Knowledge Graph Embedding via Dynamic Mapping Matrix

Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu and Jun Zhao
National Laboratory of Pattern Recognition (NLPR)
Institute of Automation Chinese Academy of Sciences, Beijing, 100190, China
{guoliang.ji,shizhu.he,lhxu,kliu,jzhao}@nlpr.ia.ac.cn

Abstract

Knowledge graphs are useful resources for numerous AI applications, but they are far from completeness. Previous work such as TransE, TransH and TransR/CTransR regard a relation as translation from head entity to tail entity and the CTransR achieves state-of-the-art performance. In this paper, we propose a more fine-grained model named TransD, which is an improvement of TransR/CTransR. In TransD, we use two vectors to represent a named symbol object (entity and relation). The first one represents the meaning of a(n) entity (relation), the other one is used to construct mapping matrix dynamically. Compared with TransR/CTransR, TransD not only considers the diversity of relations, but also entities. TransD has less parameters and has no matrix-vector multiplication operations, which makes it can be applied on large scale graphs. In Experiments, we evaluate our model on two typical tasks including triplets classification and link prediction. Evaluation results show that our approach outperforms state-of-the-art methods.

1 Introduction

Knowledge Graphs such as WordNet (Miller 1995), Freebase (Bollacker et al. 2008) and Yago (Suchanek et al. 2007) have been playing a pivotal role in many AI applications, such as relation extraction(RE), question answering(Q&A), etc. They usually contain huge amounts of structured data as the form of triplets (*head entity*, *relation*, *tail entity*)(denoted as (h, r, t)), where relation models the relationship between the two entities. As most knowledge graphs have been built either collaboratively or (partly) automatically, they often suffer from incompleteness. Knowledge graph completion is to predict relations between entities based on existing triplets in a knowledge graph. In the past decade, much work based on symbol and logic has been done for knowledge graph completion, but they are neither tractable nor enough convergence for large scale knowledge graphs. Recently, a powerful approach for this task is to encode every element (entities and relations) of a knowledge graph into a low-dimensional embedding vector space. These methods do reasoning over knowledge graphs through algebraic operations (see section "Related Work").

Among these methods, TransE (Bordes et al. 2013) is simple and effective, and also achieves state-of-the-art prediction performance. It learns low-dimensional embeddings for every entity and relation in knowledge graphs. These vector embeddings are denoted by the same letter in boldface. The basic idea is that every relation is regarded as translation in the embedding space. For a golden triplet (h, r, t), the embedding $\mathbf{h}$ is close to the embedding $\mathbf{t}$ by adding the embedding $\mathbf{r}$, that is $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$. TransE is suitable for 1-to-1 relations, but has flaws when dealing with 1-to-N, N-to-1 and N-to-N relations. TransH (Wang et al. 2014) is proposed to solve these issues. TransH regards a relation as a translating operation on a relation-specific hyperplane, which is characterized by a norm vector $\mathbf{w}_r$ and a translation vector $\mathbf{d}_r$. The embeddings $\mathbf{h}$ and $\mathbf{t}$ are first projected to the hyperplane of relation r to obtain vectors $\mathbf{h}_\perp = \mathbf{h} - \mathbf{w}_r^\top \mathbf{h} \mathbf{w}_r$ and $\mathbf{t}_\perp = \mathbf{t} - \mathbf{w}_r^\top \mathbf{t} \mathbf{w}_r$, and then $\mathbf{h}_\perp + \mathbf{d}_r \approx \mathbf{t}_\perp$. Both in TransE and TransH, the embeddings of entities and relations are in the same space. However, entities and relations are different types of objects, it is insufficient to model them in the same space. TransR/CTransR (Lin et al. 2015) set a mapping matrix $\mathbf{M}_r$ and a vector $\mathbf{r}$ for every relation r. In TransR, $\mathbf{h}$ and $\mathbf{t}$ are projected to the aspects that relation r focuses on through the matrix $\mathbf{M}_r$

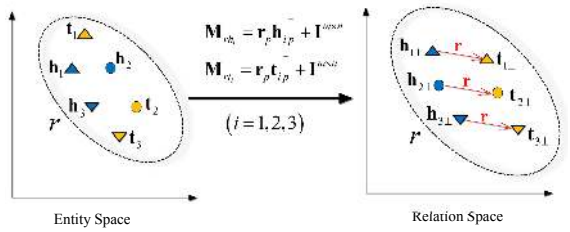

Figure 1: Simple illustration of TransD. Each shape represents an entity pair appearing in a triplet of relation r. $\mathbf{M}_{rh}$ and $\mathbf{M}_{rt}$ are mapping matrices of $\mathbf{h}$ and $\mathbf{t}$, respectively. $\mathbf{h}_{ip}$, $\mathbf{t}_{ip}(i = 1, 2, 3)$, and $\mathbf{r}_p$ are projection vectors. $\mathbf{h}_{i\perp}$ and $\mathbf{t}_{i\perp}(i = 1, 2, 3)$ are projected vectors of entities. The projected vectors satisfy $\mathbf{h}_{i\perp} + \mathbf{r} \approx \mathbf{t}_{i\perp}(i = 1, 2, 3)$.

and then $\mathbf{M}_r\mathbf{h} + \mathbf{r} \approx \mathbf{M}_r\mathbf{t}$. CTransR is an extension of TransR by clustering diverse head-tail entity pairs into groups and learning distinct relation vectors for each group. TransR/CTransR has significant improvements compared with previous state-of-the-art models. However, it also has several flaws: (1) For a typical relation r, all entities share the same mapping matrix $\mathbf{M}_r$. However, the entities linked by a relation always contain various types and attributes. For example, in triplet (*friedrich_burklein, nationality, germany*), *friedrich_burklein* and *germany* are typical different types of entities. These entities should be projected in different ways; (2) The projection operation is an interactive process between an entity and a relation, it is unreasonable that the mapping matrices are determined only by relations; and (3) Matrix-vector multiplication brings large amount of calculation, and when relation number is large, it also has much more parameters than TransE and TransH. As the complexity, TransR/CTransR is difficult to apply on large-scale knowledge graphs.

