Link prediction results on FB15k and FB15k-237. The results of RESCAL, UM and SE were reported by [15]. MRR (filter setting) and Hits@10 (filter setting) of DistMult and ComplEx were copied from [23]. MR and Hits@10 for FB15k of TKRL [32] were copied from and for FB15k-237 (TKRL) the code has been taken from https://github.com/thunlp/TKRL. We implemented the extended models of TransE, TransR and TransD. The code of TransE, TransR, TransD, DistMult and ComplEx are taken from https://github.com/thunlp/TensorFlow-TransX and https://github.com/ttrouill/complex. The sign " - " means result is not available in the corresponding paper for that particular matric or dataset.

Dataset	FB15k						FB15k-237					
Metric	MR		MRR		Hits@10		MR		MRR		Hits@10	
	Raw	Filter	Raw	Filter	Raw	Filter	Raw	Filter	Raw	Filter	Raw	Filter
RESCAL	828	683	-	-	0.284	0.441	-	-	-	-	-	-
DistMult	164	95	0.242	0.654	0.401	0.818	391	251	0.133	0.244	0.262	0.419
ComplEx	212	98	0.242	0.692	0.534	0.840	490	339	0.139	0.247	0.248	0.428
UM	1,074	979	-	-	0.045	0.063	-	-	-	-	-	-
SE	273	162	-	-	0.288	0.398	-	-	-	-	-	-
TKRL (RHE)	184	68	0.242	0.410	0.492	0.694	512	232	0.139	0.240	0.283	0.416
TKRL (WHE)	186	68	0.244	0.417	0.492	0.696	523	228	0.135	0.236	0.287	0.419
TKRL (RHE+STC)	202	89	0.267	0.459	0.504	0.731	531	238	0.137	0.249	0.293	0.426
TKRL (WHE+STC)	202	87	0.280	0.464	0.503	0.734	519	236	0.141	0.251	0.291	0.423
TransE (unif)	297	99	0.181	0.353	0.402	0.518	614	401	0.994	0.176	0.244	0.344
TransE (bern)	256	91	0.231	0.386	0.424	0.621	587	375	0.122	0.207	0.267	0.378
TransR (unif)	226	82	0.205	0.388	0.438	0.657	54	387	0.091	0.211	0.270	0.419
TransR (bern)	198	78	0.242	0.408	0.487	0.688	510	401	0.132	0.247	0.273	0.420
TransD (unif)	240	77	0.251	0.462	0.491	0.744	509	360	0.142	0.235	0.275	0.406
TransD (bern)	210	90	0.456	0.658	0.546	0.781	545	396	0.147	0.252	0.289	0.443
$TPRC_{TransE}$ (unif)	223	92	0.241	0.403	0.433	0.568	599	356	0.124	0.199	0.288	0.384
$TPRC_{TransE}$ (bern)	198	87	0.241	0.398	0.464	0.642	568	370	0.129	0.215	0.299	0.378
$TPRC_{TransR}$ (unif)	222	79	0.243	0.411	0.488	0.667	510	360	0.121	0.242	0.301	0.419
$TPRC_{TransR}$ (bern)	166	83	0.272	0.448	0.507	0.690	499	321	0.141	0.251	0.302	0.428
$TPRC_{TransD}$ (unif)	141	**61**	0.355	0.599	0.613	0.802	480	256	0.149	0.255	0.312	0.437
$TPRC_{TransD}$ (bern)	**102**	81	**0.506**	**0.701**	**0.644**	**0.846**	**387**	**157**	**0.168**	**0.286**	**0.322**	**0.468**

$B = 1440$, and using L_1 as the dissimilarity function for FB15k-237 dataset; $\gamma = 1$, $d = 200$, $\alpha = 0.0005$, $B = 1440$, and using L_1 as the dissimilarity function for YAGO3-10 dataset. The optimal configurations for TransD for FB15k and FB15k-237 datasets were: $\gamma = 1$, $d = 100$, $\alpha = 0.0001$, $B = 300$, and using L_2 as the dissimilarity function; for YAGO3-10 dataset configurations were $\gamma = 1$, $d = 200$, $\alpha = 0.0005$, $B = 300$, and using L_1 as the dissimilarity function.

