Link Prediction in Knowledge Graphs: A Hierarchy-Constrained Approach

Manling Li, Yuanzhuo Wang, *Member, IEEE*, Denghui Zhang, Yantao Jia, *Member, IEEE*, and Xueqi Cheng, *Member, IEEE*

Abstract—Link prediction over a knowledge graph aims to predict the missing head entities h or tail entities t and missing relations r for a triple (h, r, t). Recent years have witnessed great advance of knowledge graph embedding based link prediction methods, which represent entities and relations as elements of a continuous vector space. Most methods learn the embedding vectors by optimizing a margin-based loss function, where the margin is used to separate negative and positive triples in the loss function. The loss function utilizes the general structures of knowledge graphs, e.g., the vector of r is the translation of the vector of h and t, and the vector of t should be the nearest neighbor of the vector of $h + r$. However, there are many particular structures, and can be employed to promote the performance of link prediction. One typical structure in knowledge graphs is hierarchical structure, which existing methods have much unexplored. We argue that the hierarchical structures also contain rich inference patterns, and can further enhance the link prediction performance. In this paper, we propose a hierarchy-constrained link prediction method, called hTransM, on the basis of the translation-based knowledge graph embedding methods. It can adaptively determine the optimal margin by detecting the single-step and multi-step hierarchical structures. Moreover, we prove the effectiveness of hTransM theoretically, and experiments over three benchmark datasets and two sub-tasks of link prediction demonstrate the superiority of hTransM.

Index Terms—Link prediction, knowledge graph embedding, hierarchy

1 Introduction

A knowledge graph is a graph whose nodes represent entities and edges correspond to relations. As an effective organization of structural and non-structural information, large-scale knowledge graphs are constantly emerging, including Freebase [1], WordNet [2], OpenKN [3], Probase [4], etc. In the past decades, knowledge graphs have played a pivotal role in many areas such as semantic search, question answering systems and so on. However, all of them have a need to improve their coverage of facts, for which link prediction is an effective strategy and has been paid much attention [5], [6], [7], [8]. Link prediction is a task to predict missing edges under the supervision of given part of knowledge graphs. Since knowledge graphs are often represented as triples $\{(h, r, t)\}$, where h and t denote the head entities and the tail entities respectively, and r denotes the relation between them, link prediction over a knowledge graph aims to predict the missing triples i.e., to predict t (or h) given (h, r)(or (r, t)), or predict r given (h, t).

There are a quantity of techniques to tackle this problem, which mainly fall into two categories. The first category contains rule-based and path-based methods, namely, the relations are predicted by explicitly learning rules and relation paths. Compared to the rule-based predictive methods, the path-based methods have greatly improved predictive performance, which indicates that relation paths are of great use in predicting relations. Nevertheless, path-based methods fail to predict links well when faced with entities having no relations with others before, and their performance plunges definitely for sparsely connected graph.

The second category is knowledge graph embedding based predictive methods, where the relations of entities are learned implicitly in the embedding vectors. Most methods embed a knowledge graph into a low-dimensional vector space according to a margin-based loss function, and then a score for each triple (h, r, t) is calculated based on score function $f_r(h, t)$. Finally, a list of candidate triples is returned in the decreasing order of their scores. The typical methods are translation-based methods, for example, TransE [9]. Moreover, TransA [10] employs more structural information of knowledge graphs, i.e., the local distance of entities and the proximity of relations. The success of TransA indicates that the special structures in knowledge graphs can further enhance the link prediction performance. Furthermore, there are other methods that employ the particular structures to achieve better performance. For instance, inspired by PRA, PTransE [11] integrates the relation paths into the learning process, which improves the link prediction performance significantly.

However, there are many particular structures of knowledge graphs that are not taken full advantage of. One typical structure is the hierarchical structure, which is a structure where entities are organized in a tree, and their relations are hierarchical relations [12]. We argue that hierarchical

• *The authors are with the CAS Key Laboratory of Network Data Science and Technology, Institute of Computing Technology, Chinese Academy of Science, Beijing 100864, China. E-mail: limanlingcs@gmail.com, {wangyuanzhuo, zhangdenghui, jiayantao, cxq}@ict.ac.cn.*

Manuscript received 30 Apr. 2017; revised 17 Jan. 2018; accepted 20 May 2018. Date of publication 0 . 0000; date of current version 0 . 0000.
(Corresponding authors: Manling Li and Yuanzhuo Wang.)
Recommended for acceptance by P. Luo, Y. Song, and H. Wang.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TBDATA.2018.2867583

structures can further improve the link prediction performance, since they also contain rich patterns, similar to relation paths. In addition, hierarchical structures are extremely common in knowledge graphs due to the ubiquitousness of hierarchical relations. For instance, WN18, a subset of the knowledge graph WordNet, has about 50 percent hierarchical relations. Furthermore, hierarchical structures will lead to a special distribution of entities in the embedding space, which provides more constraints compared to non-hierarchical structures, so that the performance of link prediction will be promoted.

Motivated by this intuition, we propose a hierarchy-constrained link prediction method based on knowledge graph embedding, called hTransM. Specifically, we seek out a method to detect hierarchical structures and model them from two aspects, i.e., the single-step hierarchical structures and the multi-step hierarchical structures, and the optimal hierarchy-constrained margin is learned with respect to different hierarchical structures. As a result, the embedding vectors of the entities and relations of the knowledge graph can be trained under the supervision of hierarchical structures. Experiments are conducted over three benchmark datasets for two subtasks of link prediction, and the results demonstrate that the proposed method outperforms the state-of-the-art method. Specifically, the contributions of the paper are four-fold.

- We divide the hierarchical structures into two categories, i.e., single-step hierarchical structures and multi-step hierarchical structures. Besides, we provide one way to detect the hierarchical structures in knowledge graphs by employing the properties of hierarchical relations. Moreover, the influence of hierarchical structures on the performance of link prediction is analyzed in an intuitional way.
- We propose a hierarchy-constrained link prediction method based on knowledge graph embedding, called hTransM. According to whether the relations and relation paths that connect the entities are hierarchical or not, it finds the optimal loss function by adaptively determining the margins.
- We further prove the convergence of hTransM by demonstrating its uniform stability and provide the upper bound of the error of the proposed model. What's more, hTransM possesses the same model complexity (i.e., the number of parameters) as other simple methods such as TransE.
- Experiments over three benchmark datasets of two sub-tasks, i.e., entity prediction task and relation prediction task, suggest that the proposed method can achieve better prediction performance. Furthermore, the superiority of hierarchy-constrained margin is validated by experiments through studying the variation of the optimal margin value along with the optimization process, and compare with the methods which do not considering hierarchical information.

2 Related Work

Link prediction in knowledge graphs, i.e., predicting links between entities, has received much attention in recent years. Since knowledge graphs are often represented as triples (h, r, t), the link prediction task can be formulated as predicting t given (h, r), or predicting h given (r, t), or predicting r given (h, t).

Existing link prediction methods mainly fall into two categories according to the ways in modeling the existing relation between entities. The first category is to explicitly model the existing relations, including rule-based and path-based methods. For example, FOIL [13], [14], [15] is a rule-based method by employing the first order inductive logic, and is followed by the models reducing manual work in defining rules, e.g., NELL [15], Sherlock-Holmes [16]. In the meanwhile, PRA [17], [18] is a typical path-based method, which models the existing relations by relation paths. The relation path is made up of the relations from head entity to tail entity, such as $p = \left(\cdot \xrightarrow{r_1} \cdot \xrightarrow{r_2^{-1}} \cdot \xrightarrow{r_3} \cdots \xrightarrow{r_l} \cdot \right)$, where l is the length of the relation path p. For a given relation $\mathbf{r}$, it first find the triples formed by this relation (h, r, t), and then the paths between all h and t. It regards the relation paths as features, and predicts the relations by ranking candidates in terms of the weighted score of these relation paths. PRA has achieved a great success in link prediction, which demonstrates the effectiveness of relation path information in link prediction task. Then, a bunch of methods [19], [20],[21] emerge to enrich the relation paths for better predictive performance. Besides, the relation paths already have been widely used in a quantity of applications, such as expert path finding [22], relation extraction based on KB structure [17] [23], etc. Although path-based methods greatly improve the link prediction performance, the performance of them is hampered for sparsely connected graph and they are not scalable quite well for large scale knowledge graphs.

The second category is to implicitly model the relations of entities, including knowledge graph embedding based predictive methods. According to the structures of the graph, these methods embed a knowledge graph into a low-dimensional latent space, and then a score function $f_r(h, t)$ for each triple (h, r, t) is learned. Finally, a list of candidate entities is returned in the decreasing order in term of their scores. In recent years, knowledge graph embedding based methods have received much attention, which mainly fall into two key branches, i.e., the translation-based methods and others. Translation-based models regard the relation of a triple as the translation from the head entity to the tail entity, including TransE [9], TransH [24], TransR [25], PTransE [11], TransA [10], etc. They usually define a margin-based loss function to separate the negative triples from positive triples in the embedding space.

