Contents lists available at ScienceDirect
Knowledge-Based Systems
journal homepage: www.elsevier.com/locate/knosys

Path-specific knowledge graph embedding

Yantao Jia*, Yuanzhuo Wang, Xiaolong Jin, Xueqi Cheng

CAS Key Lab of Network Data Science and Technology, Institute of Computing Technology, Chinese Academy of Sciences, China

A R T I C L E I N F O

Article history:
Received 18 June 2017
Revised 11 March 2018
Accepted 13 March 2018
Available online xxx

Keywords:
Path-specific
Knowledge graph embedding
Relation path

A B S T R A C T

Knowledge graph embedding aims to represent entities, relations and multi-step relation paths of a knowledge graph as vectors in low-dimensional vector spaces, and supports many applications, such as entity prediction, relation prediction, etc. Existing embedding methods learn the representations of entities, relations, and multi-step relation paths by minimizing a general margin-based loss function shared by all relation paths. This setting fails to consider the differences among different relation paths.

In this paper, we propose an embedding method by minimizing a path-specific margin-based loss function for knowledge graph embedding, called PaSKoGE. For each path, it adaptively determines its margin-based loss function by encoding the correlation between relations and multi-step relation paths for any given pair of entities. PaSKoGE outperforms the-state-of-the-art methods.

1. Introduction

Knowledge graph, such as Freebase [1] and WordNet [2], is a graph with entities of different types as vertices and various relations among entities as edges. Here entities are real objects in the world (e.g., Barack Obama), or abstract concepts in human's mind (e.g., the 44th President of United States). Relations describe the relationships between two entities (e.g., Barack Obama is the 44th President of United States, where the relation between Barack Obama and the 44th President of United States is "is-a"). Two entities and one relation between them form a triple in a knowledge graph (e.g., (Barack Obama, is-a, the 44th President of United States) is a triple). A knowledge graph usually contains billions of vertices, multi-typed edges and triples. Thus it is not easy to model such a large-scale graph in real applications, such as entity prediction [3].

Recently, some methods are proposed to embed the entities and relations of a knowledge graph into low-dimensional vector spaces, called knowledge graph embedding [4,5]. They represent entities and relations as vectors by optimizing a margin-based loss function, where the margin is a nonnegative number and is used to separate positive triples from negative ones. Typical methods include TransE [3], TransH [5], TransR [4], HOLE [6], TransA [7,8], etc. For example, TransE optimizes the margin-based loss function $\sum_{(h,r,t)}\sum_{(h',r,t')}(f_r(h,t)-f_r(h',t')+M)_+$, where $f_r(h, t)$ and $f_r(h', t')$ are score functions of the positive triple (h, r, t) and the negative triple (h', r, t'), respectively. The positive triple (h, r, t) is a triple in a knowledge graph, where h and t are two entities and r is the relation between them. The negative triple (h', r, t') is obtained from (h, r, t) by replacing h by h' and t by t'. $(x)_+ = \max(0, x)$ returns the maximum between 0 and x. M denotes the margin which is set to be a constant shared by all triples for different knowledge graphs and its optimal value is determined by experiments. For instance, the optimal margin on FB15K, a subset of Freebase [3], is equal to 1 by using TransE. In order to adaptively find the optimal margin-based loss function for different knowledge graphs, TransA introduces a margin-varying loss function such that each triple possesses its own optimal margin. TransA significantly improves the performance of knowledge graph embedding.

In addition to the representations of entities and relations, a type of path-based method has been studied. It represents multi-step relation paths between entities as vectors in low-dimensional vector spaces by minimizing a *general* margin-based loss function, which means that the margin-based loss function is shared by all multi-step relation paths. Here a multi-step relation path is a sequence of edges connecting a sequence of vertices. Typical path-based embedding methods include PTransE [9] and RTransE [10]. For instance, PTransE employs a margin-based loss function where the optimal margin M is set as a constant shared by all relation paths and its optimal value is also determined by experiments on different knowledge graphs. For example, the optimal margin on FB15K is equal to 1 by using PTransE. Although the path-based embedding methods greatly boost the performances of real applications such as entity prediction and relation prediction, ~~they fail to consider the differences among relation paths~~. For example, the relation path Barack Obama $\xrightarrow{\text{FatherOf}}$ Sasha Obama $\xrightarrow{\text{DaughterOf}}$ Michelle Obama connecting the two entities "Barack Obama"

* Corresponding author.
E-mail address: jiayantao@ict.ac.cn (Y. Jia).

https://doi.org/10.1016/j.knosys.2018.03.020
0950-7051/© 2018 Published by Elsevier B.V.

and "Michelle Obama" who have the spouse relationship, and the relation path Barack Obama $\xrightarrow{\text{ColleagueOf}}$ John Kerry $\xrightarrow{\text{ColleagueOf}}$ Hillary Clinton connecting the two entities "Barack Obama" and "Hillary Clinton" who are colleagues, are obviously different, but ~~they share the same margin-based loss function~~ where the margin is equal to 1. Moreover, it can be empirically demonstrated that the settings of margin in the margin-based loss functions for different knowledge graphs are different. For example, for two different subgraphs of Freebase, i.e., Subgraph1 and Subgraph2, stated in Section 3, the optimal performances of embeddding are obtained when the margins are set to be 6 and 2, respectively. Most importantly, when performing knowledge graph embedding on the whole knowledge graph, the specific setting of margins, namely, the relation paths in Subgraph1 and Subgraph2 with the optimal margin being 6 and 2 respectively, takes better results compared with those where the margins of all relation paths are set to be 1.