Analysis: Accuracies are reported in the Table 3 and Table 4. Results in bold font are the best obtained results. In Table 4 we reported the filter setting results of YAGO3-10 dataset of our models and other models as well. Here, we have employed "bern" for all the translation distanced based models. From Tables 3 and 4 we have the following findings: (1) Our models outperformed all other methods, using the experimental datasets FB15k and FB15k-237 with both Raw and Filter settings on all metrics. $TPRC_{TransD}$ with "bern" achieves the best results on MR, MRR and Hits@10 in the both experimental datasets except the MR value (filter setting) of FB15k of $TPRC_{TransD}$ with "unif" achieves the best performance. The closest competitor is ComplEx model. DistMust also showed very promising results. Though ComplEx was slightly better than TransD but exploiting type information in relational context helps the performance of $TPRC_{TransD}$. In YAGO3-10 dataset ComplEX and $TPRC_{TransD}$ achieved the best performance for hits@10 and $TPRC_{TransD}$ outperformed all other methods on other metrics. Another interesting observation is no model

Table 4 Link prediction results on YAGO3-10 (filter setting). We implemented the extended models of TransE, TransR and TransD. The code of TransE, TransR, TransD, DistMult and ComplEx are taken from https://github.com/thunlp/TensorFlow-TransX and https://github.com/ttrouill/complex

Models	MR	MRR	Hits@10
RESCAL	-	-	-
DistMult	6051	0.351	0.563
ComplEx	6134	0.380	**0.592**
UM	-	-	-
SE	-	-	-
TransE	6499	0.211	0.435
TransR	5926	0.224	0.471
TransD	4803	0.361	0.558
$TPRC_{TransE}$	6400	0.241	0.476
$TPRC_{TransR}$	5115	0.229	0.482
$TPRC_{TransD}$	**4386**	**0.382**	**0.592**

achieved very significant performance compare to other state-of-the-art models in YAGO3-10 dataset. The test samples of this dataset is 5,000 only, which is very small compare to FB15k and FB15k-237 datasets. We also observe that prior information of head or tail entities' types contribute in the prediction face on our other models; (2) The results appear to show that the "bern" (Table 3) sampling method performs slightly better than the "unif" method for the translation distance based models; (3) All the translation distance based models showed good performance for the dataset FB15k-237. For this dataset $TPRC_{TransD}$ (bern) significantly outperform DistMult, ComplEx and other models on MR, MRR and Hits@10. As we mentioned earlier FB15k-237 has no inverse relations, so it shows that transla-