TransE [9] is a pioneering work of Translation-based knowledge graph embedding methods. It models the embedding vector of relation $\mathbf{r}$ as translation from the head entity embedding vector $\mathbf{h}$ to the tail entity embedding vector $\mathbf{t}$, i.e., $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$, where $\mathbf{h}, \mathbf{r}, \mathbf{t} \in \mathbb{R}^d$ are the embedding vectors of h, r and t, and d is the dimension of embedding space. As a result, the score function for each triple (h, r, t) is $f_r(h, t) = ||\mathbf{h} + \mathbf{r} - \mathbf{t}||$, and the candidate triples are ranked in the decreasing order of their scores. TransE works well for 1-to-1 relations but has issues for N-to-1, 1-to-N and N-to-N relations. For instance, it can be derived that $h_0 = \cdots = h_i = \cdots = h_m$, for all (h_i, r, t) in the knowledge graph when r is a 1-to-N relation, where m is the number of these triples. This is obviously in conflict with the fact.

To address this issue, TransH [24] considers that the distance between h and t is distinct from different relations, and each relation represents a different hyperplane. Consequently, TransH formulates the relation as a projection transformation, namely, $f_r(h,t) = ||\mathbf{h}_\perp + \mathbf{r} - \mathbf{t}_\perp||$, where $\mathbf{h}_\perp = \mathbf{h} - \mathbf{w}_\mathbf{r}^\top \mathbf{h} \mathbf{w}_\mathbf{r}$ and $\mathbf{t}_\perp = \mathbf{t} - \mathbf{w}_\mathbf{r}^\top \mathbf{t} \mathbf{w}_\mathbf{r}$, with $\mathbf{w}_\mathbf{r}$ as the normal vector of the hyperplane related to r. Furthermore, TransR [25] models the relation as a rotation transformation, namely, $f_r(h,t) = ||\mathbf{h}_\mathbf{r} + \mathbf{r} - \mathbf{t}_\mathbf{r}||$, where $\mathbf{h}_\mathbf{r} = \mathbf{M}_\mathbf{r}\mathbf{h}$ and $\mathbf{t}_\mathbf{r} = \mathbf{M}_\mathbf{r}\mathbf{t}$. In other words, it utilizes a projection matrix $M_r \in \mathbb{R}^{k\times d}$, where k is the dimension of the entity embedding vector space, and d is the dimension of the relation embedding vector space. Similar work also contains TransD [26] and TransM [27].

Besides translation-based embedding methods, there are also other models to learn the embedding vectors of the knowledge graphs. For example, energy-based methods aim to assign low energies to the triples and are optimized by neural networks. Unstructured model [28] is one of the typical energy-based models. It ignores the relation information and the score function is simplified to $f_r(h,t) = ||\mathbf{h} - \mathbf{t}||$. The Structured Embedding (SE) model [29] defines two matrix corresponding to the relations to transform the entities, and the score function is formulated as $f_r(h,t) = ||\mathbf{M}_{\mathbf{h},\mathbf{r}}\mathbf{h} - \mathbf{M}_{\mathbf{t},\mathbf{r}}\mathbf{t}||$. The Semantic Matching Energy (SME) model [30] formulates the score function by employing the correlations between entities and relations with two matrix operators, i.e., $f_r(h,t) = (\mathbf{M_1 h} + \mathbf{M_2 h} + \mathbf{b_1})^\top (\mathbf{M_1 h} + \mathbf{M_2 h} + \mathbf{b_2})$ using add operator, and $f_r(h,t) = (\mathbf{M_1 h} \otimes \mathbf{M_2 h} + \mathbf{b_1})^\top (\mathbf{M_1 h} \otimes \mathbf{M_2 h} + \mathbf{b_2})$ using Hadamard operator. The LFM model [31], [32] considers a quadratic form to model the second-order correlations between entity embedding vectors, and the score function is $f_r(h,t) = \mathbf{h}^\top \mathbf{M}_\mathbf{r} \mathbf{t}$.

Both TransE, TransH, TransR and other translation-based methods employ a fixed margin based loss function, and the value of the fixed margin is chosen during experiments. However, knowledge graphs exhibit different locality with regard to different types of entities and relations. Namely, when the entities and relations change, the optimal margin between the negative and positive triple scores should vary accordingly. Hence, TransA [10] determines the optimal margin adaptively. It employs the entity-specific margin, which is the local distance between negative and positive entities, and relation-specific margin, which is the proximity of relations. Since the predictive performance of TransA is fairly good, the effectiveness of structural information to promote the predictive performance has been verified.

There are other methods improving the knowledge embedding performance by making use of the rich information of typical structures, such as PTransE [11]. It combines the original triple loss with the loss generated by path $\sum_p \sum_{(h,r',t)} (||\mathbf{p} - \mathbf{r}|| - ||\mathbf{p} - \mathbf{r}'|| + M)$, where $\mathbf{p}$ is the embedding vector of the multi-step relation path p. PTransE significantly outperform the methods that do not utilize the special structure of knowledge graphs, which demonstrates the superiority of employing special structures in knowledge graphs. As a result, we consider that one of the typical structures of knowledge graphs, i.e., hierarchical structure, contains rich inference information, and can be employed to promote the performance of link prediction. However, the existing knowledge graph embedding methods fail to utilize the rich information in hierarchical structures.

The hierarchical structure is a structure where entities are organized in a tree, and their relations are hierarchical relations [9]. Owing to its universality, the hierarchical structure has been explored a lot recently. Existing works most focus on type hierarchy of knowledge graph or class hierarchy of classification task. For example, Taxonomy Embedding represents the class and entities into a latent semantic space that underlies the class hierarchy, and then the classification is done with simple nearest neighbor rule [33]. Label Embedding Trees approach aims to learn a tree-structure by optimizing the overall tree loss for multi-class classification [34]. Besides, TKRL [35] leverages the type hierarchy and the entity type constraints for each relation during knowledge graph embedding. However, they did not use the hierarchical structures formed by entities. Wang et al. [36] propose to employ the hierarchical information by defining the entity similarity as the distance along the tree. They explore the hierarchical structures from the global aspect, which ignores the hierarchical information of a single-step, i.e., from local aspect. Actually, the hierarchical structures can be analyzed from two aspects, i.e., the single-step aspect and the multi-step aspect, both of which can provide rich information to promote the performance of link prediction. Consequently, we shall propose a hierarchy-constrained link prediction method based on knowledge graph embedding, to integrate both the single-step and multi-step hierarchical information into the predictive method.

3 Hierarchical Structure

3.1 Hierarchical Structure Formulation

3.1.1 Hierarchical Structure

A hierarchical structure is a structure where entities are organized into layers by one relation [37]. Different layers imply different vertical orders, and for each entity, the other entities are above, below, or in the same layer as it [38]. The meaning varies from relations, e.g., in the hierarchical structure organized by relation *child*, different layers indicate different generations. More formally, hierarchical structures are defined following [39].

Definition 1. *A* hierarchical structure *generated by relation* r^* *is* $H(r^*) = (\{(h,r,t)\}, l)$, *where the subgraph connected by* r^* *is a directed acyclic graph, and* l *is a mapping from nodes to layer indexes* $l(h), l(t) \in \{1, 2, \ldots, k\}, k \geq 1$, *with properties that* $l(t) > l(h)$ *for* (h,r,t), *and* $l(t) = l(h) + 1$ *for* (h, r^*, t).

Taking the knowledge graph Fig. 1a as an example, *Barack Obama Sr.*, his child *Barack Obama* and his two grandchildren *Malia* and *Sasha* compose a three-layer hierarchical structure by relation *child*, illustrated in Fig. 1b, and the subgraph connected by relation *child* is a directed acyclic graph, shown in Fig. 1c. $l(Barack\,Obama\,Sr.) = 1$, $l(Barack\,Obama) = 2$, $l(Sasha) = l(Malia) = 3$. Given $(Barack\,Obama\,Sr.,\ grandchild,\ Malia)$, $l(Malia) > l(Barack\,Obama\,Sr.)$. Given $(Barack\,Obama,\ child,\ Malia)$, $l(Malia) = l(Barack\,Obama) + 1$.

3.1.2 Hierarchical Relation and Relation Path

Hierarchical relations are relations generating hierarchical structures and distributing entities into different layers, e.g., the relation *child* in Fig. 1.

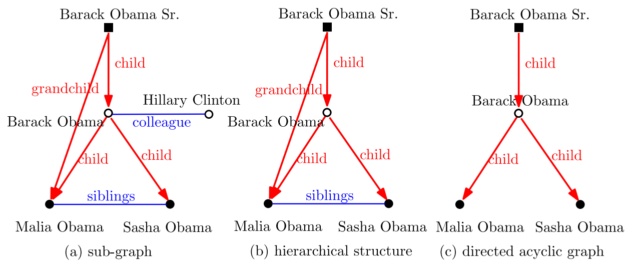


Fig. 1. (a) represents a sub-graph of knowledge graph, and (b) is the three-layered hierarchical structure extracted from it, which is composed by relation *child*, and (c) is the directed acyclic graph connected by relation *child*. Red arrows represent hierarchical relations, and the blue lines represent non-hierarchical relations.