In this paper, we aim to propose a path-based knowledge graph embedding method which adaptively finds the optimal margin-based loss function. Specifically, we propose a path-specific knowledge graph embedding method, called PaSKoGE, which learns the representations of entities, relations and multi-step relation paths by minimizing a *path-specific* margin-based loss function for knowledge graph embedding. Namely, for each path, it adaptively determines its margin individually. It should be mentioned that this task is difficult for two reasons. Firstly, the relation paths in a knowledge graph have different lengths, and the determination of margins should be applied to all paths of different lengths. Secondly, the number of relation paths in a real knowledge graph is always enormous. For instance, there are about 10 million relation paths with length 2 in FB15K. And the determination of margins is supposed to be simple to handle such huge amount of relation paths. To this end, we define the margin for each path of arbitrary length by simply encoding the correlation between relations and multi-step relation paths for any given pair of entities. Moreover, for the complexity of PaSKoGE, it possesses the same number of parameters as other simple methods such as TransE, TransA, and PTransE. Experiments on standard benchmarks demonstrate the effectiveness of PaSKoGE.

The remainder of the paper are organized as follows. In Section 2, we provide some backgrounds on existing knowledge graph embedding methods. Section 3 studies the impact of path-specific setting of margins on the performance of knowledge graph embedding. Section 4 presents the path-specific method by minimizing a path-specific margin-based loss function. Section 5 presents the experimental results of the proposed method. Section 6 concludes the paper.

2. Related work

Classic knowledge graph embedding methods represent entities and relations of knowledge graphs as vectors in low-dimensional vector spaces by minimizing a margin-based loss function, which employs a nonnegative number, called the margin, as a hyperparameter to well separate positive triples from negative ones [5]. The introduction of margin is commonly used in many margin-based models such as support vector machines [11]. Typical classic embedding methods include TransE [3], TransH [5], TransR [4], Unstructured method [12,13], Structured Embedding (SE) [14], Semantic Matching Energy (SME) [12], Neural Tensor Network (NTN) [15], Latent Factor Model (LFM) [16,17], RESCAL [18,19], TransA [7,8], HOLE [6], TransG [20], etc. For example, for all triples (h, r, t) in a knowledge graph where r represents the relation between h and t, TransE uses the loss function

$$\sum_{(h,r,t)} \sum_{(h',r,t')} \left(\|\mathbf{h}+\mathbf{r}-\mathbf{t}\|_2^2 - \|\mathbf{h}'+\mathbf{r}-\mathbf{t}'\|_2^2 + M\right)_+,$$

where $\|\cdot\|_2$ is the L_2-norm and the boldface characters $\mathbf{h}$, $\mathbf{r}$ and $\mathbf{t}$ denote the embedding vectors of h, r and t, respectively. The triple (h', r, t') is obtained from (h, r, t) by replacing h by h' and t by t', and it is called a negative triple if it is not contained in the knowledge graph. $(x)_+ = \max(0, x)$ returns the maximum between 0 and x. M denotes the margin and is set to be a constant shared by all triples for different knowledge graphs. TransE applies well to 1-to-1 relations but has issues for N-to-1, 1-to-N and N-to-N relations. To address the issue of TransE when modeling N-to-1, 1-to-N and N-to-N relations, TransH is proposed to enable an entity to have distinct distributed representations when involved in different relations. For a relation r, TransH models the relation as a vector r on a hyperplane with w_r as the normal vector. Both TransE and TransH assume embeddings of entities and relations in the same space. However, an entity may have multiple aspects, and various relations focus on different aspects of entities. To model the entities and relations in separate spaces, TransR is proposed. Specifically, for each triple (h, r, t), entity embeddings are set as $h, t \in \mathbb{R}^k$ and relation embedding is set as $r \in \mathbb{R}^d$. The above three methods use the same setting of margin for different knowledge graphs, which ignores the locality of knowledge graphs. To adaptively determine the margin for different knowledge graphs, TransA introduces a promising loss function with flexible margins, denoted by $M_{opt}(h, r, t)$, such that each triple (h, r, t) possesses its own optimal margin. These classic methods employ the direct relations between entities, and are very simple to understand in practice. However, they fail to consider the multi-step relation paths between entities in a knowledge graph, which indicate more semantic information or structures between entities. Therefore, the performances of these methods are not satisfactory in some cases.

To incorporate more structural information into the knowledge graph embedding process, a type of path-based method for studying the representations of multi-step relation paths has been proposed (e.g., PTransE [9], RTransE [10], COMP [21]). These methods also minimize a margin-based loss functions, where the margin is shared by all relation paths. For instance, PTransE uses the loss function which contains the term $\sum_p \sum_{(h,r',t)} (\|\mathbf{p}-\mathbf{r}\| - \|\mathbf{p}-\mathbf{r}'\| + M)_+$, where $\|\mathbf{p}-\mathbf{r}\|$ is the L_1-norm or L_2-norm of $\mathbf{p}-\mathbf{r}$. And $\mathbf{p}$ and $\mathbf{r}$ are the embedding vectors of the multi-step relation path p and the relation r between h and t, respectively. r' is called the corrupted relation such that (h, r', t) is not a triple in the knowledge graph. The margin M is set as a constant shared by all relation paths. It obtains the embedding vector of a relation path from its component translation vectors by using three types of composition operations, i.e., addition, multiplication, and recurrent neural network. However, those path-based embedding methods finally learn the representations by minimizing a global margin-based loss function whose margin is selected from a fixed set of candidates shared by different knowledge graphs. This also ignores the difference between knowledge graphs. And it can be seen from Section 3 that determining the individual margin for each path tends to yield better performance. In this paper, we intend to propose a path-based embedding method to adaptively determine the margin of each path of arbitrary length in the loss function so as to further improve the performance of knowledge graph embedding.

3. The impact of path-specific setting of margins on the performance

In this section, we will investigate whether the path-specific setting of margin improves the performance of knowledge graph embedding by experiments. Experimental results demonstrate the

Table 1
Different choices of optimal margins and the predictive performances over different knowledge graphs. Partition0 means that the knowledge graph FB15K is not partitioned.

Knowledge graphs		Margin	Hits@10	
			Raw	Filter
Partition0	Whole FB15K	1	51.8	83.4
Partition1	Subgraph1	6	52.5	87.0
	Subgraph2	2		
Partition2	Subgraph A	4	53.1	87.1
	Subgraph B	3		
	Subgraph C	3		
	Subgraph D	4		
	Subgraph E	1		

sensitivity of the performance of knowledge graph embedding to the setting of margins, and show the necessity of choosing margins adaptively for each path.