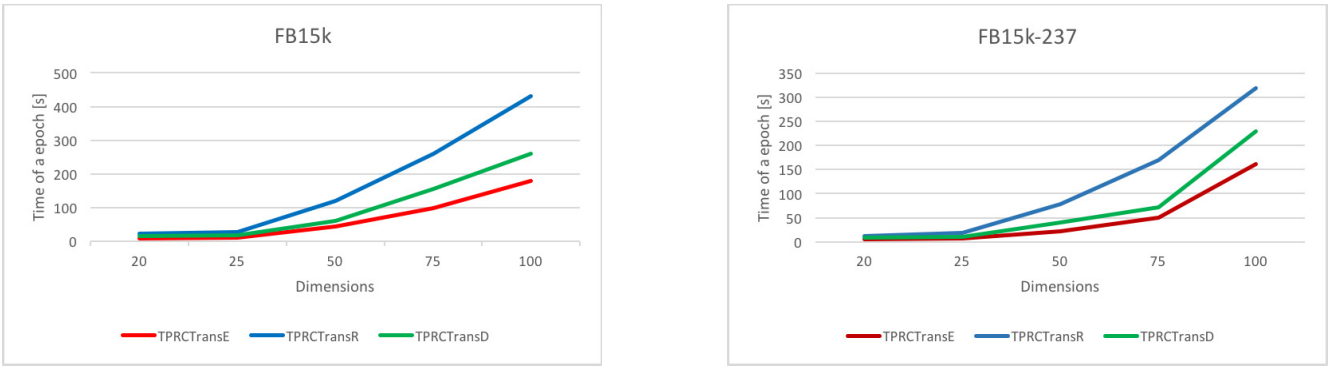

Fig. 1 Calculation time of TPRC models on FB15k and FB15k-237 datasets.

Table 5 Relation-wise (head and tail have the different types) analysis on MRR

Relations	ComplEx	TransD	$\text{TPRC}_{\text{TransD}}$ (bern)
profession	0.801	0.799	**0.817**
written_by	**0.637**	0.636	0.636
film_in_this_genre	0.739	0.734	**0.751**
olympics_participated_in	0.699	0.700	**0.719**
directed_by	0.681	0.679	**0.684**
films_production_designed	**0.574**	0.571	0.572
place_of_birth	0.721	0.714	**0.730**

Table 6 Relation-wise (head and tail have the same type) analysis on MRR.

Relations	ComplEx	TransD	$\text{TPRC}_{\text{TransD}}$ (bern)
includes_diseases	**1.000**	**1.000**	**1.000**
administrative_parent	**0.697**	0.600	0.614
sibling	**0.559**	0.557	**0.559**
spouse	**0.624**	5.436	5.433
is_part_of	**0.723**	**0.723**	0.702
award_nominee	0.758	0.742	**0.772**
children	**0.666**	0.615	0.647
parents	**0.453**	0.441	0.422

tion distance based models can work well with such datasets than bilinear models. Another interesting observation is the results of raw setting of translation distance based models is higher than the bilinear models; (4) From all the results, based on the good basic model TransD, the extended models of TransD can achieve the best performance compared with other state-of-the-art baselines TKRL, ComplEx, DistMult, TransE, TransR and TransD itself. Despite of very high time and space complexity, TKRL achieved very good results in FB15k and FB15k-237 datasets but it has failed to outperform ComplEx, DistMult, TransE, TransR and TransD. TKRL exploited very rich type information in their model. The advantage of TPRC over TKRL is: TPRC can learn KG embedding jointly with other state-of-the-art translation distanced based models with less parameters overhead. $\text{TPRC}_{\text{TransD}}$ (bern) achieved 0.846, 0.468 and 0.592 of Hits@10 for the datasets FB15k, FB15k-237 and YAGO3-10 respectively, which are 8.3%, 5.6% and 6.1% higher than the original TransD (bern). It also obtains very impressive results over ComplEx on FB15k-237 dataset.

Tables 5 and 6 show the relation-wise analysis for FB15k dataset for few relations. In our datasets some relations comprise with same type of head and tail entities and some have different types in the head and tail entity. Our model achieves a bit better results for the relations which have different types of head and tail entities than the other type relation. We observed that same-type entities tend to converge and form clusters, which can lead to errors in some cases. In the near future, we aim to develop a data-sampling algorithm to address this problem.

The calculation time of TPRC models are shown in Fig. 1. They are measured by using four cores in a 3.90 GHz processor. Theoretically, the time complexities of $\text{TPRC}_{\text{TransE}}$, $\text{TPRC}_{\text{TransR}}$ and $\text{TPRC}_{\text{TransD}}$ are same. $\text{TPRC}_{\text{TransE}}$ model took less amount of time than other two models. For FB15k dataset, $\text{TPRC}_{\text{TransE}}$, $\text{TPRC}_{\text{TransR}}$ and $\text{TPRC}_{\text{TransD}}$ took 179.7, 431.1 and 260.6 seconds respectively to complete one epoch when the dimension was 100. On the other hand, for FB15k-237 dataset, $\text{TPRC}_{\text{TransE}}$, $\text{TPRC}_{\text{TransR}}$ and $\text{TPRC}_{\text{TransD}}$ took 161.2, 321.8 and 230.2 seconds respectively to complete one epoch with same dimension. These models were trained for 1000 epochs for each dataset.