Definition 2. *A* hierarchical relation *is the relation r that enables entities to be organized into hierarchical structures, i.e., $l(h) \neq l(t)$ given (h, r, t).*

A *relation path* is made up of the relations from the head entity to the tail entity [17], e.g., the solid line in Fig. 2 illustrates the relation paths extracted from Fig. 1. Besides, it has been proved in [11] that, for a given entity pair (h, t), the relation paths p connecting (h, t) are consistent with the direct relation r between (h, t), since the relationships between (h, t) can be interpreted by both of them. In this case, we call the relation as *consistent relation* of the path.

Definition 3. *A* relation path *is a path consisted of relations $p = \left(\xrightarrow{r_1} \cdot \xrightarrow{r_2} \dots \xrightarrow{r_l}\right)$. If $\exists(h, p, t) \wedge (h, r, t)$, relation r is the* consistent relation *of path p.*

For instance, the paths (solid line in Fig. 2) are highly correlated to the direct single-step relation (dotted line in Fig. 2). Taking Fig. 2a as an example, the consistent relation of the relation path $\left(\xrightarrow{child} \cdot \xrightarrow{child}\right)$ is the relation *grandchild*.

Note that the relation paths in knowledge graphs consist of the inverse relations [17]. For example, the triple (*Barack Obama*, *child*, *Malia*) can be inversed to (*Malia*, *child*$^{-1}$, *Barack Obama*), and the relation path $p = \left(\xrightarrow{child^{-1}} \cdot \xrightarrow{child}\right)$ contains inverse relation, which can be interpreted by relation *siblings*, so its consistent relation is *siblings*. Besides, the non-hierarchical relations are considered as existing in both directions, that is to say, (*Sasha*, *siblings*, *Malia*) and (*Malia*, *siblings*, *Sasha*) both hold, so the inverse relation of a non-hierarchical relation is itself.

Moreover, the relation path could be a hierarchical path or not. A relation path is a hierarchical relation path, if there is at least one hierarchical relation among the related relations. In this case, the head entities and tail entities are enabled to be on different layers. More formally,

Definition 4. *A* hierarchical relation path *is a multi-step relation path $p = \left(\xrightarrow{r_1} \cdot \xrightarrow{r_2} \dots \xrightarrow{r_l}\right)$, which satisfies*

$$\exists i \in \{1, 2, \dots, l\}, \quad r_i \in H_r,$$

where l is the length of the relation path, and H_r is the set of hierarchical relations in the knowledge graph.

Otherwise, the path is *non-hierarchical relation path*.

For instance, the relation path $\left(\xrightarrow{child} \cdot \xrightarrow{child}\right)$ in Fig. 2a is a hierarchical relation path, since it contains two hierarchical relations. Similarly, the other ten relation paths in Fig. 2b, 2c, 2d, 2e, 2f, 2g, 2h, 2i, 2j are all hierarchical paths, as they all possess at least one hierarchical relation.

3.1.3 Single-Step and Multi-Step Hierarchical Structure

Hierarchical structures imply two kinds of inference information that can be used to link prediction, i.e., single-step and multi-step hierarchical structures. Hierarchical single-step structures are structures where entities distribute on two different layers and are connected by single-step relation, with tail entities sharing the same parent.

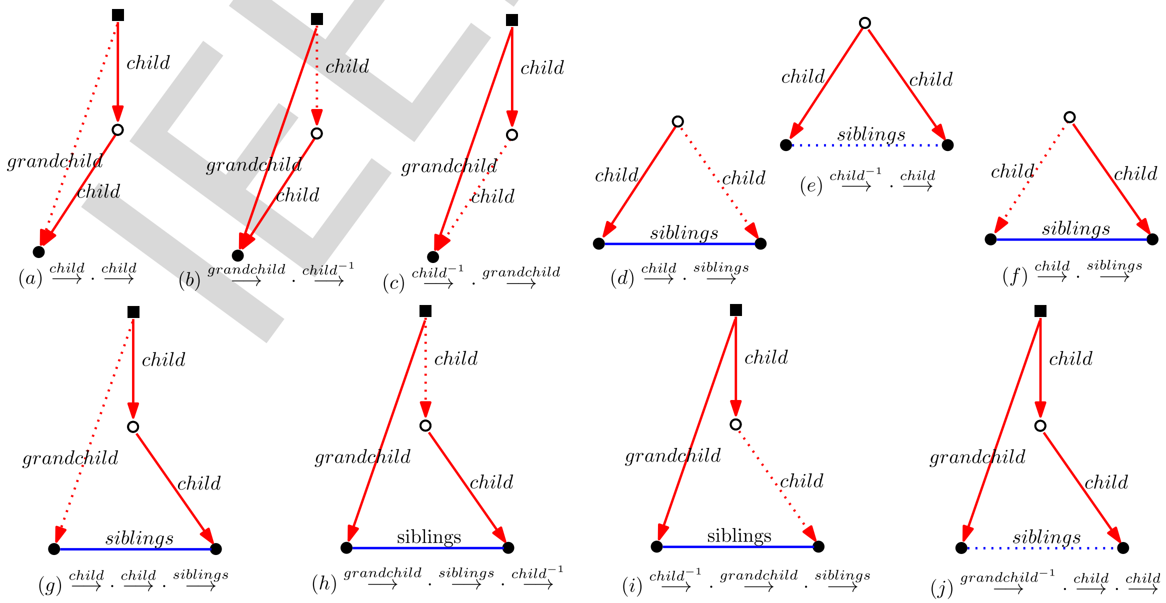
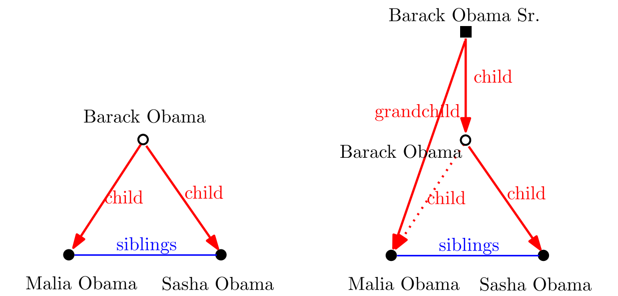


Fig. 2. Relation paths extracted from the knowledge graph in Fig. 1a, where the dotted edges represent the relation consistent with the path. For instance, the relation path (*child* · *child*) is highly correlated with the relation *grandchild*.



(a) single-step hierarchical structure (b) multi-step hierarchical structure

Fig. 3. An example of single-step and multi-step hierarchical structures extracted from Fig. 1b.

Definition 5. *A* hierarchical single-step structure *is a subgraph* $H(r^*, (h^*, r^*, t)) = (\{(h, r, t)\}, l)$ *of a hierarchical structure by extracting triples* (h^*, r^*, t) *and related relations connecting* $h^* \cup \{t\}$*, with the property that* $l(h^*) = 1$*,* $l(t) = 2$ *for all* $\{(h^*, r^*, t)\}$*. Here* $\{t\}$ *have the same parent* h^**, and* r^* *is hierarchical relation.*

For example, Fig. 3a is a hierarchical single-step structure $H(child, (Barack\ Obama, child, t))$ by extracting triples $(Barack\ Obama, child, Malia)$ and $(Barack\ Obama, child, Sasha)$, and the related relation $(Malia, siblings, Sasha)$. The three entities are on two layers, and $\{t\} = \{Malia, Sasha\}$ have the same parent *Barack Obama*, and *child* is a hierarchical relation. On the contrary, a *general single-step structure* is composed by non-hierarchical relations, e.g., a single triple with non-hierarchical relation. For instance, the relation *siblings* and the connected entities, i.e., *Sasha* and *Malia*, form a single-step structure, but it is a general single-step structure, since *siblings* is a non-hierarchical relation.

Instead, hierarchical multi-step structures are structures where entities distribute on multiple different layers and are connected by relation paths, with paths sharing same head and tail entities, as well as same consistent relation.

Definition 6. *A* hierarchical multi-step structure *is a subgraph* $H(r^*, (h^*, r^*, t^*)) = (\{(h, r, t)\}, l)$ *of a hierarchical structure by extracting a triple* (h^*, r^*, t^*) *and paths consistent with* r^* *and connecting* h^* *and* t^**, with the property that* $l_{\max}(t) - l_{\min}(h) > 1$ *for all* $\{(h, r, t)\}$*. Here* r^* *is hierarchical relation.*

For instance, Fig. 3b is a hierarchical multi-step structure $H(child, (Barack\ Obama, child, Malia))$ by extracting a triple $(Barack\ Obama, child, Malia)$, and its consistent paths $p_1 = \left(\xrightarrow{child^{-1}} \cdot \xrightarrow{grandchild}\right)$ and $p_2 = \left(\xrightarrow{child} \cdot \xrightarrow{siblings}\right)$. The three entities are on three layers, and $l_{\max}(t) - l_{\min}(h) = 2$, and *child* is a hierarchical relation. Otherwise, the multi-step structure is *general multi-step structure*. For example, the multi-step structure in Fig. 2e is a general one. Although the relation path $p = \left(\xrightarrow{child^{-1}} \cdot \xrightarrow{child}\right)$ is a hierarchical path, the consistent relation *siblings* is non-hierarchical, and the head entity *Malia* and tail entity *Sasha* are on the same layer.