In this paper, we propose a novel method named TransD to model knowledge graphs. Figure 1 shows the basic idea of TransD. In TransD, we define *two* vectors for each entity and relation. The first vector represents the meaning of an entity or a relation, the other one (called *projection vector*) represents the way that how to project a entity embedding into a relation vector space and it will be used to construct mapping matrices. Therefore, every entity-relation pair has an unique mapping matrix. In addition, TransD has no matrix-by-vector operations which can be replaced by vectors operations. We evaluate TransD with the task of triplets classification and link prediction. The experimental results show that our method has significant improvements compared with previous models.

Our contributions in this paper are: (1)We propose a novel model TransD, which constructs a dynamic mapping matrix for each entity-relation pair by considering the diversity of entities and relations simultaneously. It provides a flexible style to project entity representations to relation vector space; (2) Compared with TransR/CTransR, TransD has fewer parameters and has no matrix-vector multiplication. It is easy to be applied on large-scale knowledge graphs like TransE and TransH; and (3) In experiments, our approach outperforms previous models including TransE, TransH and TransR/CTransR in link prediction and triplets classification tasks.

2 Related Work

Before proceeding, we define our mathematical notations. We denote a triplet by (h, r, t) and their column vectors by bold lower case letters $\mathbf{h}, \mathbf{r}, \mathbf{t}$; matrices by bold upper case letters, such as $\mathbf{M}$; tensors by bold upper case letters with a hat, such as $\widehat{\mathbf{M}}$. Score function is represented by $f_r(\mathbf{h}, \mathbf{t})$. For a golden triplet (h, r, t) that corresponds to a true fact in real world, it always get a relatively higher score, and lower for an negative triplet. Other notations will be described in the appropriate sections.

2.1 TransE, TransH and TransR/CTransR

As mentioned in Introduction section, TransE (Bordes et al. 2013) regards the relation $\mathbf{r}$ as translation from $\mathbf{h}$ to $\mathbf{t}$ for a golden triplet (h, r, t). Hence, $(\mathbf{h}+\mathbf{r})$ is close to $(\mathbf{t})$ and the score function is

$$f_r(\mathbf{h}, \mathbf{t}) = -\|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_2^2. \quad (1)$$

TransE is only suitable for 1-to-1 relations, there remain flaws for 1-to-N, N-to-1 and N-to-N relations.

To solve these problems, TransH (Wang et al. 2014) proposes an improved model named translation on a hyperplane. On hyperplanes of different relations, a given entity has different representations. Similar to TransE, TransH has the score function as follows:

$$f_r(\mathbf{h}, \mathbf{t}) = -\|\mathbf{h}_\perp + \mathbf{r} - \mathbf{t}_\perp\|_2^2. \quad (2)$$

Model	#Parameters	# Operations (Time complexity)
Unstructured (Bordes et al. 2012; 2014)	$O(N_e m)$	$O(N_t)$
SE (Bordes et al. 2011)	$O(N_e m + 2N_r n^2)(m = n)$	$O(2m^2 N_t)$
SME(linear) (Bordes et al. 2012; 2014)	$O(N_e m + N_r n + 4mk + 4k)(m = n)$	$O(4mkN_t)$
SME (bilinear) (Bordes et al. 2012; 2014)	$O(N_e m + N_r n + 4mks + 4k)(m = n)$	$O(4mksN_t)$
LFM (Jenatton et al. 2012; Sutskever et al. 2009)	$O(N_e m + N_r n^2)(m = n)$	$O((m^2 + m)N_t)$
SLM (Socher et al. 2013)	$O(N_e m + N_r(2k + 2nk))(m = n)$	$O((2mk + k)N_t)$
NTN (Socher et al. 2013)	$O(N_e m + N_r(n^2 s + 2ns + 2s))(m = n)$	$O(((m^2 + m)s + 2mk + k)N_t)$
TransE (Bordes et al. 2013)	$O(N_e m + N_r n)(m = n)$	$O(N_t)$
TransH (Wang et al. 2014)	$O(N_e m + 2N_r n)(m = n)$	$O(2mN_t)$
TransR (Lin et al. 2015)	$O(N_e m + N_r(m + 1)n)$	$O(2mnN_t)$
CTransR (Lin et al. 2015)	$O(N_e m + N_r(m + d)n)$	$O(2mnN_t)$
TransD (this paper)	$O(2N_e m + 2N_r n)$	$O(2nN_t)$

Table 1: Complexity (the number of parameters and the number of multiplication operations in an epoch) of several embedding models. 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 the dimension of relation embedding space. d denotes the average number of clusters of a relation. k is the number of hidden nodes of a neural network and s is the number of slice of a tensor.

In order to ensure that $\mathbf{h}_\perp$ and $\mathbf{t}_\perp$ are on the hyperplane of r, TransH restricts $\|\mathbf{w}_r\| = 1$.

Both TransE and TransH assume that entities and relations are in the same vector space. But relations and entities are different types of objects, they should not be in the same vector space. TransR/CTransR (Lin et al. 2015) is proposed based on the idea. TransR set a mapping matrix $\mathbf{M}_r$ for each relation r to map entity embedding into relation vector space. Its score function is:

$$f_r(\mathbf{h}, \mathbf{t}) = -\|\mathbf{M}_r\mathbf{h} + \mathbf{r} - \mathbf{M}_r\mathbf{t}\|_2^2. \quad (3)$$

where $\mathbf{M}_r \in \mathbb{R}^{m\times n}$, $\mathbf{h}, \mathbf{t} \in \mathbb{R}^n$ and $\mathbf{r} \in \mathbb{R}^m$. CTransR is an extension of TransR. As head-tail entity pairs present various patterns in different relations, CTransR clusters diverse head-tail entity pairs into groups and sets a relation vector for each group.