As noted, the entities' type plays a crucial role with respect to its relations. It is therefore logical that incorporating type information in relational context could be utilized to achieve better performance in the link-prediction task. We believe that the projection of entities based on the entities' type-mapping matrix would improve the performance of the proposed models. In these models, we are using entities' type information considering the relation it appers, in addition to the entities themselves, enabling the projected vectors to exhibit more semantic information than the vectors in TransE, TransR and TransD models.

5.3 Triple Classification

The triple classification task [24] checks whether a given triple (h, r, t) is correct or incorrect. When it was first introduced by Socher et al. in the NTN model, it acted as a binary classification. Because the task requires negative samples, we employed the "FB13" dataset, which is a benchmark dataset from Freebase involving 13 relations.

Table 7 Accuracy of the triple classification task on FB13 dataset. The results of SLM, NTN, SE, TransE, TransR and TransD were reported by Ji et al. [15].

Datasets	Accuracy (%)
SLM	85.3
NTN	87.1
SE	75.2
TransE (unif)	70.9
TransE (bern)	81.5
TransR (unif)	74.7
TransR (bern)	82.5
TransD (unif)	85.9
TransD (bern)	89.1
$\text{TPRC}_{\text{TransE}}$ (unif)	75.3
$\text{TPRC}_{\text{TransE}}$ (bern)	85.0
$\text{TPRC}_{\text{TransR}}$ (unif)	81.6
$\text{TPRC}_{\text{TransR}}$ (bern)	88.8
$\text{TPRC}_{\text{TransD}}$ (unif)	87.6
$\text{TPRC}_{\text{TransD}}$ (bern)	**89.9**

Table 8 Parameter settings of FB13.

Models	α	B	γ	d	$D.S$
$\text{TPRC}_{\text{TransE}}$	0.001	50	1	100	L1
$\text{TPRC}_{\text{TransR}}$	0.001	300	1	100	L1
$\text{TPRC}_{\text{TransD}}$	0.0005	1440	2	100	L2

Experimental Protocol: The experimental setup for triple classification task is very simple. To implement this task, we set a relation-specific threshold σ_r. For a triple (h, r, t), if the dissimilarity score (computed by the scoring function f_r) is below the σ_r threshold, then the predicted triple is positive. Otherwise, the prediction is negative. The value for σ_r is determined in accordance with the classification accuracy.

Optimization and Results: Table 7 shows the evaluation results for triple classification. The parameters and evaluation results for SLM, NTN, SE, TransE, TransR, TransD were obtained directly from the paper [15]. The parameter values for training $\text{TPRC}_{\text{TransE}}$, $\text{TPRC}_{\text{TransR}}$ and $\text{TPRC}_{\text{TransD}}$ are shown in Table 8.

We see from the Table 7 that $\text{TPRC}_{\text{TransD}}$ (bern) achieved the best performance. $\text{TPRC}_{\text{TransE}}$ is more accurate than TransE, $\text{TPRC}_{\text{TransR}}$ is more accurate than TransR and $\text{TPRC}_{\text{TransR}}$ outperform TransD. These results imply that incorporating type information in the relation context can improve the model accuracy. Papers of the bilinear/semantic matching models (RESCAL, DistMult and ComplEx) usually don't include triple classification task. So, we report the results of translational models and neural network based models.