Compared to hierarchical single-step structure, the entities in hierarchical multis-step structure can be distributed on more than two layers. Actually, the hierarchical single-step structure aims to capture the *local* hierarchical structure of knowledge graphs, where entities are on neighbour layers. Namely, it focus on the inter-layer information. On the contrary, the hierarchical multi-step structure focuses on the *global* hierarchical structure, which looks upon the whole hierarchical structure, and considering the information across layers.

3.2 Hierarchical Structure Effectiveness

Compared with the general relations, the entities connected by hierarchical relations will be distributed distinguishingly in the embedding space, since the hierarchical relations enforce the connected entities to be on different layers. Hence, it is intuitive that these constraints will contribute to the link prediction process. First, in the single-step hierarchical structures, the siblings are close to each other since they have the same parent and are semantic similar [9]. However, the traditional learning method only constrains the head entities and tail entities should be close, and fail to consider the semantic similarity among tail siblings. We argue that this similarity between siblings contains rich inference information for link prediction. which is useful in predicting links. For example, given that *Sasha* is a child of *Barack Obama*, and *Sasha* is very close to *Malia*, the inference of siblings between *Sasha* and *Malia* will assist the prediction of the child relation between *Barack Obama* and *Malia*. Second, in the multi-step hierarchical structures, the hierarchical paths are more relevant to hierarchical relations, since they both enables entities to distribute in different layers. For example, the hierarchical path $\left(\xrightarrow{child} \cdot \xrightarrow{child}\right)$ will better help to predict the hierarchical relation *grandchild* than the non-hierarchical path $\left(\xrightarrow{alumni} \cdot \xrightarrow{alumni}\right)$. Moreover, the more hierarchical relations the path contains, the more inference information the path will contribute. Taking Fig. 3b as example, to predict the triple $(Barack\ Obama, child, Malia)$, the inference confidence of its hierarchical consistent path $p_1 = \left(\xrightarrow{child^{-1}} \cdot \xrightarrow{grandchild}\right)$ is higher than path $p_2 = \left(\xrightarrow{child} \cdot \xrightarrow{siblings}\right)$.

Furthermore, hierarchical structures are ubiquitous in knowledge graphs, since hierarchical relations widely exist. For example, FB15K and WN18, the subset from widely used knowledge graph WordNet and Freebase respectively, both have about 50 percent hierarchical relations at least. The knowledge graphs of different domain may have different hierarchical relation proportions, but hierarchical structure is a natural and typical local structure of knowledge graphs. First, the "superior/subordinate" relation is common in real life. Apart from the example in genealogy graph aforementioned, the "advisor/advisee" widely exist in the academic knowledge graph, as well as "employer/employee" in the business knowledge graph. Second, a large proportion of relations have different entity types for head entity and tail entity, such as *directed_by* with type MOVIE for head entity and type PERSON for tail entity. These relations can naturally form singe-step hierarchical structures with a PERSON entity as parent, MOVIE entities as children, and $directed\_by^{-1}$ as hierarchical relation. The semantic similarity brought by the single-step hierarchical structure still works in this case, since the movies of one director is likely to be similar in style.

However, there are some domain knowledge graphs that fundamentally non-hierarchical, such as the social network knowledge graph, which is mainly formed by "follower/ followee" relation. This relation forms circles and thus the hierarchy-constrained knowledge graph embedding will suffer in such knowledge graphs.

3.3 Hierarchical Structure Discovery

It has been proved in [40], a DAG has a unique hierarchical structure, and each node can be labeled with a unique layer index, which can be calculated using a bunch of algorithms [39], [40], [41]. However, one may find the results disappointing when applied to a digraph that is fundamentally non-hierarchical. As a result, the difficulty to detect local hierarchical structures is to find subgraphs that are more fundamentally hierarchical, and the key is the discovery of hierarchical relations. One way to distinguish hierarchical relations in a knowledge graph is to study their properties, and there are many properties to rely on.

First, hierarchical relations can not form circles since hierarchical structures are DAGs. In this paper, we detect circles through topological sorting algorithm, which generates a linear ordering of its vertices such that for every triple, the head entity comes before the tail entity in the ordering. As shown in Algorithm 1, for a relation r^*, we extract the subgraph containing only r^*, and conduct topological sorting by deleting the vertices without incoming edges. If there lefts vertices that can not make topological sorting, then the circles exist, and the number of left vertices indicates the number of vertices participating in forming circles. Considering the fault tolerance in knowledge graphs, a hierarchical relation must have a low proportion of left vertices.

Algorithm 1. Detecting Circles Through Topological Sorting

Require:
Training set $S = \{(h, r, t)\}$, entities and relations set E, R;
Ensure:
The set of relations that can form circles R_c;

```
for all r* ∈ R do
  Extract a subgraph S_{r*} = {(h, r*, t)}
  N_{r*} ← |S_{r*}|
  Q_root ← nodes with no incoming edge
  while Q_root ≠ ∅ do
    Select h* ∈ Q_root
    for all t* has incoming edge from h* do
      S_{r*} ← S_{r*} − {(h*, r*, t*)}
      if t* has no other incoming edges then
        Q_root ← {t*} ∪ Q_root
      end if
    end for
  end while
  N'_{r*} ← |S_{r*}|
  Score(r*) ← N'_{r*} / N_{r*}
  if Score(r*) ≤ 0.2 then
    R_c ← r* ∪ R_c
  end if
end for
```

Second, hierarchical relations are irreflexive, since they are relations between layers, such as *child* between *Barack Obama* and *Sasha*. That is to say, head entity and tail entity are on different layers, so hierarchical relations have directions from the head entity to the tail entity, which means they are irreflexive. In this paper, we detect the irreflexive relations by the proportion of irreflexive triples, where a triple is called irreflexive triple if it does not hold when interchanging its head and tail. Provided that the proportion is bigger than 50 percent, the relation is considered as irreflexive relation. For instance, in WN18 [9], a subset of WordNet, the proportion of irreflexive triples related to relation *_similar_to* is 7.5 percent, so that *_similar_to* is non-hierarchical relation.

Third, hierarchical relations are closely in accordance with unbalanced mapping properties. Mapping properties can be defined following [9]: A given relation is *1-to-1* if a head can appear with at most one tail, *1-to-N* if a head can appear with many tails, and *1-to-1* and *N-to-N* analogously. Specially, *1-to-N* (e.g., WordNet's *_hyponym*) or *N-to-1* (e.g., WordNet's *_hypernym*) relations are always hierarchical ones. On the contrary, *1-to-1* (e.g., *passport id*, *successor*) relations do not have sibling information to help link prediction, so they are not considered for hierarchy-constraind link prediction in this paper. As for *N-to-N* relations, when the cardinalities of head and tail entities are almost the same (e.g., *siblings*, *colleagues*), the relations are usually reflexive or can form circles. However, when there is a gap between the cardinalities (e.g., *parent_to_child*, *advisor_to_advisee*), i. e, the larger one is more than 1.5 times than the smaller one in this paper, the relations are always can be used to form hierarchical structures if the relations are not reflexive without circles. For example, in WN18, relation *_also_see* has 1.65 heads per tail and 1.84 tail per head in average, so that it is labeled as *N-to-N*. With similar cardinalities of head and tail entities, *_also_see* is not a hierarchical relation. On the contrary, *_hypernym* has an average of 3.67 heads per tail and of 1.02 tail per head, so it is labeled as *N-to-1*. Plus the proportion of irreflexive related triples is 100 percent, and it can not form circles, the relation *_hypernym* is detected as hierarchical relation consequently.

Therefore, hierarchical relations are detected by irreflexive, non-circle, and unbalanced mapping properties. Notice that, utilizing these properties of relations is just one way to discover hierarchical relations in the statistical sense, and may not hold for every single hierarchical relation.

4 The Link Prediction Method

4.1 The Predictive Method hTransM

Since it has been verified in [10] that the appropriate value of M_{opt} will significantly improve the performance of link prediction, the idea of hTransM is to define a hierarchy-constrained margin M_{opt} by detecting the hierarchical and general structures, where M_{opt} is used to separate positive triples from negative triples. Specifically, the positive triples are the golden triples in the knowledge graph, denoted by $(h, r, t) \in \Delta$, and the negative triples are the corrupted triples constructed from (h, r, t), denoted by $(h', r', t') \in \Delta'$. They do not exist in the knowledge graph, and are constructed by substituting one of the entities or the relations following [9].