Specifically, we conduct the experiments in four steps. Firstly, we partition the knowledge graph into several subgraphs in a uniform manner. Secondly, the representations of entities, relations as well as multi-step relation paths on each subgraph are learnt by using the well-known path-based embedding method PTransE. On each subgraph, it follows from PTransE that the optimal settings of general margin-based loss functions on each subgraph can be found, where the margin is shared by all relation paths on each individual subgraph. Thirdly, we learn the embedding on the whole knowledge graph by using PTransE. Finally, we learn the embedding on the whole knowledge graph by using PTransE such that if the relation path appears in one of the subgraphs, then its margin is set to be the optimal one discovered in the second step. Otherwise, its margin is set to be the one discovered in the third step. We would like to see whether the performances of the two settings in the last two steps are different.

To this end, we adopt the widely used knowledge graph FB15K[1], which contains 1345 different types of relations and 14,951 entities. And we partition FB15K into two subgraphs with almost equal size of relations, and denote them by Subgraph1 and Subgraph2, respectively, and denote this partition by Partition1. More precisely, we randomly select 672 relations as the relations of Subgraph1. Then the entities related to these 672 relations are regarded as the entities of Subgraph1. In this way, we find 14,585 entities in Subgraph1. The rest 673 relations are the relations of Subgraph2, and the number of related entities is equal to 14,639. It can be seen that the set of relations of the two subgraphs are disjoint with each other. This leads to the fact that the relation paths of the two subgraphs are also disjoint with each other. We conduct entity prediction task over these two subgraphs by using PTransE and use Hits@10 (i.e., the proportion of correct entities in top-10 ranked entities) to evaluate the results shown in Table 1.

It follows from Table 1 that the settings of optimal margins are different over the two subgraphs, Subgraph1 and Subgraph2. More precisely, in order to obtain the best performance, the two subgraphs take different values of margin, i.e., 6 and 2, respectively. Meanwhile, for the whole knowledge graph FB15K, it has been shown in [9] that the best performance is achieved when the optimal margin is equal to 1. Most importantly, when performing knowledge graph embedding on the whole knowledge graph FB15K, the specific setting of margins, namely, the relation paths in Subgraph1 and Subgraph2 with the optimal margin being 6 and 2 respectively, takes higher Hits@10 (e.g., 87.0) compared with those (i.e., 83.4) obtained by directly using PTransE on FB15K where the margins of all relation paths are set to be 1. This suggests that knowledge graph embedding with different settings of margins of relation paths, i.e., path-specific setting of margins, can obtain better performance, as compared with the general setting of margins such that all relation paths share a common margin.

To further validate this finding, we partition the knowledge graph into more subgraphs. For instance, we partition FB15K into five subgraphs with equal size of relations, and denote them by Subgraph A, Subgraph B, Subgraph C, Subgraph D, Subgraph E, respectively, and denote this partition by Partition2. Each subgraph has 169 relations and the set of relations of all subgraphs are disjoint with each other. Specifically, similar to the process mentioned before, we randomly select 169 relations as the relations of Subgraph A, and find the entities related to these relations as the entities of Subgraph A. Then we randomly select 169 relations among the rest 1176 relations as the relations of Subgraph B, and find the entities related to those relations as the entities of Subgraph B. Repeat this process, and finally we obtain all the subgraphs. Table 1 illustrates the Hits@10 over these subgraphs on the task of entity prediction. It can be seen that when conducting knowledge graph embedding on the whole graph FB15K, the specific settings of margins that the relation paths in each subgraph take their own margins listed in Table 1 take higher Hits@10 compared with the direct employment of PTransE on FB15K (i.e., Partition0) where the optimal margin of all relation paths is set to be 1. Moreover, we find that Partition2 with 5 subgraphs takes higher Hits@10 than Partition1 with 2 subgraphs.

It seems that partitioning the whole knowledge graph into more subgraphs (e.g., 10 subgraphs) such that the relation paths in each subgraph have their own optimal settings of margins leads to better performance for knowledge graph embedding. In particular, if each relation path itself becomes a subgraph individually, and has its specific setting of margin, it is possible that the performance of embedding becomes the best. However, since the number of relation paths in FB15K is enormous (e.g., 14,230,850 paths with length 2) and the lengths of relation paths are different, it needs a simple way to conduct this partition and determine the margin for each path. In the following section, we regard each relation path between two entities as a subgraph, and simply determine its optimal margin to yield better performance of knowledge graph embedding.

4. The path-specific method

In previous section, we have examined by experiments that finding a specific setting of margin for each relation path over different knowledge graphs helps to promote better performance. In this section, we propose a path-specific method, called PaSKoGE, to adaptively choose the optimal margin for each path. More specifically, given a relation path of arbitrary length between a head entity and a tail entity, we define its optimal margin simply by modeling the correlation between the relations and multi-step relation paths connecting them. We then present PaSKoGE by taking the optimal margin into consideration.

4.1. Path-specific margin

Given a pair of entities (h, t) in the knowledge graph, let $P_{h,t}$ denote the set of multi-step relation paths from h to t of arbitrary lengths. Let $R_{h,t}$ be the set of direct relations r such that (h, r, t) is a correct triple in the knowledge graph. Meanwhile, let $N_{h,t}$ be the set of corrupted relations r', which is the relation between h and r such that (h, r', t) is obtained from (h, r, t) by substituting r' for r, and (h, r', t) is not a golden triple in the knowledge graph. It has been mentioned in [22] that for the given pair (h, t), the paths in $P_{h,t}$ highly correlate with the direct relations in $R_{h,t}$, since the relationships between h and t can be interpreted by both of them.

[1] https://everest.hds.utc.fr/lib/exe/fetch.php?media=en:fb15k.tgz.