6. Conclusion

In this paper, we have proposed an embedding model that leverages entity-type properties in the relational context. The strengths of this model are: it can produce find-grained representation of entities considering the entity types in relational context and it can be combined easily with other translation-distance-based models to improve accuracy without making the models more complex. The TPRC model is conceptually simple and can demonstrate highly competitive results for link prediction and triple classification. The underlying idea of TPRC was applied to the most popular translation-distance-based models (TransE, TransR and TransD) with the experimental results showing better performances than for the basic TransE, TransR, and TransD models along with ComplEx and DistMult. We can therefore conclude that the entities' type-based diversity in relations in a KG is an important factor and that the entities' type-mapping matrix in relational context is suitable for modeling KGs. In the future, we will utilize more sophisticated models to leverage entity types information. We also intend to use entity type ranking information in relational context and perform experiments on a wider variety of datasets.

Acknowledgments

This work was supported by Cross-ministerial Strategic Innovation Promotion Program (SIP) Second Phase, "Big-data and AI-enabled Cyberspace Technologies" by New Energy and Industrial Technol- ogy Development Organization (NEDO).

References

[1] D. Vrandečić and M. Krötzsch, "Wikidata: a free collaborative knowledge base," vol.57, no.10, pp.78–85, 2014.
[2] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P.N. Mendes, S. Hellmann, M. Morsey, P. van Kleef, S. Auer, and C. Bizer, "Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia," Semantic Web, vol.6, no.2, pp.167–195, 2015.
[3] F. Mahdisoltani, J. Biega, and F.M. Suchanek, "Yago3: A knowledge base from multilingual wikipedias," In the Proceedings of CIDR, 2015.
[4] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, "Freebase: a collaboratively created graph database for structuring human knowledge," Proceedings of the 2008 ACM SIGMOD international conference on Management of data, pp.1247–1250, AcM, 2008.
[5] F. Zhang, N.J. Yuan, D. Lian, X. Xie, and W.-Y. Ma, "Collaborative knowledge base embedding for recommender systems," Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, pp.353–362, ACM, 2016.
[6] A. Fader, L. Zettlemoyer, and O. Etzioni, "Open question answering over curated and extracted knowledge bases," Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, pp.1156–1165, ACM, 2014.
[7] E. Palumbo, G. Rizzo, and R. Troncy, "Entity2rec: Learning user-item relatedness from knowledge graphs for top-n item recommendation," Proceedings of the Eleventh ACM Conference on Recommender Systems, pp.32–36, ACM, 2017.
[8] Y. Cao, X. Wang, X. He, Z. Hu, and T.-S. Chua, "Unifying knowledge graph learning and recommendation: Towards a better understanding of user preferences," The World Wide Web Conference, pp.151–161, ACM, 2019.
[9] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," Journal of machine learning research, vol.3, no.Feb, pp.1137–1155, 2003.