Then the embedding of entities and relations to a vector space $\mathbb{R}^d$ for each triple (h, r, t) is learned by minimizing a loss function concerning M_{opt},

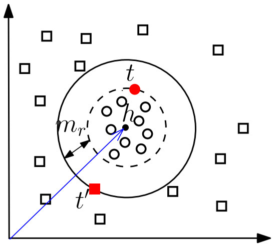


Fig. 4. The illustration of the general single-step specific margin, where circles stand for positive entities and rectangles represent negative ones in the embedding space $\mathbb{R}^d$.

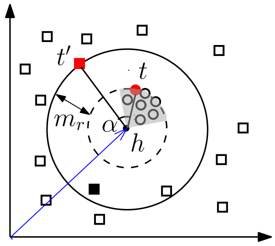


Fig. 5. The illustration of the hierarchical single-step specific margin, where circles stand for positive entities and rectangles represent negative ones in the embedding space $\mathbb{R}^d$.

$$L = \sum_{(h,r,t)\in\Delta} \sum_{(h',r',t')\in\Delta'} \max(0, f_r(h,t) + M_{opt} - f_r(h',t')), \quad (1)$$

where $f_r(h,t)$ is the score function for the triple (h,r,t). In this paper, the score function adopts the form defined in [9] without loss of generality,

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

where the boldface characters denote the embedding vectors of entities and relations in $\mathbb{R}^d$, and d is the dimension of the embedding space. For instance, $\mathbf{h}$ is the embedding vector of the entity h. And $||\cdot||$ is the L_1-norm or L_2-norm of the vector. In order to predict t given (h,r) or predict h given (r,t), candidate entities are ranked in terms of $f_r(h,t)$ and a list is returned in the decreasing order of $f_r(h,t)$.

Since hierarchical structures are fully composed of single-step hierarchical structures and multi-step hierarchical structures, the optimal hierarchy-constrained margin M_{opt} is modelled from two aspects, i.e., single-step aspect and multi-step aspect. Moreover, it is natural to linearly combine the two specific margins via parameters α, β to control the trade-off between them. Therefore, the optimal hierarchy-constrained margin satisfies

$$M_{opt} = \alpha M_{single} + \beta M_{multi}, \quad (3)$$

where $0 \leq \alpha, \beta \leq 1$, M_{single} denotes the single-step specific margin and M_{multi} denotes the multi-step specific margin. For the sake of obtaining optimal hierarchy-constrained margin M_{opt}, it is sufficient to find the optimal single-step specific margin M_{single} and the optimal multi-step specific margin M_{multi}, so we will elaborate the setting of two margins respectively in the following.

4.2 Single-Step Specific Margin

For a given triple (h,r,t), the single-step specific margin is generated according to the corresponding single-step specific structure. First, let us denote some notations. For a given entity h and its related relation r, the set of positive entities, denoted by P_r, contains entities that have relation with h of type r. And the set of negative entities, denoted by N_r, contains entities that have relations with h of other relation r'.

Provided that the single-step structure is a general one, it has been verified in [42] that, for a given head entity h and relation r, the optimal performance is achieved when the embedding vectors bring positive tail entities P_r close to each other, and move negative tail entities N_r away with a margin. Hence, the positive entities should be closer to h than the negative ones. Namely, given relation r, the distance from positive entities $t \in P_r$ to h, i.e., $||\mathbf{h} - \mathbf{t}||$, should be shorter than the distance from negative entities $t' \in N_r$ to h, i.e., $||\mathbf{h} - \mathbf{t}'||$. Furthermore, when the difference between $||\mathbf{h} - \mathbf{t}'||$ and $||\mathbf{h} - \mathbf{t}||$ obtains minimum, the difference between them is exactly the margin. As a result, $\min(||\mathbf{h} - \mathbf{t}'|| - ||\mathbf{h} - \mathbf{t}||), t \in P_r, t' \in N_r$ is introduced to model the difference between the distance from P_r to h and the distance from N_r to h. Specifically, the minimum difference obtains when it takes the nearest negative entity (red rectangle in Fig. 4) and the farthest positive entity (red circle in Fig. 4). From geometry aspect, the margin of general single-step structure is the distance between two concentric spheres, illustrated in Fig. 4. This definition is kind of similar to the margin defined in Support Vector Machine [43], [44], where the margin of two classes is equal to the minimum absolute distance of any two different-class instances to the classification hyperplane. Notice that the above analysis applies to both the head entity h and the tail entity t, here we simply take head entity h as an example in the rest of the paper.

Provided that the single-step structure is a hierarchical one, it has been mentioned in [9] that the siblings are close to each other considering the natural representation of hierarchical structures. Namely, the positive entities of h should lie close to each other in a small area, since they are siblings with h as the common father. In other words, the positive entities can be enclosed in a circular sector (shaded area in Fig. 5). It means that, when separating the negative examples from the positive ones, the negative entities near the circular sector (red rectangle in Fig. 5) plays more important role, rather than the negative entities with small distance to h but actually away from the circular sector (black rectangle in Fig. 5). As a result, the nearest negative entity that used to determine margin should not only have small distance with h, but also close to the circular sector. As a result, we introduce an angle θ to model this hierarchy-constraint, where θ is the angle between the vector $\mathbf{h} - \mathbf{t}'_\mathbf{i}$ and the vector $\mathbf{h} - \mathbf{t}_\mathbf{i}$. Consequently, we applied a regularization parameter with respect to θ to force the nearest negative entity that determining the margin close to the circular sector.

Formally, for a given triple (h,r,t), M_{single} is defined as the average of m_r for different relations r related to h (or t), where m_r is the margin between P_r and N_r for the given r and h (or t). More formally, taking h as an example,

$$M_{single} = \frac{\sum_{r\in R_h} m_r}{|R_h|}, \quad (4)$$

where R_h is the set of all relations related to h and $|R_h|$ is the cardinality of R_h.

Moreover, m_r is defined according to whether r is hierarchical or not. Specifically, let H_r be the set of hierarchical relations in a knowledge graph. Then for all $t \in P_r$ and $t' \in N_r$, we define

$$m_r = \begin{cases} \min\limits_{t,t'} \sigma(||\mathbf{h} - \mathbf{t}'|| - ||\mathbf{h} - \mathbf{t}||), & r \notin H_r \\ \min\limits_{t,t'} \sigma(||\mathbf{h} - \mathbf{t}'|| - ||\mathbf{h} - \mathbf{t}||) + \lambda_{hr}\phi(\theta), & r \in H_r \end{cases}, \quad (5)$$

where

$$\sigma(x) = \begin{cases} x & \text{when } x \geq 0; \\ -x & \text{otherwise.} \end{cases} \quad (6)$$

returns the absolute value of x. Here, θ is the angle between the two vectors $\mathbf{h} - \mathbf{t}$ and $\mathbf{h} - \mathbf{t}'$ in the vector space $\mathbb{R}^d$. $\phi(\theta) = 1 - \cos\theta$ is penalty function which is monotonically increasing with respect to θ, so that the penalty increases when θ becomes larger, and approximates zero when θ is close to zero. λ_{hr} is a regularization parameter to leverage the penalty, which satisfies $0 \leq \lambda_{hr} \leq 1$. More formally,

$$\lambda_{hr} = \exp\left[-\frac{1}{|E_{h,r}|}\right], \quad (7)$$

where $E_{h,r}$ is the set of tail entities of the given h and r, and $|E_{h,r}|$ is the cardinality of $E_{h,r}$. Notice that λ_{hr} is monotonically increasing with respect to $|E_{h,r}|$, which means the larger $|E_{h,r}|$ is, the more the siblings t has, leading to increasing penalty. In particular, when $N_r = \emptyset$, we set $m_r = 0$, which is reasonable since all positive entities are within the internal sphere. And when $P_r = \emptyset$, we set $m_r = \min_{t'}||\mathbf{h} - \mathbf{t}'||$ to move the negative entities away.

Actually, the margin with penalty of θ can be regarded as soft margin, which is similar to the margin defined in Support Vector Machine, to avoid overfitting. Namely, the negative examples with small distance but large θ is not fairly close to the positive entities, and the errors caused by them are allowed while fitting the model.

4.3 Multi-Step Specific Margin

To define the multi-step specific margin for a given triple (h, r, t), the relation paths connecting h and t are extracted, and are used to generate corresponding multi-step structures with r as consistent relation. Above all, let us give some notations. For a given multi-step relation path p, the set of positive relations, denoted by P_p, contains the consistent relations r. If p connects h and t, the positive relations r connect them as well, which form the golden triples (h, r, t) in the knowledge graph. On the contrary, the set of negative relations, denoted by N_p, contains the relations r' that are not consistent with p. If p connects h and t, the negative relations r' is obtained by replacing the relation r of golden triples (h, r, t), such that the corrupted triple (h, r', t) does not exist in the knowledge graph.