2.2 Other Models

Unstructured. Unstructured model (Bordes et al. 2012; 2014) ignores relations, only models entities as embeddings. The score function is

$$f_r(\mathbf{h}, \mathbf{t}) = -\|\mathbf{h} - \mathbf{t}\|_2^2. \quad (4)$$

It's a simple case of TransE. Obviously, Unstructured model can not distinguish different relations.

Structured Embedding (SE). SE model (Bordes et al. 2011) sets two separate matrices $\mathbf{M}_{rh}$ and $\mathbf{M}_{rt}$ to project head and tail entities for each relation. Its score function is defined as follows:

$$f_r(\mathbf{h}, \mathbf{t}) = -\|\mathbf{M}_{rh}\mathbf{h} - \mathbf{M}_{rt}\mathbf{t}\|_1 \quad (5)$$

Semantic Matching Energy (SME). SME model (Bordes et al. 2012; 2014) encodes each named symbolic object (entities and relations) as a vector. Its score function is a neural network that captures correlations between entities and relations via matrix operations. Parameters of the neural network are shared by all relations. SME defines two semantic matching energy functions for optimization, a linear form

$$\mathbf{g}_\eta = \mathbf{M}_{\eta 1}\mathbf{e}_\eta + \mathbf{M}_{\eta 2}\mathbf{r} + \mathbf{b}_\eta \quad (6)$$

and a bilinear form

$$\mathbf{g}_\eta = (\mathbf{M}_{\eta 1}\mathbf{e}_\eta) \otimes (\mathbf{M}_{\eta 2}\mathbf{r}) + \mathbf{b}_\eta \quad (7)$$

where $\eta = \{left, right\}$, $\mathbf{e}_{left} = \mathbf{h}$, $\mathbf{e}_{right} = \mathbf{t}$ and $\otimes$ is the Hadamard product. The score function is

$$f_r(\mathbf{h}, \mathbf{t}) = \mathbf{g}_{left}{}^\top \mathbf{g}_{right} \quad (8)$$

In (Bordes et al.2014), matrices of the bilinear form are replaced by tensors.

Latent Factor Model (LFM). LFM model (Jenatton et al. 2012; Sutskever et al. 2009) encodes each entity into a vector and sets a matrix for every relation. It defines a score function $f_r(\mathbf{h}, \mathbf{t}) = \mathbf{h}^\top \mathbf{M}_r \mathbf{t}$, which incorporates the interaction of the two entity vectors in a simple and effecitve way.

Single Layer Model (SLM). SLM model is designed as a baseline of Neural Tensor Network (Socher et al. 2013). The model constructs a nonlinear neural network to represent the score function defined as follows.

$$f_r(\mathbf{h}, \mathbf{t}) = \mathbf{u}_r^\top f(\mathbf{M}_{r1}\mathbf{h} + \mathbf{M}_{r2}\mathbf{t} + \mathbf{b}_r) \quad (9)$$

where $\mathbf{M}_{r1}$, $\mathbf{M}_{r2}$ and $\mathbf{b}_r$ are parameters indexed by relation r, $f()$ is $tanh$ operation.

Neural Tensor Network (NTN). NTN model (Socher et al. 2013) extends SLM model by considering the second-order correlations into nonlinear neural networks. The score function is

$$f_r(\mathbf{h},\mathbf{t}) = \mathbf{u}_r^\top f(\mathbf{h}^\top \widehat{\mathbf{W}}_r \mathbf{t} + \mathbf{M}_r \begin{bmatrix} \mathbf{h} \\ \mathbf{t} \end{bmatrix} + \mathbf{b}_r) \quad (10)$$

where $\widehat{\mathbf{W}}_r$ represents a 3-way tensor, $\mathbf{M}_r$ denotes the weight matrix, $\mathbf{b}_r$ is the bias and $f()$ is $tanh$ operation. NTN is the most expressive model so far, but it has so many parameters that it is difficult to scale up to large knowledge graphs.

Table 1 lists the complexity of all the above models. The complexity (especially for time) of TransD is much less than TransR/CTransR and is similar to TransE and TransH. Therefore, TransD is effective and train faster than TransR/CTransR. Beyond these embedding models, there is other related work of modeling multi-relational data, such as matrix factorization, recommendations, etc. In experiments, we refer to the results of **RESCAL** presented in (Lin et al. 2015) and compare with it.

3 Our Method

We first define notations. Triplets are represented as $(h_i, r_i, t_i)(i = 1, 2, \ldots, n_t)$, where h_i denotes a head entity, t_i denotes a tail entity and r_i denotes a relation. Their embeddings are denoted by $\mathbf{h}_i, \mathbf{r}_i, \mathbf{t}_i (i = 1, 2, \ldots, n_t)$. We use Δ to represent golden triplets set, and use $\Delta^{'}$ to denote negative triplets set. Entities set and relations set are denoted by E and R, respectively. We use $\mathbf{I}^{m\times n}$ to denote the identity matrix of size $m \times n$.

3.1 Multiple Types of Entities and Relations

Considering the diversity of relations, CTransR segments triplets of a specific relation r into several groups and learns a vector representation for each group. However, entities also have various types. Figure 2 shows several kinds of head and tail entities of relation *location.location.partially_containedby* in FB15k. In both TransH and TransR/CTransR, all types of entities share the same mapping vectors/matrices. However, different types of entities have different attributes and functions, it is insufficient to let them share the same transform parameters of a relation. And for a given relation, similar entities should have similar mapping matrices and otherwise for dissimilar entities. Furthermore, the mapping process is a transaction between entities and relations that both have various types. Therefore, we propose a more fine-grained model TransD, which considers different types of both entities and relations, to encode knowledge graphs into embedding vectors via dynamic mapping matrices produced by projection vectors.