[10] T. Mikolov, I. Sutskever, K. Chen, G.S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," Advances in neural information processing systems, pp.3111–3119, 2013.
[11] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," Advances in neural information processing systems, pp.2787–2795, 2013.
[12] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," arXiv preprint arXiv:1301.3781, 2013.
[13] Z. Wang, J. Zhang, J. Feng, and Z. Chen, "Knowledge graph embedding by translating on hyperplanes," AAAI, pp.1112–1119, 2014.
[14] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu, "Learning entity and relation embeddings for knowledge graph completion," AAAI, pp.2181–2187, 2015.
[15] G. Ji, S. He, L. Xu, K. Liu, and J. Zhao, "Knowledge graph embedding via dynamic mapping matrix," Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pp.687–696, 2015.
[16] M.M. Rahman and A. Takasu, "Knowledge graph embedding via entities' type mapping matrix," International Conference on Neural Information Processing, vol.11303, pp.114–125, Springer, 2018.
[17] A. Bordes, X. Glorot, J. Weston, and Y. Bengio, "Joint learning of words and meaning representations for open-text semantic parsing," Artificial Intelligence and Statistics, pp.127–135, 2012.
[18] A. Bordes, X. Glorot, J. Weston, and Y. Bengio, "A semantic matching energy function for learning with multi-relational data," Machine Learning, vol.94, no.2, pp.233–259, 2014.
[19] A. Bordes, J. Weston, R. Collobert, Y. Bengio, et al., "Learning structured embeddings of knowledge bases," AAAI, p.6, 2011.
[20] M. Nickel, V. Tresp, and H.P. Kriegel, "A three-way model for collective learning on multi-relational data," ICML, pp.809–816, 2011.
[21] M. Nickel, V. Tresp, and H.-P. Kriegel, "Factorizing yago: scalable machine learning for linked data," Proceedings of the 21st international conference on World Wide Web, pp.271–280, ACM, 2012.
[22] B. Yang, W.t. Yih, X. He, J. Gao, and L. Deng, "Embedding entities and relations for learning and inference in knowledge bases," arXiv preprint arXiv:1412.6575, 2014.
[23] T. Trouillon, C.R. Dance, É. Gaussier, J. Welbl, S. Riedel, and G. Bouchard, "Knowledge graph completion via complex tensor factorization," The Journal of Machine Learning Research, vol.18, no.1, pp.4735–4772, 2017.
[24] R. Socher, D. Chen, C.D. Manning, and A. Ng, "Reasoning with neural tensor networks for knowledge base completion," Advances in neural information processing systems, pp.926–934, 2013.
[25] Q. Liu, H. Jiang, A. Evdokimov, Z.H. Ling, X. Zhu, S. Wei, and Y. Hu, "Probabilistic reasoning via deep learning: Neural association models," arXiv preprint arXiv:1603.07704, 2016.
[26] T. Dettmers, P. Minervini, P. Stenetorp, and S. Riedel, "Convolutional 2d knowledge graph embeddings," Thirty-Second AAAI Conference on Artificial Intelligence, pp.1811–1818, 2018.
[27] X. Jiang, Q. Wang, and B. Wang, "Adaptive convolution for multi-relational learning," Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pp.978–987, 2019.
[28] S. Guo, Q. Wang, B. Wang, L. Wang, and L. Guo, "Semantically smooth knowledge graph embedding," Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pp.84–94, 2015.
[29] S. Guo, Q. Wang, B. Wang, L. Wang, and L. Guo, "Sse: Semantically smooth embedding for knowledge graphs," IEEE Trans. Knowl. Data Eng., vol.29, no.4, pp.884–897, 2017.
[30] M. Belkin and P. Niyogi, "Laplacian eigenmaps and spectral techniques for embedding and clustering," Advances in neural information processing systems, pp.585–591, 2002.
[31] S.T. Roweis and L.K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," science, vol.290, no.5500, pp.2323–2326, 2000.
[32] R. Xie, Z. Liu, and M. Sun, "Representation learning of knowledge graphs with hierarchical types," IJCAI, pp.2965–2971, 2016.
[33] D. Krompaß, S. Baier, and V. Tresp, "Type-constrained representation learning in knowledge graphs," International Semantic Web Conference, vol.9366, pp.640–655, Springer, 2015.
[34] M. Nickel, L. Rosasco, and T. Poggio, "Holographic embeddings of knowledge graphs," Thirtieth Aaai conference on artificial intelligence, pp.1955–1961, 2016.
[35] K. Toutanova, D. Chen, P. Pantel, H. Poon, P. Choudhury, and M. Gamon, "Representing text for joint embedding of text and knowledge bases," Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pp.1499–1509, 2015.

Md Mostafizur Rahman received his B.S. and M.S. degrees in Computer Science and Engineering from University of Dhaka, Bangladesh in 2010 and 2012, respectively. During 2012–2015, he worked in Samsung Electronics Ltd. He is currently a Ph.D candidate at SOKENDAI (The Graduate University for Advanced Studies), Japan.

Atsuhiro Takasu received B.E., M.E. and Dr. Eng. from the University of Tokyo in 1984,1986 and 1989, respectively. He is a professor of National Institute of Informatics, Japan. His research interests are data engineering and data mining. He is a member of ACM, IEEE, IEICE, IPSJ and JSAI.