Provided that the multi-step structure is a general one, there are three conditions, according to the hierarchical properties of the relation paths and the consistent relations. 1) First, supposing that the path p is non-hierarchical relation path and the consistent relation r is non-hierarchical relation, they should be close to each other in the embedding space [11], i.e., $\mathbf{p} \approx \mathbf{r}$, since they connect the same entities, and both

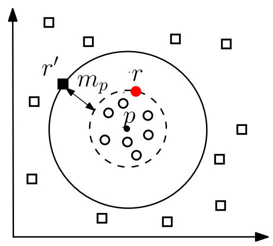


Fig. 6. The illustration of m_p, where circles stand for positive relations and rectangles represent negative relations in the embedding space $\mathbb{R}^d$.

can be regarded as translation from the same head to same tail entities. As a result, for a given p, the positive relations $r \in P_p$ should be closer to p than the negative relations $r' \in N_p$. Similar to general single-step specific margin, we introduce $\min(||\mathbf{r}' - \mathbf{p}|| - ||\mathbf{r} - \mathbf{p}||), r \in P_p, r' \in N_p$ to model the margin between positive and negative relations. Hence, the margin is determined by the nearest negative relation (red square in Fig. 6) and furtherest positive relation (red circle in Fig. 6). From geometry aspect, the positive relations (circles in Fig. 6) are constrained within the internal sphere, along with the negative relations (squares in Fig. 6) outside the external sphere. Hence, as illustrated in Fig. 6, the general multi-step specific margin should be the distance between two concentric spheres in the embedding space. 2) Second, supposing that the path is hierarchical but the relation is not, owing to the existence of inverse relation, it is possible that the two entities connected by the path are on the same layer, e.g., the multi-step structure in Fig. 2e, 2j. As a result, there is little difference whether the relation path is hierarchical or not. In this case, the margin is the same as above. 3) Third, supposing that the relation is hierarchical but the path is not, the relation forces head and tail entities to be on different layers, which is conflict with the hierarchical property of path. Hence, this case can be regarded as no positive relations. Namely, the margin calculation is the same as above, except clearing the positive relations before calculation.

Provided that the multi-step structure is a hierarchical one, the path and the consistent relation are both hierarchical. Since the hierarchical relations force the head and tail entities to be on different layers, the consistent relation paths should be hierarchical, which enables the head and tail entities to be on different layers. That is to say, the hierarchical relation paths are highly correlated to hierarchical relations, and are indispensable for the existence of hierarchical relations. Hence, the hierarchical relation paths should be paid more attention than non-hierarchical ones when detecting the optimal margin. To this end, we introduce μ_p, which is the proportion of hierarchical relations to all relations contained in p. More formally,

$$\mu_p = \frac{|\{r : r \in p \wedge r \in H_r\}|}{|\{r : r \in p\}|}, \quad (8)$$

where $|\cdot|$ is the cardinality of the set. For instance, the relation path in Fig. 2a, $p = \left(\xrightarrow{child} \cdot \xrightarrow{child}\right)$, is hierarchical relation path, and the consistent relation *grandchild* is hierarchical as well. As there are two hierarchical relations in this path, $\mu_p = \frac{2}{2} = 1$. Similarly, Fig. 2b, 2c, 2d, 2f, 2g, 2h, 2i are hierarchical relation paths, and have hierarchical consistent relations, μ_p equals 1, 1, 0.5, 0.5, 0.67, 0.67, 0.67 respectively.

Formally, for a given triple (h,r,t), the multi-step specific margin M_{multi} is the weighted average of m_p for different paths p consistent to r, where m_p denote the margin between P_p and N_p. More formally,

$$M_{multi} = \frac{\sum_{p \in Path_{h,t}} a_{pr} m_p}{\sum_{p \in Path_{h,t}} a_{pr}}, \tag{9}$$

where

$$a_{pr} = \begin{cases} R(p|h,t), & p \in H_p \wedge r \in H_r \\ (1+\mu_p)R(p|h,t), & p \notin H_p \vee r \notin H_r \end{cases}, \tag{10}$$

where $Path_{h,t}$ is the set of relation paths connecting h and t, and H_p denotes the set of hierarchical relation paths in the knowledge graph, with H_r the set of hierarchical relations in knowledge graphs.

Specifically, m_p is defined as follows. Given r and p, for all $r \in P_p$ and $r' \in N_p$,

$$m_p = \min_{r,r'} \sigma(||\mathbf{r}' - \mathbf{p}|| - ||\mathbf{r} - \mathbf{p}||), \tag{11}$$

where

$$\sigma(x) = \begin{cases} x & \text{when } x \geq 0; \\ -x & \text{otherwise.} \end{cases} \tag{12}$$

returns the absolute value of x. And $\mathbf{r}, \mathbf{r}', \mathbf{p} \in \mathbb{R}^d$ denote the embedding vectors of r, r', p respectively. Note that the embedding vector of p can be composed of the embedding vectors of entities following [11]. In this paper, the add operator is adopted as it achieved the best performance in [11]. Namely, $\mathbf{p} = \mathbf{h} + \mathbf{r_1} + \mathbf{r_2} + \cdots + \mathbf{t}$. In particular, similar to single-step specific margin, we set $m_p = 0$ if $N_p = \emptyset$, and set $m_p = \min_{r'} ||\mathbf{r}' - \mathbf{p}||$ when $P_p = \emptyset$.

Besides, $R(p|h,t)$ represents the reliability of a relation path p for the given entities h and t, and it is determined by the resource amount that eventually flows to the tail entity t from the head entity h along the path p by the path-constraint resource allocation algorithm following [11]. More precisely, for a relation path $p = \left(e_0 \xrightarrow{r_1} e_1 \xrightarrow{r_2} e_2 \xrightarrow{r_3} \cdots \xrightarrow{r_1} e_1\right)$, the resource flowing to a entity m, which is connected by the path p, is defined as follows,

$$R_p(m) = \sum_{n \in S_{i-1}(\cdot,m)} \frac{1}{|S_i(n,\cdot)|} R_p(n), \tag{13}$$

where $S_{i-1}(\cdot,m)$ is the direct predecessors of m along relation r_i, and $S_i(n,\cdot)$ represents the direct successors of n along relation r_i. Note that for the path between h and t, $e_0 = h$ and $e_l = t$. Besides, the initial resource of h is set to 1 in general, i.e., $R_p(h) = 1$. In the meanwhile, the reliability of a relation path p is measured by resource amount flows to t, i.e., $R(p|h,t) = R_p(t)$.

Furthermore, another application of the reliability score is to filter the unreliable paths, since candidate paths can be numerous, and the unreliable path may even drag down the predictive performance. Similar path selection techniques can be found in [45], [46], [47].

5 Analysis of hTransM

5.1 The Convergence of hTransM

The convergence of hTransM is analyzed by studying the uniform stability instead of directly proving the uniform convergence following [10], since it has been proved that uniform stability is a sufficient condition for learnability of learning problem [48].

To show the effectiveness of a learning algorithm $\mathcal{A}$, the generalization error, i.e., true risk, is used generally, which can not be calculate directly and often approximated by the empirical risk. Let the training data set is denoted by $S = \{(h_1,r_1,t_1),\ldots,(h_i,r_i,t_i),\ldots,(h_n,r_n,t_n)\}$, where n is the size of the training set. $\mathcal{R}_{emp}(\mathcal{A},S)$ stands for the empirical risk, and $\mathcal{R}(\mathcal{A},S)$ for the true risk. Provided that the training data S is drawn independent and identically distributed (i.e., i.i.d.) from an unknown distribution $\mathcal{D}$, hTransM is said to be convergent if the empirical risk $\mathcal{R}_{emp}(\mathcal{A},S)$ converges to the true risk $\mathcal{R}(\mathcal{A},S)$, where

$$\mathcal{R}(\mathcal{A},S) = \mathbb{E}_z[\mathcal{L}(\mathcal{A},z)], \tag{14}$$

$$\mathcal{R}_{emp}(\mathcal{A},S) = \frac{1}{n}\sum_{k=1}^{n} \mathcal{L}(\mathcal{A},z_k), \tag{15}$$

where $z = (h,r,t)$ is a triple sampled according to $\mathcal{D}$, $z_k = (h_k,r_k,t_k)$ is the k-th element of S, $k \in \{1,2,\cdots,n\}$, and $\mathbb{E}_z[\cdot]$ denotes the expectation. To this end, we define the Uniform-Replace-One stability motivated by [48].

Definition 7. *The learning algorithm $\mathcal{A}$ has Uniform-Replace-One stability γ with respect to the loss function $\mathcal{L}$ for $i \in \{1,2,\ldots,n\}$, the following inequality holds*

$$||\mathcal{L}(\mathcal{A}_S,\cdot) - \mathcal{L}(\mathcal{A}_{S^i},\cdot)||_\infty \leq \gamma, \tag{16}$$

where

$$S^i = \{S \backslash (h_i,r_i,t_i) \cup (h'_i,r_i,t'_i)\}, \tag{17}$$

Here, $\mathcal{A}_S$ means that the learning algorithm $\mathcal{A}$ is trained on the data set S, and $\|\cdot\|_\infty$ is the maximum norm. h'_i and t'_i are the corrupted entities. The loss function $\mathcal{L}$ of hTransM takes the form

$$\mathcal{L}(\mathcal{A}_S,z) = f_r(h,t) + M_{opt} - f_r(h',t'), \tag{18}$$

from which we have the following lemma.