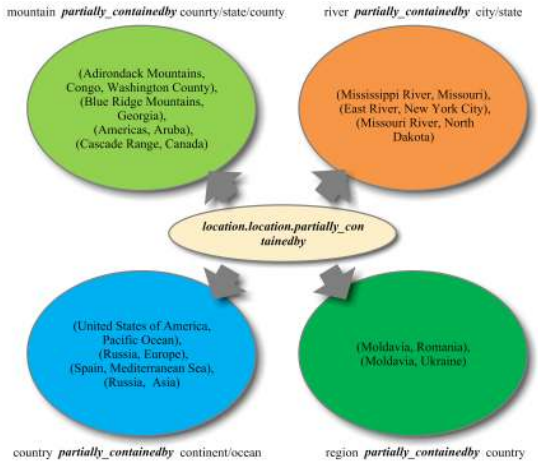

Figure 2: Multiple types of entities of relation *location.location.partially_containedby*.

3.2 TransD

Model In TransD, each named symbol object (entities and relations) is represented by *two* vectors. The first one captures the meaning of entity (relation), the other one is used to construct mapping matrices. For example, given a triplet (h, r, t), its vectors are $\mathbf{h}, \mathbf{h}_p, \mathbf{r}, \mathbf{r}_p, \mathbf{t}, \mathbf{t}_p$, where subscript p marks the projection vectors, $\mathbf{h}, \mathbf{h}_p, \mathbf{t}, \mathbf{t}_p \in \mathbb{R}^n$ and $\mathbf{r}, \mathbf{r}_p \in \mathbb{R}^m$. For each triplet (h, r, t), we set two mapping matrices $\mathbf{M}_{rh}, \mathbf{M}_{rt} \in \mathbb{R}^{m\times n}$ to project entities from entity space to relation space. They are defined as follows:

$$\mathbf{M}_{rh} = \mathbf{r}_p \mathbf{h}_p^\top + \mathbf{I}^{m\times n} \quad (11)$$

$$\mathbf{M}_{rt} = \mathbf{r}_p \mathbf{t}_p^\top + \mathbf{I}^{m\times n} \quad (12)$$

Therefore, the mapping matrices are determined by both entities and relations, and this kind of operation makes the two projection vectors interact sufficiently because each element of them can meet every entry comes from another vector. As we initialize each mapping matrix with an identity matrix, we add the $\mathbf{I}^{m\times n}$ to $\mathbf{M}_{rh}$ and $\mathbf{M}_{rh}$. With the mapping matrices, we define the projected vectors as follows:

$$\mathbf{h}_\perp = \mathbf{M}_{rh}\mathbf{h}, \quad \mathbf{t}_\perp = \mathbf{M}_{rt}\mathbf{t} \quad (13)$$

Then the score function is

$$f_r(\mathbf{h},\mathbf{t}) = -\|\mathbf{h}_\perp + \mathbf{r} - \mathbf{t}_\perp\|_2^2 \quad (14)$$

In experiments, we enforce constrains as $\|\mathbf{h}\|_2 \leq 1, \|\mathbf{t}\|_2 \leq 1, \|\mathbf{r}\|_2 \leq 1, \|\mathbf{h}_\perp\|_2 \leq 1$ and $\|\mathbf{t}_\perp\|_2 \leq 1$.

Training Objective We assume that there are n_t triplets in training set and denote the ith triplet by $(h_i, r_i, t_i)(i = 1, 2, \ldots, n_t)$. Each triplet has a label y_i to indicate the triplet is positive ($y_i = 1$) or negative ($y_i = 0$). Then the golden and negative triplets are denoted by $\Delta = \{(h_j, r_j, t_j) \mid y_j = 1\}$ and $\Delta^{'} = \{(h_j, r_j, t_j) \mid y_j = 0\}$, respectively. Before training, one important trouble is that knowledge graphs only encode positive training triplets, they do not contain negative examples. Therefore, we obtain Δ from knowledge graphs and generate $\Delta^{'}$ as follows: $\Delta^{'} = \{(h_l, r_k, t_k) \mid h_l \neq h_k \wedge y_k = 1\} \cup \{(h_k, r_k, t_l) \mid t_l \neq t_k \wedge y_k = 1\}$. We also use two strategies "unif" and "bern" described in (Wang et al. 2014) to replace the head or tail entity.

Let us use ξ and $\xi^{'}$ to denote a golden triplet and a corresponding negative triplet, respectively. Then we define the following margin-based ranking loss as the objective for training:

$$L = \sum_{\xi \in \Delta} \sum_{\xi' \in \Delta'} [\gamma + f_r(\xi^{'}) - f_r(\xi)]_+ \quad (15)$$

where $[x]_+ \triangleq \max(0, x)$, and γ is the margin separating golden triplets and negative triplets. The process of minimizing the above objective is carried out with stochastic gradient descent (SGD). In order to speed up the convergence and avoid overfitting, we initiate the entity and relation embeddings with the results of TransE and initiate all the transfer matrices with identity matrices.

3.3 Connections with TransE, TransH and TransR/CTransR

TransE is a special case of TransD when the dimension of vectors satisfies $m = n$ and all projection vectors are set zero.