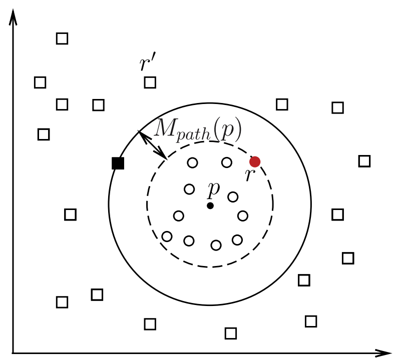

Fig. 1. The illustration of $M_{path}(p)$ of a path p. The circles and rectangles are correct and corrupted relations, respectively.

In other words, if we represent a path $p \in P_{h,t}$ and a direct relation $r \in R_{h,t}$ as vectors in the vector space $\mathbb{R}^d$, it is natural that the corresponding vectors, denoted as $\mathbf{p}$ and $\mathbf{r}$, are supposed to be close to each other in their locations. On the contrary, the embedding vector $\mathbf{r'}$ of the corrupted relation $r' \in N_{h,t}$ should keep far away from $\mathbf{p}$ than $\mathbf{r}$ because it is corrupted.

To characterize this obvious intuition, the optimal margin for the path p, denoted by $M_{path}(p)$, is set to co-locate the similar pair $(\mathbf{p}, \mathbf{r})$, and move the corrupted relation vector $\mathbf{r'}$ with a certain distance away from $\mathbf{p}$. Geometrically, for a given pair of entities (h, t) and a path $p \in P_{h,t}$, the path-specific margin $M_{path}(p)$ is equal to the distance between two concentric spheres with $\mathbf{p}$ as the same center, where the internal sphere contains the correct relations, whilst the corrupted relations lie outside the external sphere (see Fig. 1). More formally, we define $M_{path}(p)$ as follows.

Definition 1 (Path-specific margin). For a given pair of entities (h, t) and a path $p \in P_{h,t}$, for all $r \in R_{h,t}$ and $r' \in N_{h,t}$,

$$M_{path}(p) = \min_{r \in R_{h,t}, r' \in N_{h,t}} \left(\|\mathbf{p} - \mathbf{r'}\| - \|\mathbf{p} - \mathbf{r}\| \right). \tag{1}$$

It can be seen from Definition 1 that for each relation path p, the value $\min_{r,r'} \left(\|\mathbf{p} - \mathbf{r'}\| - \|\mathbf{p} - \mathbf{r}\| \right)$ takes the minimum, when it takes the nearest corrupted relation and the farthest correct relation with respect to $\mathbf{p}$. This definition is similar to the margin defined in support vector machine [11,23], where the margin of two classes is equal to the minimum absolute difference of the distances of any two different-class instances projected to the norm vector of the hyperplane.

To find the corrupted relation $r' \in N_{h,t}$ for a large knowledge graph in a more efficient way, for a given pair of entities (h, t), we adopt the active set strategy proposed in [7,24]. Specifically, we check a very small fraction of corrupted relations typically lying nearby the farthest correct relation for several rounds. Averaging the path-specific margins obtained in all rounds leads to the final optimal path-specific margin.

4.2. The path-specific method paskoge

Let Δ be the set of triples in a knowledge graph. The margin-varying objective function of PaSKoGE is

$$\sum_{(h,r,t) \in \Delta} \left[E_{h,r,t} + \frac{1}{Z} \sum_{p \in P_{h,t}} R(p|h,t) H_{p,r} \right], \tag{2}$$

where

$$E_{h,r,t} = \sum_{(h',r',t')\Delta'} \left(\|\mathbf{h} + \mathbf{r} - \mathbf{t}\| + \gamma_{opt} - \|\mathbf{h'} + \mathbf{r'} - \mathbf{t'}\| \right)_+,$$

the set Δ' contains the triples obtained by replacing one of the three components h, t and r of $(h, r, t) \in \Delta$ by h', t' and r'; $\mathbf{h}, \mathbf{r}, \mathbf{t}, \mathbf{h'}, \mathbf{r'}, \mathbf{t'} \in \mathbb{R}^d$, d is the dimension of the embedding vector space; $(x)_+$ returns the maximum between x and 0; $\|\cdot\|$ represents L_1-norm or L_2-norm; γ_{opt} is the optimal margin to separate positive triples from negative ones, and is defined in the same way as TransA [7].

$$Z = \sum_{p \in P_{h,t}} R(p|h,t)$$

is a normalization factor, where $R(p|h, t)$ denotes a score to measure the reliability of the relation path p given the pair of entities (h, t). We follow the path-constraint resource allocation algorithm presented in [9] to compute its value such that it is equal to the resource amount allocated from the head entity h to the tail entity t along the path p. Another application of $R(p|h, t)$ is to rank and select reliable paths from h to t, since the number of candidate paths between two given entities may be enormous for a large knowledge graph. For any given pairs of entities (h, t), we select the path p whose score $R(p|h, t)$ is larger than 0.01 as a reliable path in the same way as [9]. Similar path selection techniques can be found in [10,22,25,26]. In Eq. (2),

$$H_{p,r} = \alpha \|\mathbf{p} - \mathbf{r}\| + \sum_{(h,r',t) \in \Delta''} L_{p,r},$$

where α is a nonnegative smoothing factor, the term $\|\mathbf{p} - \mathbf{r}\|$ guarantees that the embedding vectors of relation path p and the direct relation r are close to each other as expected.

$$L_{p,r,r'} = \left(\|\mathbf{p} - \mathbf{r}\| + M_{path}(p) - \|\mathbf{p} - \mathbf{r'}\| \right)_+,$$

where $M_{path}(p)$ is the optimal path-specific margin defined by Eq. (1), and Δ'' is the set of corrupted triples, i.e., $\Delta'' = \{(h, r', t) | (h, r, t) \in \Delta\}$. The relation path $p = (r_1, r_2, \ldots, r_l)$ is the composition of the relations $r_1, r_2, \ldots, r_l$, and its embedding vector $\mathbf{p} \in \mathbb{R}^d$ is simply equal to the addition of the embedding vectors of its components, namely,

$$\mathbf{p} = \mathbf{r}_1 + \mathbf{r}_2 + \cdots + \mathbf{r}_l.$$

The learning process employs the Stochastic Gradient Descent (SGD) method. The set of positive triples (the triples from the knowledge graph) are randomly traversed multiple times. When a positive triple (h, r, t) is visited, a negative triple (h', r', t') is randomly constructed by replacing one of the three components of (h, r, t) with the other entities or relations in the knowledge graph. The above generation of negative triples is called as "unif" and is widely used in the open literature, e.g., see [5,7]. After a mini-batch, the gradient is computed and the model parameters are updated.