Lemma 8. *The Uniform-Replace-One stability γ of hTransM with respect to the given loss function $\mathcal{L}(\mathcal{A}_S,z)$ is equal to $2\hat{f}_r + 2\hat{R}$, where $\hat{f}_r = \max_{h,t} f_r(h,t)$ is the maximum over the triples $(h,r,t) \in S$, and $\hat{R} = 2R_{ent} + 2R_{rel} + 2$ with R_{ent} be the radius of the smallest sphere containing the learning entities, and R_{rel} be the radius of the smallest sphere containing the learning relations .*

Proof. By Eqs. (16) and (18) we deduce that

$$\begin{aligned} ||\mathcal{L}(\mathcal{A}_S,\cdot) - \mathcal{L}(\mathcal{A}_{S^i},\cdot)||_\infty &= \max_{z_i} |\mathcal{L}(\mathcal{A}_S,z_i) - \mathcal{L}(\mathcal{A}_{S^i},z_i)| \\ &= |f_r(h,t) + M_{opt} - f_r(h',t') \\ &\quad - (f_r(h,t) + M'_{opt} - f_r(h'',t''))| \\ &\leq |f_r(h'',t'') - f_r(h',t')| + |M_{opt} - M'_{opt}| \\ &\leq |f_r(h'',t'')| + |f_r(h',t')| + |M_{opt}| + |M'_{opt}| \\ &\leq 2\max_{h,t} f_r(h,t) + |M_{opt}| + |M'_{opt}|, \end{aligned}$$

TABLE 1
Numbers of Parameters and Their Values

Method	Model complexity	on FB15K
RESCAL	$\mathcal{O}(dn_e + n_r d^2)$	14.9M
TransE	$\mathcal{O}(dn_e + dn_r)$	1.6M
TransH	$\mathcal{O}(dn_e + 2dn_r)$	1.8 M
TransR	$\mathcal{O}(dn_e + dn_r + n_r d^2)$	15.1M
PTransE	$\mathcal{O}(dn_e + dn_r)$	1.6M
TransA	$\mathcal{O}(dn_e + dn_r)$	1.6M
hTransM	$\mathcal{O}(dn_e + dn_r)$	1.6M

where h', t' are the corrupted entities for $\mathcal{L}(\mathcal{A}_S, \cdot)$ and h'', t'' for $\mathcal{L}(\mathcal{A}_{S^i}, z_i)$. By Eq.(5) and definition of R_{ent}, we can deduce that

$$m_r \leq \sigma(||\mathbf{h} - \mathbf{t}'|| - ||\mathbf{h} - \mathbf{t}||) + \lambda_{hr}\phi(\theta) \leq 2R_{ent} + 2,$$

since $0 \leq \lambda_{hr} \leq 1$ and $0 \leq \phi(\theta) \leq 2$. Namely, $m_r \leq \hat{R}$. Similarly, by Eq.(11) and definition of R_{rel}, it can be deduced that

$$m_p \leq \sigma(||\mathbf{r}' - \mathbf{p}|| - ||\mathbf{r} - \mathbf{p}||) \leq 2R_{rel}.$$

This leads to

$$M_{opt} \leq \alpha 2R_{ent} + \beta 2R_{rel} + 2 \leq \alpha 2R_{ent} + \beta 2R_{rel} + 2,$$

i.e., $M_{opt} \leq \hat{R}$ by Eq.(3). Setting $\hat{f}_r = \max_{h,t} f_r(h,t)$ completes the proof. □

Then, the difference between two risks can be derived.

Theorem 9. *For the embedding method $\mathcal{A}$ with Uniform-Replace-One stability γ with respect to the given loss function $\mathcal{L}$, we have the following inequality with probability at least $1 - \delta$,*

$$\mathcal{R}(\mathcal{A}, S) \leq \mathcal{R}_{emp}(\mathcal{A}, S) + \sqrt{\frac{(\hat{R} + \hat{f}_r)^2}{2n\delta} + \frac{6\hat{f}_r(\hat{R} + \hat{f}_r)}{\delta}}, \quad (19)$$

Before proving Theorem 9, we first present the following Lemma verified in [48].

Lemma 10. *For any algorithm $\mathcal{A}$ and loss function $\mathcal{L}(\mathcal{A}_S, z)$ such that $0 \leq \mathcal{L}(\mathcal{A}_S, z) \leq \hat{L}$, set $z_i = (h_i, r_i, t_i) \in S$, we have for any different $i, j \in \{1, 2, \ldots, n\}$ that*

$$\begin{aligned} &\mathbb{E}_S\left[\left(\mathcal{R}(\mathcal{A}, S) - \mathcal{R}_{emp}(\mathcal{A}, S)\right)^2\right] \\ &\leq \frac{\hat{L}^2}{2n} + 3\hat{L}\mathbb{E}_{S \cup z_i'}[|\mathcal{L}(\mathcal{A}_S, z_i) - \mathcal{L}(\mathcal{A}_{S^i}, z_i)|]. \end{aligned}$$

Proof of Theorem 9. First, from Eq.(18), we deduce that

$$\mathcal{L}(\mathcal{A}_S, z) \leq M_{opt} + f_r(h,t) \leq \hat{R} + \hat{f}_r.$$

Then from Definition 7, we find that

$$\mathbb{E}_{S \cup z_i'}[|\mathcal{L}(\mathcal{A}_S, z_i) - \mathcal{L}(\mathcal{A}_{S^i}, z_i)|] \leq \gamma.$$

Hence, by Lemmas 10 and 8, we obtain that

$$\mathbb{E}_S\left[\left(\mathcal{R}(\mathcal{A}, S) - \mathcal{R}_{emp}(\mathcal{A}, S)\right)^2\right] \leq \frac{(\hat{R} + \hat{f}_r)^2}{2n} + 6(\hat{R} + \hat{f}_r)\hat{f}_r$$

TABLE 2
The Datasets

Datasets	# Relation	# Entitiy	#Train	#Valid	#Test
WN18	18	40,943	141,442	5,000	5,000
FB15K	1,345	14,951	483,142	50,000	59,071
FAMILY	7	721	8,461	2,820	2,821

By Chebyshev's inequality, it can be derived that

$$\begin{aligned} &Prob(\left(\mathcal{R}(\mathcal{A}, S) - \mathcal{R}_{emp}(\mathcal{A}, S)\right) \geq \epsilon) \\ &\leq \frac{\mathbb{E}_S\left[\left(\mathcal{R}(\mathcal{A}, S) - \mathcal{R}_{emp}(\mathcal{A}, S)\right)^2\right]}{\epsilon^2} \\ &\leq \left(\frac{(\hat{R} + \hat{f}_r)^2}{2n} + 6(\hat{R} + \hat{f}_r)\hat{f}_r\right) \cdot \frac{1}{\epsilon^2}. \end{aligned}$$

Let the right hand side of the above inequality be δ, then we have with probability at least $1 - \delta$ that

$$\mathcal{R}(\mathcal{A}, S) \leq \mathcal{R}_{emp}(\mathcal{A}, S) + \sqrt{\frac{(\hat{R} + \hat{f}_r)^2}{2n\delta} + \frac{6\hat{f}_r(\hat{R} + \hat{f}_r)}{\delta}}.$$

This completes the proof. □

5.2 Model Complexity of hTransM

The model complexity of hTransM is studied from the aspect of model parameters, which has been widely used to evaluate the complexity of knowledge graph embedding methods [9], [49], [24]. The number of parameters of hTransM is the same as other simple embedding methods, since the main parameters of hTransM is the embedding vectors for each entity and each relation. As a result, the number of model parameters of hTransM is $\mathcal{O}(dn_e + dn_r)$, where n_e denotes the number of entities in a knowledge graph, and n_r represents the number of relations in a knowledge graph. d stands for the embedding dimension.

The comparison and the values for a typical knowledge graph FB15K (in millions) are illustrated in detail in Table 1, where the embedding dimension is set to 100 for all methods. Besides, the statistics of the FB15K is shown in Table 2.

6 Experiments

The experiments are carried out on three public knowledge graphs, WN18 introduced in [9], FB15K introduced in [9] and FAMILY introduced in [50]. WN18 and FB15K are subsets of the widely used knowledge graph WordNet and Freebase respectively. FAMILY is an artificial hierarchical knowledge graph expressing family relationships among the members of 5 families along 6 generations, where entities are organized in a layered tree. The statistics of the datasets are listed in Table 2.