TransH is related to TransD when we set $m = n$. Under the setting, projected vectors of entities can be rewritten as follows:

$$\mathbf{h}_\perp = \mathbf{M}_{rh}\mathbf{h} = \mathbf{h} + \mathbf{h}_p^\top \mathbf{h}\mathbf{r}_p \quad (16)$$

$$\mathbf{t}_\perp = \mathbf{M}_{rt}\mathbf{t} = \mathbf{t} + \mathbf{t}_p^\top \mathbf{t}\mathbf{r}_p \quad (17)$$

Hence, when $m = n$, the difference between TransD and TransH is that projection vectors are determinded only by relations in TransH, but TransD's projection vectors are determinded by both entities and relations.

As to TransR/CTransR, TransD is an improvement of it. TransR/CTransR directly defines a mapping matrix for each relation, TransD consturcts two mapping matrices dynamically for each triplet by setting a projection vector for each entity and relation. In addition, TransD has no matrix-vector multiplication operation which can be replaced by vector operations. Without loss of generality, we assume $m \geq n$, the projected vectors can be computed as follows:

$$\mathbf{h}_\perp = \mathbf{M}_{rh}\mathbf{h} = \mathbf{h}_p^\top \mathbf{h}\mathbf{r}_p + \left[\ \mathbf{h}^\top, \mathbf{0}^\top\ \right]^\top \quad (18)$$

$$\mathbf{t}_\perp = \mathbf{M}_{rt}\mathbf{t} = \mathbf{t}_p^\top \mathbf{t}\mathbf{r}_p + \left[\ \mathbf{t}^\top, \mathbf{0}^\top\ \right]^\top \quad (19)$$

Therefore, TransD has less calculation than TransR/CTransR, which makes it train faster and can be applied on large-scale knowledge graphs.

4 Experiments and Results Analysis

We evaluate our apporach on two tasks: triplets classification and link prediction. Then we show the experiments results and some analysis of them.

4.1 Data Sets

Triplets classification and link prediction are implemented on two popular knowledge graphs: WordNet (Miller 1995) and Freebase (Bollacker et al. 2008). WordNet is a large lexical knowledge graph. Entities in WordNet are synonyms which express distinct concepts. Relations in WordNet are conceptual-semantic and lexical relations. In this paper, we use two subsets of WordNet: WN11 (Socher et al. 2013) and WN18 (Bordes et al. 2014). Freebase is a large collaborative knowledge base consists of a large number of the world facts, such as triplets (*anthony_asquith, location, london*) and (*nobuko_otowa, profession, actor*). We also use two subsets of Freebase: FB15k (Bordes et al. 2014) and FB13 (Socher et al. 2013). Table 2 lists statistics of the 4 datasets.

Dataset	#Rel	#Ent	#Train	#Valid	#Test
WN11	11	38,696	112,581	2,609	10,544
WN18	18	40,943	141,442	5,000	5,000
FB13	13	75,043	316,232	5908	23,733
FB15k	1,345	14,951	483,142	50,000	59,071

Table 2: Datesets used in the experiments.

4.2 Triplets Classification

Triplets classification aims to judge whether a given triplet (h, r, t) is correct or not, which is a binary classification task. Previous work (Socher et al. 2013; Wang et al. 2014; Lin et al. 2015) had explored this task. In this paper ,we use three datasets WN11, FB13 and FB15k to evaluate our approach. The test sets of WN11 and FB13 provided by (Socher et al. 2013) contain golden and negative triplets. As to FB15k, its test set only contains correct triplets, which requires us to construct negative triplets. In this parper, we construct negative triplets following the same setting used for FB13 (Socher et al. 2013).

For triplets classification, we set a threshold δ_r for each relation r. δ_r is obtained by maximizing the classification accuracies on the valid set. For a given triplet (h, r, t), if its score is larger than δ_r, it will be classified as positive, otherwise negative.

We compare our model with several previous embedding models presented in Related Work section. As we construct negative triplets for FB15k by ourselves, we use the codes of TransE, TransH and TransR/CTransR provied by (Lin et al. 2015) to evaluate the datasets instead of reporting the results of (Wang et al.2014; Lin et al. 2015) directly.

In this experiment, we optimize the objective with ADADELTA SGD (Zeiler 2012). We select the margin γ among {1, 2, 5, 10}, the dimension of entity vectors m and the dimension of relation vectors n among {20, 50, 80, 100}, and the mini-batch size B among {100, 200, 1000, 4800}. The best configuration obtained by valid set are:$\gamma = 1, m, n = 100, B = 1000$ and taking L_2 as dissimilarity on WN11; $\gamma = 1, m, n = 100, B = 200$ and taking L_2 as dissimilarity on FB13; $\gamma = 2, m, n = 100, B = 4800$ and taking L_1 as dissimilarity on FB15k. For all the three datasets, We traverse to training for 1000 rounds. As described in Related Work section, TransD trains much faster than TransR (On our PC, TransR needs 70 seconds and TransD merely spends 24 seconds a round on FB15k).

Table 3 shows the evaluation results of triplets classification. On WN11, we found that there are 570 entities appearing in valid and test sets but not appearing in train set, we call them "NULL Entity". In valid and test sets, there are 1680 (6.4%) triplets containing "NULL Entity". In NTN(+E), these entity embeddings can be obtained by word embedding. In TransD, however, they are only initialized randomly. Therefore, it is not fair for TransD, but we also achieve the accuracy 86.4% which is higher than that of NTN(+E) (86.2%). From Table 3, we can conclude that: (1) On WN11, TransD outperforms any other previous models including TransE, TransH and TransR/CTransR, especially NTN(+E); (2) On FB13, the classification accuracy of TransD achieves 89.1%, which is significantly higher than that of TransE, TransH and TransR/CTransR and is near to the performance of NTN(+E) (90.0%); and (3) Under most circumstances, the "bern" sampling method works better than "unif".