Next, we consider the complexity of PaSKoGE from the aspect of model parameters, which has been widely used to evaluate the complexity of knowledge graph embedding methods in the open literature, e.g., see [3,5,6]. Since PaSKoGE learns one low-dimensional vector for each entity and each relation, its number of parameters is the same as other simple embedding methods, e.g., TransE, TransA, HOLE, PTransE, see Table 2 for detailed numbers of parameters of those methods. Let n_e and n_r denote the number of entities and relations in a knowledge graph, respectively. d is the embedding dimension. Then we have.

Proposition 2. *The number of model parameters of PaSKoGE is equal to $\mathcal{O}(dn_e + dn_r)$.*

5. Experiments

To validate the proposed method PaSKoGE, we conduct experiments on two representative tasks, i.e., entity prediction [3] and

Table 2
Number of parameters and their values on FB15K(in millions) with $d = 100$.

Methods	Model Parameters	FB15K
TransE	$\mathcal{O}(dn_e + dn_r)$	1.6M
TransH	$\mathcal{O}(dn_e + 2dn_r)$	1.8M
RESCAL	$\mathcal{O}(dn_e + n_r d^2)$	14.9M
TransR	$\mathcal{O}(dn_e + dn_r + n_r d^2)$	15.1M
HOLE	$\mathcal{O}(dn_e + dn_r)$	1.6M
TransA	$\mathcal{O}(dn_e + dn_r)$	1.6M
PTransE	$\mathcal{O}(dn_e + dn_r)$	1.6M
PaSKoGE	$\mathcal{O}(dn_e + dn_r)$	1.6M

Table 3
The statistics of the data sets.

Data	#Rel	#Ent	#Train	#Valid	#Test
WN18	18	40,943	141,442	5000	5000
FB15K	1345	14,951	483,142	50,000	59,071

Table 4
Experimental results about entity prediction.

Data sets	WN18		FB15K	
Metric	Hits@10		Hits@10	
	Raw	Filter	Raw	Filter
Unstructured	35.3	38.2	4.5	6.3
RESCAL	37.2	52.8	28.4	44.1
SE	68.5	80.5	28.8	39.8
SME(linear)	65.1	74.1	30.7	40.8
SME(bilinear)	54.7	61.3	31.3	41.3
LFM	71.4	81.6	26.0	33.1
TransE	75.4	89.2	34.9	47.1
TransH	75.4	86.7	42.5	58.5
TransR	78.3	91.7	43.8	65.5
PTransE(2-step)	19.4	35.4	51.8	83.4
PTransE(3-step)	19.4	34.6	51.4	84.6
RTransE	39.8	79.5	13.4	48.2
TransG	81.4	93.3	53.1	79.8
PaSKoGE	81.3	95.0	53.1	88.0

Table 5
Evaluation results on FB15K by mapping properties of relations.(%).

Tasks	Predicting Heads (Hits@10)				Predicting Tails (Hits@10)			
Relation	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	34.5	2.5	6.1	6.6	34.3	4.2	1.9	6.6
SE	35.6	62.6	17.2	37.5	34.9	14.6	68.3	41.3
SME (linear)	35.1	53.7	19.0	40.3	32.7	14.9	61.6	43.3
SME (bilinear)	30.9	69.6	19.9	38.6	28.2	13.1	76.0	41.8
TransE	43.7	65.7	18.2	47.2	43.7	19.7	66.7	50.0
TransH	66.7	81.7	30.2	57.4	63.7	30.1	83.2	60.8
TransR	76.9	77.9	38.1	66.9	76.2	38.4	76.2	69.1
TransG	85.4	95.7	44.7	80.8	84.0	56.5	95.0	83.6
PaSKoGE	89.7	94.8	62.3	86.7	89.3	72.9	93.4	88.9

relation prediction [9], which have been widely utilized to evaluate knowledge graph embedding methods. The data sets used in the experiment are publicly available from two widely used knowledge graphs, WordNet [2] and Freebase [1]. For the data sets from WordNet and Freebase, we employ WN18 and FB15K, respectively, as used in [3–5]. The statistics of the data sets are listed in Table 3.

5.1. *Entity prediction*

Entity prediction aims to predict the missing entities h or t for a triple (h, r, t). Specifically, it predicts t given (h, r) or h given (r, t). Similar to the settings in [3,14,27] , the task returns a list of candidate entities from the knowledge graph instead of one best answer. Following the procedure used in [9,10], we also adopt the proportion of correct entities in top-10 ranked entities (denoted by Hits@10) instead of mean rank because Hits@10 is less sensitive to outliers. It is clear that a good predictor has high Hits@10. The baseline methods for comparison include path-based embedding methods PTransE [9], RTransE [10], and classical embedding methods, i.e., TransE [3], TransH [5], TransR [4], and others shown in Table 4. Since the data sets we adopt are the same as those of the baselines, we compare our experimental results with those reported in [3–5,9]. Note that for the path-based embedding methods PTransE, and RTransE, the experimental results on WN18 are not reported. We employ their publicly available codes to obtain the results on WN18. For the parameter tuning process, we determine their values in the same way as the existing methods. Specifically, the learning rate λ during the SGD process is selected among {0.1, 0.01, 0.001}, the embedding dimension d in {20, 50, 100}, the batch size B among {20, 120, 480, 1440, 4800}, and the smoothing factor α is selected among {1, 0.1, 0.01, 0.001}. Note that the path-specific margin for PaSKoGE is not predefined but calculated by Eq. (1). All parameters are determined on the validation set. The optimal parameter setting for WN18 is $\lambda = 0.001$, $d = 50$, $B = 1440$, and $\alpha = 0.01$, while that for FB15K is $\lambda = 0.001$, $d = 100$, $B = 4800$, and $\alpha = 0.01$. For both WN18 and FB15K, L_1 is taken as dissimilarity.

Experiment results are presented in Table 4, where “Filter” denotes that the correct triples among the corrupted triples are filtered out before the ranking of candidate entities, while “Raw” indicates that correct triples among the corrupted triples are not filtered. The length of path is equal to 2 in the following results. And for paths whose lengths are less than or equal to 3, the results are briefly stated in Section 5.4. On WN18, PaSKoGE obtains the highest filter Hits@10. In particular, compared with the best baseline method TransG, PaSKoGE obtains almost the same raw Hits@10, and increases the filter Hits@10 by about 2%, where the relative increase is equal to 25% (i.e., 1.7/(100-93.3)). And it can be seen that PaSKoGE outperforms the two path-based method PTransE and RTransE. On FB15K, PaSKoGE obtains the highest filter Hits@10. Again compared with the best predictor TransG, PaSKoGE obtains the same raw Hits@10, and increases the filter Hits@10 by about 8.2%, where the relative increase is equal to 40% (i.e., 8.2/(100-79.8)).

We also present the evaluation results by mapping properties of relations in terms of the proportion of correct entities in top-10 ranked entities (denoted by Hits@10) in Table 5, where the mapping properties of relations are defined in the same way as TransR. It follows from Table 5 that PaSKoGE performs the best among the baselines. In particular, PaSKoGE outperforms TransG for six cases, while for the rest two cases, PaSKoGE also has competitive performance with TransG.

5.2. *Relation prediction*

Relation prediction is to predict the relations for a pair of entities. The baseline methods for comparison include path-based embedding method PTransE [9], RTransE [10], as well as the classic embedding methods, TransE [3], TransR [4], as shown in Table 6. Following the procedure used in [9], we also adopt the proportion of correct entities in top-1 ranked entities (denoted by Hits@1). Note that we report Hits@1 instead of Hits@10 for comparison, because Hits@10 for all methods exceeds 95%.

As for the parameters of PaSKoGE, the learning rate λ during the SGD process is selected among {0.1, 0.01, 0.001}, the embedding dimension d in {20, 50, 100}, the parameter γ being larger than 0, batch size B among {20, 120, 480, 1440, 4800}, and the smoothing factor α is selected among {1, 0.1, 0.01, 0.001}. The optimal margin for PaSKoGE is again calculated according to Eq. (1). All parameters are determined on the validation set. The optimal

Table 6
Results about relation prediction.

Data sets	WN18		FB15K	
Metric	Hits@1(%)		Hits@1(%)	
	Raw	Filter	Raw	Filter
TransE	66.1	66.1	65.1	84.3
TransR	–	–	67.6	87.6
RTransE	28.8	28.9	33.3	33.3
PTransE(2-step)	67.1	67.3	69.5	93.6
PTransE(3-step)	64.4	64.6	68.5	94.0
PaSKoGE	97.2	97.6	69.7	94.2

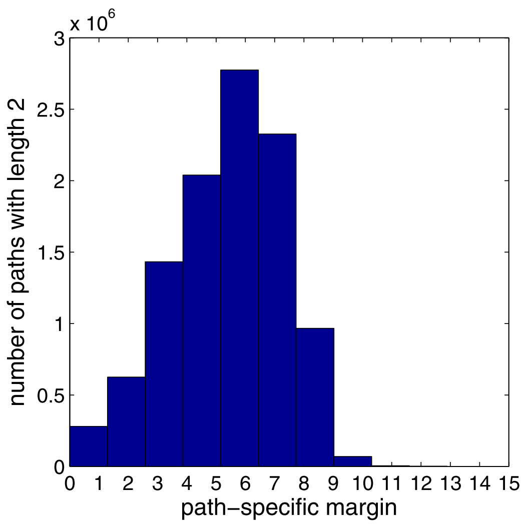

Fig. 2. The relation between the path-specific margins and the number of corresponding paths.

parameter setting for WN18 is $\lambda = 0.001$, $d = 50$, $B = 1440$, and $\alpha = 0.01$, whilst for FB15K it is $\lambda = 0.001$, $d = 50$, $B = 4800$, and $\alpha = 0.01$. Again, for both of them, L_1 is taken as dissimilarity. Experiment results are presented in Table 6, where "Filter" and "Raw" denote the same meanings as in Table 4. We can see that on both data sets, PaSKoGE outperforms the baseline methods. In particular, on WN18, compared with other baseline methods, PaSKoGE significantly increases Hits@1 by 30%, i.e., from about 65% to 97%.

5.3. Discussions about the path-specific margins

In what follows, we compute the path-specific margins for the 14,230,850 paths of length 2 in FB15K by virtue of Eq. (1) and show their path-specific margins in a histogram (see Fig. 2). Note that the margins of paths of length 3 can be also computed, and the histogram can be depicted similarly.

From Fig. 2, it follows that most relation paths take the margins between 5 and 6 (i.e., 19.34% of all relation paths). And it can be computed that the minimum path-specific margin is equal to 4.14×10^{-6}, while the maximum path-specific margin is equal to 12.8909. Moreover, we find that about 39% paths have margins less than 5, and about 42% paths have margins greater than 6. Fig. 2 actually demonstrates that the proposed method can find vastly different margins for different paths.