Data sets	WN11	FB13	FB15K
SE	53.0	75.2	-
SME(bilinear)	70.0	63.7	-
SLM	69.9	85.3	-
LFM	73.8	84.3	-
NTN	70.4	87.1	68.2
NTN(+E)	86.2	**90.0**	-
TransE(unif)	75.9	70.9	77.3
TransE(bern)	75.9	81.5	79.8
TransH(unif)	77.7	76.5	74.2
TransH(bern)	78.8	83.3	79.9
TransR(unif)	85.5	74.7	81.1
TransR(bern)	85.9	82.5	82.1
CTransR(bern)	85.7	-	84.3
TransD(unif)	85.6	85.9	86.4
TransD(bern)	**86.4**	89.1	**88.0**

Table 3: Experimental results of Triplets Classification(%). "+E" means that the results are combined with word embedding.

Figure 3 shows the prediction accuracy of different relations. On the three datasets, different relations have different prediction accuracy: some are higher and the others are lower. Here we focus on the relations which have lower accuracy. On WN11, the relation *similar_to* obtains accuracy 51%, which is near to random prediction accuracy. In the view of intuition, *similar_to* can be inferred from other information. However, the number of entity pairs linked by relation *similar_to* is only 1672, which accounts for 1.5% in all train data, and prediction of the relation needs much information about entities. Therefore, the insufficient of train data is the main cause. On FB13, the accuracies of relations *cuase_of_death* and *gender* are lower than that of other relations because they are difficult to infer from other imformation, especially *cuase_of_death*. Relation *gender* may be inferred from a person's name (Socher et al. 2013), but we learn a vector for each name, not for the words included in the names, which makes the

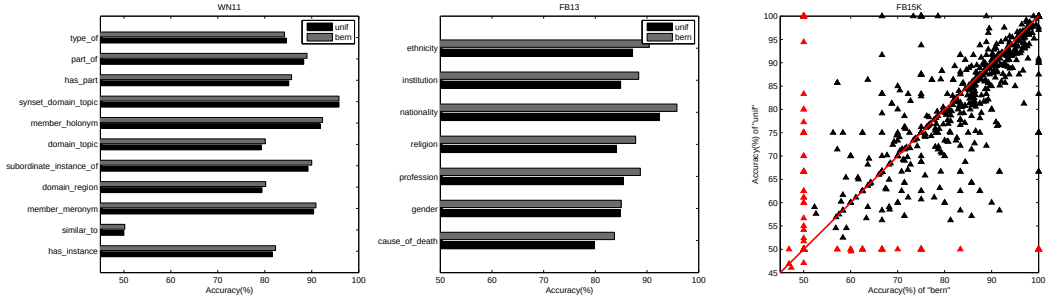

Figure 3: Classification accuracies of different relations on the three datasets. For FB15k, each triangle represent a relation, in which the red triangles represent the relations whose accuracies of "bern" or "unif" are lower than 50% and the blacks are higher than 50%. The red line represents the function $y = x$. We can see that the most relations are in the lower part of the red line.

names information useless for *gender*. On FB15k, accuracies of some relations are lower than 50%, for which some are lack of train data and some are difficult to infer. Hence, the ability of reasoning new facts based on knowledge graphs is under a certain limitation, and a complementary approach is to extract facts from plain texts.

4.3 Link Prediction

Link prediction is to predict the missing h or t for a golden triplet (h, r, t). In this task, we remove the head or tail entity and then replace it with all the entities in dictionary in turn for each triplet in test set. We first compute scores of those corrupted triplets and then rank them by descending order; the rank of the correct entity is finally stored. The task emphasizes the rank of the correct entity instead of only finding the best one entity. Similar to (Bordes et al. 2013), we report two measures as our evaluation metrics: the average rank of all correct entites (*Mean Rank*) and the proportion of correct entities ranked in top 10 (*Hits@10*). A lower *Mean Rank* and a higher *Hits@10* should be achieved by a good embedding model. We call the evaluation setting "Raw'. Noting the fact that a corrupted triplet may also exist in knowledge graphs, the corrupted triplet should be regard as a correct triplet. Hence, we should remove the corrupted triplets included in train, valid and test sets before ranking. We call this evaluation setting "Filter". In this paper, we will report evaluation results of the two settings .

In this task, we use two datasets: WN18 and FB15k. As all the data sets are the same, we refer to their experimental results in this paper. On WN18, we also use ADADELTA SGD (Zeiler 2012) for optimization. We select the margin γ among $\{0.1, 0.5, 1, 2\}$, the dimension of entity vectors m and the dimension of relation vectors n among $\{20, 50, 80, 100\}$, and the mini-batch size B among $\{100, 200, 1000, 1400\}$. The best configuration obtained by valid set are:$\gamma = 1, m, n = 50, B = 200$ and taking L_2 as dissimilarity. For both the two datasets, We traverse to training for 1000 rounds.

Experimental results on both WN18 and FB15k are shown in Table 4. From Table 4, we can conclude that: (1) TransD outperforms other baseline embedding models (TransE, TransH and TransR/CTransR), especially on sparse dataset, i.e., FB15k; (2) Compared with CTransR, TransD is a more fine-grained model which considers the multiple types of entities and relations simultaneously, and it achieves a better performance. It indicates that TransD handles complicated internal correlations of entities and relations in knowledge graphs better than CTransR; (3) The "bern" sampling trick can reduce false negative labels than "unif".

For the comparison of *Hits@10* of different kinds of relations, Table 5 shows the detailed results by mapping properties of relations[1] on FB15k. From Table 5, we can see that TransD outperforms TransE, TransH and TransR/CTransR significantly in both "unif" and "bern" settings. TransD achieves better performance than CTransR in all types of relations (1-to-1, 1-to-N, N-to-1 and N-to-N). For N-to-N relations in predicting both head and tail, our approach improves the *Hits@10* by almost 7.4% than CTransR. In particular, for

[1]Mapping properties of relations follows the same rules in (Bordes et al. 2013)

Data sets	WN18				FB15K			
Metric	Mean Rank		Hits@10		Mean Rank		Hits@10	
	Raw	Filt	Raw	Filt	Raw	Filt	Raw	Filt
Unstructured (Bordes et al. 2012)	315	304	35.3	38.2	1,074	979	4.5	6.3
RESCAL (Nickle, Tresp, and Kriegel 2011)	1,180	1,163	37.2	52.8	828	683	28.4	44.1
SE (Bordes et al. 2011)	1,011	985	68.5	80.5	273	162	28.8	39.8
SME (linear) (Bordes et al.2012)	545	533	65.1	74.1	274	154	30.7	40.8
SME (Bilinear) (Bordes et al. 2012)	526	509	54.7	61.3	284	158	31.3	41.3
LFM (Jenatton et al. 2012)	469	456	71.4	81.6	283	164	26.0	33.1
TransE (Bordes et al. 2013)	263	251	75.4	89.2	243	125	34.9	47.1
TransH (unif) (Wang et al. 2014)	318	303	75.4	86.7	211	84	42.5	58.5
TransH (bern) (Wang et al. 2014)	401	388	73.0	82.3	212	87	45.7	64.4
TransR (unif) (Lin et al. 2015)	232	219	78.3	91.7	226	78	43.8	65.5
TransR (bern) (Lin et al. 2015)	238	225	**79.8**	92.0	198	77	48.2	68.7
CTransR (unif) (Lin et al. 2015)	243	230	78.9	92.3	233	82	44.0	66.3
CTransR (bern) (Lin et al. 2015)	231	218	79.4	92.3	199	75	48.4	70.2
TransD (unif)	242	229	79.2	**92.5**	211	**67**	**49.4**	**74.2**
TransD (bern)	**224**	**212**	79.6	92.2	**194**	91	**53.4**	**77.3**

Table 4: Experimental results on link prediction.

Tasks	Prediction Head (Hits@10)				Prediction Tail (Hits@10)			
Relation Category	1-to-1	1-to-N	N-to-1	N-to-N	1-to-1	1-to-N	N-to-1	N-to-N
Unstructured (Bordes et al. 2012)	34.5	2.5	6.1	6.6	34.3	4.2	1.9	6.6
SE (Bordes et al. 2011)	35.6	62.6	17.2	37.5	34.9	14.6	68.3	41.3
SME (linear) (Bordes et al.2012)	35.1	53.7	19.0	40.3	32.7	14.9	61.6	43.3
SME (Bilinear) (Bordes et al. 2012)	30.9	69.6	19.9	38.6	28.2	13.1	76.0	41.8
TransE (Bordes et al. 2013)	43.7	65.7	18.2	47.2	43.7	19.7	66.7	50.0
TransH (unif) (Wang et al. 2014)	66.7	81.7	30.2	57.4	63.7	30.1	83.2	60.8
TransH (bern) (Wang et al. 2014)	66.8	87.6	28.7	64.5	65.5	39.8	83.3	67.2
TransR (unif) (Lin et al. 2015)	76.9	77.9	38.1	66.9	76.2	38.4	76.2	69.1
TransR (bern) (Lin et al. 2015)	78.8	89.2	34.1	69.2	79.2	37.4	90.4	72.1
CTransR (unif) (Lin et al. 2015)	78.6	77.8	36.4	68.0	77.4	37.8	78.0	70.3
CTransR (bern) (Lin et al. 2015)	81.5	89.0	34.7	71.2	80.8	38.6	90.1	73.8
TransD (unif)	80.7	85.8	**47.1**	**75.6**	80.0	**54.5**	80.7	**77.9**
TransD (bern)	**86.1**	**95.5**	**39.8**	**78.5**	**85.4**	**50.6**	**94.4**	**81.2**

Table 5: Experimental results on FB15K by mapping properities of relations (%).

N-to-1 relations (predicting head) and 1-to-N relations (predicting tail), TransD improves the accuracy by 9.0% and 14.7% compared with previous state-of-the-art results, respectively. Therefore, the diversity of entities and relations in knowledge grahps is an important factor and the dynamic mapping matrix is suitable for modeling knowledge graphs.

5 Properties of Projection Vectors

As mentioned in Section "Introduction", TransD is based on the motivation that each mapping matrix is determined by entity-relation pair dynamically. These mapping matrices are constructed with projection vectors of entities and relations. Here, we analysis the properties of projection vectors. We seek the similar objects (entities and relations) for a given object (entities and relations) by projection vectors. As WN18 has the most entities (40,943 entities which contains various types of words. FB13 also has many entities, but the most are person's names) and FB15k has the most relations (1,345 relations), we show the similarity of projection vectors on them. Table 6 and 7 show that the same category objects have similar projection vectors. The similarity of projection vectors of different types of entities and relations indicates the rationality of our method.

6 Conclusions and Future Work

We introduced a model TransD that embed knowledge graphs into continues vector space for their completion. TransD has less complexity and more flexibility than TransR/CTransR. When learning embeddings of named symbol objects (entities or relations), TransD considers the diversity of them both. Extensive experiments show that TransD outperforms TrasnE, TransH and TransR/CTransR on two tasks including triplets classification and link prediction.