5.4. Efficiency and scalability

We further discuss the efficiency of PaSKoGE. To this end, we present the comparisons of runtime, i.e., elapsed time of embedding measured by seconds, between different methods on the two data sets WN18 and FB15K. We select five baseline methods, i.e.,

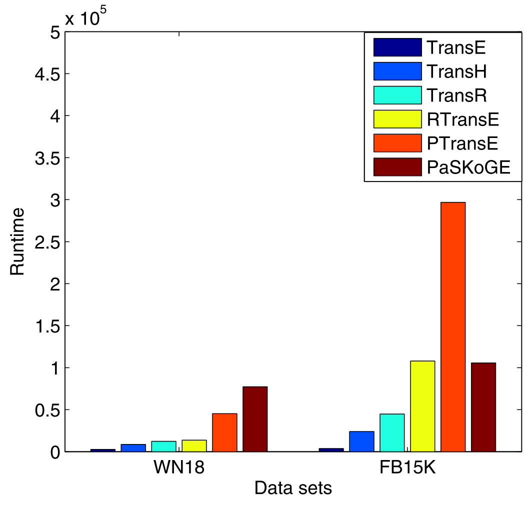

Fig. 3. Comparison of runtime between different methods.

Table 7
Comparison of runtime between different methods.

Data sets	TransE	TransH	TransR	RTransE	PTransE	PaSKoGE
WN18	2734	8594	12,399	13,639	45,321	77,302
FB15K	3692	23,946	44,733	107,850	296,786	105,661

TransE, TransH and TransR as classic methods as well as RTransE and PTransE as path-based methods. The embedding are learned with the same setting of embedding dimension, i.e., d=50. Experimental results are shown in Fig. 3 and Table 7. From Fig. 3 and Table 7, we have the following interesting observations: (1) On both data sets, TransE takes the least time than any of the other methods. (2) On WN18, compared with PTransE, PaSKoGE takes more time to additionally compute the path-specific margin, e.g., 9 additional hours, but its performance (i.e., filtered Hits@10) increases relatively by about 25% shown in Table 4. (3) On the large data set FB15K, it can be seen that PaSKoGE takes less time than PTransE. This is also due to its effective setting of margins. Based on the large amount of paths, the embedding method can take relatively few time to achieve its optimum solution, and is shown to be more powerful over the paths of length 2.

Moreover, to show the scalability of PaSKoGE over large graphs with many long paths, we present the runtime of PaSKoGE on the large data set FB15K by considering the 2-paths, 3-paths (i.e., paths of length less than or equal to 3), 4-paths (i.e., paths of length less than or equal to 4). For instance, FB15K has a huge number of 3-paths up to 17,857,839. More precisely, for $l = 2, 3, 4$, we record the runtime of embedding in each mini-batch during the optimization process of PaSKoGE by fixing the number of iterations as 1000. In each mini-batch, we randomly select a collection of triples. And for each pair of entities of the triples, we use the paths whose lengths are less than or equal to l between them for embedding. Adding the runtime for each pair of entities sampled in a mini-batch leads to the total runtime of embedding for each mini-batch. The relation between the total runtime of embedding for each mini-batch and the number of iterations of the optimization for $l = 2, 3, 4$ are shown in Fig. 4, Fig. 5, Fig. 6, respectively. It can be seen that when the number of iterations increases, the total runtime of embedding for each mini-batch ranges in a small interval, and finally it converges. For example, for the case of 3-paths in Fig. 5, the interval has the minimum 561, and the maximum 581. Moreover, after the 800th iteration, the total runtime of embedding is almost unchanged whose values are among the six numbers, i.e., 561, 562, 563, 564, 565, 566. For the case of 2-paths and 4-paths, the obser-

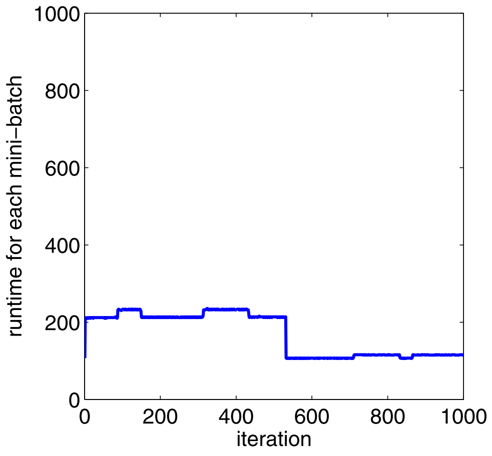

Fig. 4. The relation between the total runtime for each mini-batch and the number of iterations on FB15K with many 2-paths.

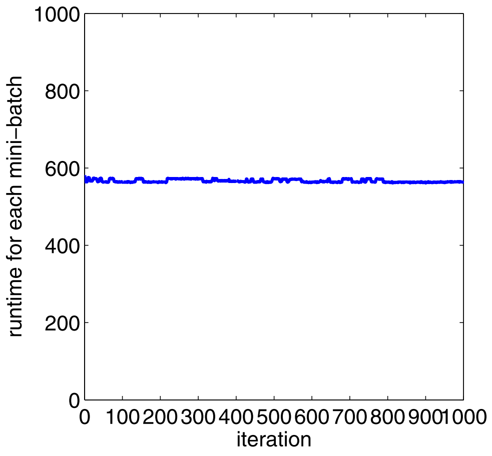

Fig. 5. The relation between the total runtime for each mini-batch and the number of iterations on FB15K with many 3-paths.

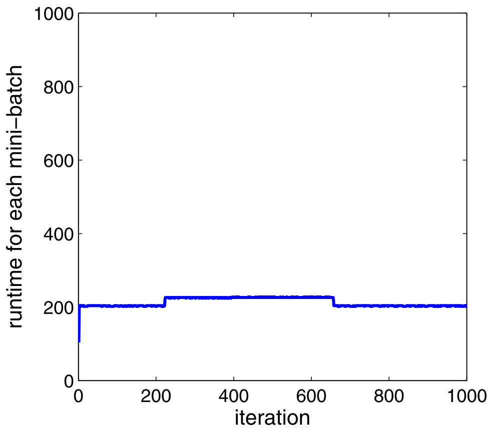

Fig. 6. The relation between the total runtime for each mini-batch and the number of iterations on FB15K with many 4-paths.

vations are similar. This suggests the scalability of PaSKoGE over the large data set with many paths.