As shown in Triplets Classification section, not all new facts can be deduced from the exist-

Datesets	WN18			
Entities and Definitions	*_upset_VB_4*	cause to overturn from an upright or normal position	*_srbija_NN_1*	a historical region in central and northern Yugoslavia
Similar Entities and Definitions	*_sway_VB_4*	cause to move back and forth	*_montenegro_NN_1*	a former country bordering on the Adriatic Sea
	_shift_VB_2	change place or direction	*_constantina_NN_1*	a Romanian resort city on the Black Sea
	_flap_VB_3	move with a thrashing motion	*_lappland_NN_1*	a region in northmost Europe inhabited by Lapps
	_fluctuate_VB_1	cause to fluctuate or move in a wave-like pattern	*_plattensee_NN_1*	a large shallow lake in western Hungary
	_leaner_NN_1	(horseshoes) the throw of a horseshoe so as to lean against (but not encircle) the stake	*_brasov_NN_1*	a city in central Romania in the foothills of the Transylvanian Alps

Table 6: Entity projection vectors similarity (in descending order) computed on WN18. The similarity scores are computed with $cosine$ function.

Datesets	FB15k
Relation	*/location/statistical_region/rent50_2./measurement_unit/dated_money_value/currency*
Similar relations	*/location/statistical_region/rent50_3./measurement_unit/dated_money_value/currency* */location/statistical_region/rent50_1./measurement_unit/dated_money_value/currency* */location/statistical_region/rent50_4./measurement_unit/dated_money_value/currency* */location/statistical_region/rent50_0./measurement_unit/dated_money_value/currency* */location/statistical_region/gdp_nominal./measurement_unit/dated_money_value/currency*
Relation	*/sports/sports_team/roster./soccer/football_roster_position/player*
Similar relations	*/soccer/football_team/current_roster./sports/sports_team_roster/player* */soccer/football_team/current_roster./soccer/football_roster_position/player* */sports/sports_team/roster./sports/sports_team_roster/player* */basketball/basketball_team/historical_roster./sports/sports_team_roster/player* */sports/sports_team/roster./basketball/basketball_historical_roster_position/player*

Table 7: Relation projection vectors similarity computed on FB15k. The similarity scores are computed with $cosine$ function.

ing triplets in knowledge graphs, such as relations *gender, place of place, parents* and *children*. These relations are difficult to infer from all other information, but they are also useful resource for practical applications and incomplete, i.e. the *place of birth* attribute is missing for 71% of all people included in FreeBase (Nickel et al. 2015). One possible way to obtain these new triplets is to extract facts from plain texts. We will seek methods to complete knowledge graphs with new triplets whose entities and relations come from plain texts.

Acknowledgments

This work was supported by the National Basic Research Program of China (No. 2014CB340503) and the National Natural Science Foundation of China (No. 61272332 and No. 61202329).

References

George A. Miller. 1995. WordNet: A lexical database for english. *Communications of the ACM,* 38(11):39-41.

Bollacker K., Evans C., Paritosh P., Sturge T., and 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.* pages 1247-1250.

Fabian M. Suchanek, Kasneci G., Weikum G. 2007. YAGO: A core of semantic Knowledge Unifying WordNet and Wikipedia. In *Proceedings of the 16th international conference on World Wide Web.*

Bordes A., Usunier N., Garcia-Durán A. 2013. Translating Embeddings for Modeling Multi-relational Data. In *Proceedings of NIPS.* pags:2787-2795.

Wang Z., Zhang J., Feng J. and Chen Z. 2014. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of AAAI.* pags:1112-1119.

Lin Y., Zhang J., Liu Z., Sun M., Liu Y., Zhu X. 2015. Learning Entity and Relation Embeddings for Knowledge Graph Completion. In *Proceedings of AAAI.*

Bordes A., Glorot X., Weston J., and Bengio Y. 2012. Joint learning of words and meaning representations for open-text semantic parsing. In *Proceedings of AISTATS.* pags:127-135.

Bordes A., Glorot X., Weston J., and Bengio Y. 2014. A semantic matching energy function for learing with multirelational data. *Machine Learning.* 94(2):pags:233-259.

Bordes A., Weston J., Collobert R., and Bengio Y. 2011. Learning structured embeddings of knowledge bases. In *Proceedings of AAAI.* pags:301-306.

Jenatton R., Nicolas L. Roux, Bordes A., and Obozinaki G. 2012. A latent factor model for highly multi-relational data. In *Proceedings of NIPS*. pags:3167-3175.

Sutskever I., Salakhutdinov R. and Joshua B. Tenenbaum. 2009. Modeling Relational Data using Bayesian Clustered Tensor Factorization. In *Proceedings of NIPS*. pags:1821-1828.

Socher R., Chen D., Christopher D. Manning and Andrew Y. Ng. 2013. Reasoning With Neural Tensor Networks for Knowledge Base Completion. In *Proceedings of NIPS*. pags:926-934.

Weston J., Bordes A., Yakhnenko O. Manning and Ununier N. 2013. Connecting language and knowledge bases with embedding models for relation extraction. In *Proceedings of EMNLP*. pags:1366-1371.

Matthew D. Zeiler. 2012. ADADELTA: AN ADAPTIVE LEARNING RATE METHOD. In *Proceedings of CVPR*.

Socher R., Huval B., Christopher D Manning. Manning and Andrew Y. Ng. 2012. Semantic Compositionality through Recursive Matrix-vector Spaces. In *Proceedings of EMNLP*.

Nickel M., Tresp V., Kriegel H-P. 2011. A three-way model for collective learning on multi-relational data. In *Proceedings of ICML*. pages:809-816.

Nickel M., Tresp V., Kriegel H-P. 2012. Factorizing YAGO: Scalable Machine Learning for Linked Data. In *Proceedings of WWW*.

Nickel M., Tresp V. 2013a. An Analysis of Tensor Models for Learning from Structured Data. *Machine Learning and Knowledge Discovery in Databases, Springer.*

Nickel M., Tresp V. 2013b. Tensor Factorization for Multi-Relational Learning. *Machine Learning and Knowledge Discovery in Databases, Springer.*

Nickel M., Murphy K., Tresp V., Gabrilovich E. 2015. A Review of Relational Machine Learning for Knowledge Graphs. In *Proceedings of IEEE.*