6. Conclusions

In this paper, we proposed a path-specific embedding method, called PaSKoGE, to adaptively learn the representations of entities, relations and relation paths in a knowledge graph. To this end, we first presented the necessity of choosing an appropriate margin in the margin-based loss function. We then defined the path-specific margin for each relation path. Experiments on two representative tasks and benchmark data sets validated the effectiveness of the proposed method.

Acknowledgments

This work is supported by National Key Research and Development Program of China under grant 2016YFB1000902, National Grand Fundamental Research 973 Program of China (Nos. 2012CB316303, 2014CB340401), National Natural Science Foundation of China (Nos. 61402442, 61572469, 91646120, 61572473, 61303244, 61402124).

References

[1] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, J. Taylor, Freebase: a collaboratively created graph database for structuring human knowledge, in: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, ACM, 2008, pp. 1247–1250.
[2] G.A. Miller, Wordnet: a lexical database for english, Commun. ACM 38 (11) (1995) 39–41.
[3] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, O. Yakhnenko, Translating embeddings for modeling multi-relational data, in: Advances in Neural Information Processing Systems, 2013, pp. 2787–2795.
[4] Y. Lin, Z. Liu, M. Sun, Y. Liu, X. Zhu, Learning entity and relation embeddings for knowledge graph completion, in: Proceedings of the 29th AAAI Conference on Artificial Intelligence, 2015, pp. 2181–2187.
[5] Z. Wang, J. Zhang, J. Feng, Z. Chen, Knowledge graph embedding by translating on hyperplanes, in: Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, Citeseer, 2014, pp. 1112–1119.
[6] M. Nickel, L. Rosasco, T. Poggio, Holographic embeddings of knowledge graphs, in: Thirtieth AAAI Conference on Artificial Intelligence, 2016.
[7] Y. Jia, Y. Wang, H. Lin, X. Jin, X. Cheng, Locally adaptive translation for knowledge graph embedding, in: Thirtieth AAAI Conference on Artificial Intelligence, 2016.
[8] Y. Jia, Y. Wang, X. Jin, H. Lin, X. Cheng, Knowledge graph embedding: a locally and temporally adaptive translation-based approach, ACM Trans. Web (TWEB) 12 (2) (2018).
[9] Y. Lin, Z. Liu, M. Sun, Modeling relation paths for representation learning of knowledge bases, in: Conference on Empirical Methods in Natural Language Processing, 2015, pp. 705–714.
[10] A. García-Durán, A. Bordes, N. Usunier, Composing relationships with translations, in: Conference on Empirical Methods in Natural Language Processing, 2015, pp. 286–290.
[11] V. Vapnik, The nature of statistical learning theory, Springer Science & Business Media, 2013.
[12] A. Bordes, X. Glorot, J. Weston, Y. Bengio, Joint learning of words and meaning representations for open-text semantic parsing, in: International Conference on Artificial Intelligence and Statistics, 2012, pp. 127–135.
[13] A. Bordes, X. Glorot, J. Weston, Y. Bengio, A semantic matching energy function for learning with multi-relational data, Mach. Learn. 94 (2) (2014) 233–259.
[14] A. Bordes, J. Weston, R. Collobert, Y. Bengio, Learning structured embeddings of knowledge bases, in: Conference on Artificial Intelligence, EPFL-CONF-192344, 2011.
[15] R. Socher, D. Chen, C.D. Manning, A. Ng, Reasoning with neural tensor networks for knowledge base completion, in: Advances in Neural Information Processing Systems, 2013, pp. 926–934.
[16] R. Jenatton, N.L. Roux, A. Bordes, G.R. Obozinski, A latent factor model for highly multi-relational data, in: Advances in Neural Information Processing Systems, 2012, pp. 3167–3175.
[17] I. Sutskever, J.B. Tenenbaum, R.R. Salakhutdinov, Modelling relational data using bayesian clustered tensor factorization, in: Advances in neural information processing systems, 2009, pp. 1821–1828.
[18] M. Nickel, V. Tresp, H.-P. Kriegel, A three-way model for collective learning on multi-relational data, in: Proceedings of the 28th International Conference on Machine Learning (ICML-11), 2011, pp. 809–816.
[19] M. Nickel, V. Tresp, H.-P. Kriegel, Factorizing yago: scalable machine learning for linked data, in: Proceedings of the 21st international conference on World Wide Web, ACM, 2012, pp. 271–280.
[20] H. Xiao, M. Huang, X. Zhu, Transg: a generative model for knowledge graph embedding, in: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 1, 2016, pp. 2316–2325.
[21] K. Guu, J. Miller, P. Liang, Traversing knowledge graphs in vector space, in: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, 2015, pp. 318–327.
[22] F. Wu, J. Song, Y. Yang, X. Li, Z. Zhang, Y. Zhuang, Structured embedding via pairwise relations and long-range interactions in knowledge base, in: Twenty-Ninth AAAI Conference on Artificial Intelligence, 2015.

[23] B.E. Boser, I.M. Guyon, V.N. Vapnik, A training algorithm for optimal margin classifiers, in: Proceedings of the fifth annual workshop on Computational learning theory, ACM, 1992, pp. 144–152.

[24] K.Q. Weinberger, L.K. Saul, Fast solvers and efficient implementations for distance metric learning, in: Proceedings of the 25th international conference on Machine learning, ACM, 2008, pp. 1160–1167.

[25] K. Toutanova, X.V. Lin, W.-t. Yih, Compositional learning of embeddings for relation paths in knowledge bases and text, in: Proceedings of the Annual Meeting of the Association for Computational Linguistics, 2016.

[26] N. Lao, T. Mitchell, W.W. Cohen, Random walk inference and learning in a large scale knowledge base, in: Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, 2011, pp. 529–539.

[27] Z. Zhao, Y. Jia, Y. Wang, Content-structural relation inference in knowledge base, in: Twenty-Eighth AAAI Conference on Artificial Intelligence, 2014.