To detect the hierarchical relations, first the relations are classified into *1-to-1*, *1-to-N*, *N-to-1*, *N-to-N* and the proportion of the four classes are 25.5, 17.4, 30.9, 26.2 percent on WN18, 31.3, 27.2, 21.5, 20.0 percent on FB15K, and 0.3, 32.0, 19.0, 48.7 percent on FAMILY. Note that FAMILY is constructed to capture the hierarchical structures in the knowledge graph, thus the *1-to-1* is little. Second, we find

TABLE 3
Evaluation Results on Entity Prediction

Metric	WN18				FB15K				FAMILY			
	Mean Rank		HITS@10		Mean Rank		HITS@10		Mean Rank		HITS@10	
	Raw	Filter	Raw	Filter	Raw	Filter	Raw	Filter	Raw	Filter	Raw	Filter
Unstructured	315	304	35.3	38.2	1,074	979	4.5	6.3	374	357	-	-
RESCAL	1,180	1,163	37.2	52.8	828	683	28.4	44.1	-	-	-	-
SE	1,011	985	273	68.5	80.5	162	28.8	39.8	362	351	-	-
SME(linear)	545	533	65.1	74.1	274	154	30.7	40.8	26	9	-	-
SME(bilinear)	526	509	54.7	61.3	284	158	31.3	41.3	29	12	-	-
TransE	263	251	75.4	89.2	243	125	34.9	47.1	29	8	57.3	87.9
TransH(bern)	401	388	73.0	82.3	212	87	45.7	64.4	-	-	-	-
TransH(unif)	318	303	75.4	86.7	211	84	42.5	58.5	-	-	-	-
TransR(bern)	238	225	79.8	92.0	198	77	48.2	68.7	25	7	60.8	90.2
TransR(unif)	232	219	78.3	91.7	226	78	43.8	65.5	25	7	58.9	88.7
PTransE(2-step)	245	237	80.2	93.4	200	54	51.8	83.4	24	7	67.9	94.9
PTransE(3-step)	245	238	79.9	92.8	207	58	50.6	82.2	24	7	66.3	93.8
TKRL (RHE)	-	-	-	-	184	68	49.2	69.4	-	-	-	-
TKRL (RHE+STC)	-	-	-	-	202	89	50.4	73.1	-	-	-	-
TransA	165	153	-	-	164	58	-	-	23	6	-	-
hTransM	139	124	80.7	94.3	161	46	50.5	74.4	19	5	67.8	95.2

the non-circle relations and irreflexive relations, and hierarchical relation set is the intersection of non-circle relations, irreflexive relations and unbalanced *1-to-N*, *N-to-1* and *N-to-N* relations. After that, the hierarchical relation proportion is 77.8 percent on WN18, 53.8 percent on FB15K, and 42.9 percent on FAMILY. Experiments are conducted on two sub-tasks of link prediction, i.e., Entity Prediction and Relation Prediction.

6.1 Entity Prediction

This task aims to predict the missing entities h or t for a triple (h, r, t). Namely, it predicts t given $(h, r, \cdot)$ or predict h given $(\cdot, r, t)$. Similar to the setting in [9],[11], [10], a list of candidate entities is returned in terms of the score function Eq. (2) of hTransM. Mean Rank and HITS@10 are adopted as the evaluation measure. Mean Rank is the average rank of correct entities in original triples, and HITS@10 is the proportion of correct entities ranked in the top 10. It is clear that a good predictor has low Mean Rank and higher HITS@10. This is called "Raw" setting. We also filter out the corrupted triples which are correct ones for evaluation following [9], which called "Filter" setting. Namely, if a corrupted triple exists in the knowledge graph, it is also acceptable to rank it ahead the original triples. To eliminate this case, the "Filter" setting is more preferred [11].

The baseline methods include classical embedding methods as shown in Table 3. Since the datasets WN18 and FB15K are also used by the baseline methods, the experimental results are compared with those reported in their papers. Note that the results of FAMILY of those baselines are obtained by employing the publicly available code, as no results are reported in the papers.

For the parameter tuning process, we determine their values in the same way as the existing methods. The learning rate η during the SGD process is selected among $\{0.1, 0.01, 0.001\}$, the embedding dimension d in $\{50, 100, 200, 300\}$, the batch size B among $\{120, 480, 1440, 4800\}$, and parameters α and β in $[0, 1]$. All parameters are determined on the validation set. Specifically, for hTransM, the optimal settings are: $\eta = 0.001$, $d = 200$, $B = 1440$, $\alpha = 0.1$, $\beta = 0.2$ on WN18, and taking L_1 as dissimilarity. $\eta = 0.001$, $d = 300$, $B = 1440$, $\alpha = 0.0$, $\beta = 0.3$ on FB15K, as well as taking L_1 as dissimilarity. $\eta = 0.001$, $d = 50$, $B = 120$, $\alpha = 0.7$, $\beta = 0.1$ on FAMILY, as well as taking L_1 as dissimilarity.

It can tell from Table 3 that (1) On Mean Rank, hTransM obtains the lowest Mean Rank on all datasets, and performs best among all baselines. (2) On HITS@10, hTransM outperforms all baselines on WN18, and outperforms all baselines except PTransE on FB15K and FAMILY. It makes sense since that PTransE uses path information, which exists for all triples, but hTransM uses hierarchical structures, which only have about 50 percent hierarchical relation proportion on these two datasets. (3) The performance on WN18 is better than FB15K and FAMILY. It is unsurprising since the proportion of hierarchical relations are far larger than FB15K and FAMILY, which proves that the method will perform better with more hierarchical information, and produce restrained performance if the dataset is fundamentally non-hierarchical.

For further analysis, we calculated the predictive results according to different relation types, i.e., hierarchical and non-hierarchical relations, respectively. Since TransA performs best in Table 3, the detailed results on three datasets are just compared with TransA, as listed in Table 4.

TABLE 4
Filtered Mean Rank of Different Relation Types

Metric		WN18		FB15K		FAMILY	
		hie	non-hie	hie	non-hie	hie	non-hie
All	TransA	168	445	103	135	5.7	177
	hTransM	136	415	80	111	3.9	122
Head	TransA	133	478	113	129	4.4	177
	hTransM	123	419	87	108	3.1	122
Tail	TransA	203	411	94	141	6.8	177
	hTransM	151	410	73	113	4.8	122

TABLE 5
Evaluation Results on Relation Prediction

Metric	Mean Rank		HITS@1	
	Raw	Filter	Raw	Filter
TransE	2.8	2.5	65.1	84.3
TransR	2.5	2.1	70.2	91.6
PTransE(2-step)	1.7	1.2	69.5	93.6
PTransE(3-step)	1.8	1.4	68.5	94.0
TKRL (RHE)	2.12	1.73	71.1	92.8
TKRL (RHE+STC)	2.38	1.97	68.7	90.7
TransA	1.5	1.1	-	-
hTransM	**1.5**	**1.0**	71.3	93.1

Note that the filter Mean Rank is adopted to do further analyzing, as the "Filter" setting is more comport with the fact. Besides, "hie" represents hierarchical relations, and "non-hie" represents non-hierarchical relations.

It can be seen from Table 4 that the decreases of Mean Rank on different types of relations are different. hTransM achieves larger decrease on hierarchical relations than non-hierarchical ones, which makes sense since when given a hierarchical relation, it is more possible that the layer of predicted entity is different from the layer of the given entity. Besides, the decrement of Mean Rank is different on predicting head and tail entities, i.e., the decrement is larger on the worse side of TransA, especially on WN18. For instance, TransA perform worse on predicting tail for hierarchical relations, as well as predicting head for non-hierarchical relations, while hTransM decrease far more on these two sides than the other sides. It is caused by the unbalanced mapping properties of relations, and the worse predicting side of TransA is the side with large cardinality, which demonstrates that hTransM can handle relations with unbalanced mapping properties.

6.2 Relation Prediction

This task aims to predict the missing relation r for two given entities $(h, \cdot, t)$. Similar to the setting in [11], given entity pair (h, t), relation prediction returns a list of candidate relations, and Mean Rank is adopted as the evaluation measure, which is the average rank of correct relations in original triple. The corrupted triples which are correct ones are filtered out for evaluation following [9] similar to entity prediction sub-task.

The baseline methods for comparison include TransE [9], TransA [10], TransR [25] and PTransE [11], which it can be seen that the performance of these methods are fairly better than the others in entity prediction sub-task, thus are adopted as baselines in this sub-task. FB15K is adopted as datasets. Besides, the relations on FAMILY and WN18 are so little that performance is very good for all methods, thus is not adopted in this experiment. Since the results of entity prediction of these baselines are not reported in the paper, we employ the publicly available code of them to obtain the results. The parameter setting is the same as the entity prediction.

The results in Table 5 indicate that hTransM outperforms all baselines except PTransE in HITS@1(filter), which is better than entity prediction. It makes sense that the hierarchical structures, which hTransM employs, can bring more room for the improvement of relation prediction, since the entities on different layers are supposed to connect by hierarchical relations. Furthermore, the result that TransA performs better that other baselines, also demonstrates the superiority of detecting the optimal margin adaptively.

6.3 Discussion about Hierarchy-Contrained Margin

To better understand how the margin affects the predictive method, the value of the optimal hierarchy-constrained margin is plotted along with the iterations of SGD. Since TransA is another method adaptively determine the optimal margin for each triples, in order to compare the optimal margin generated by both methods, the experiment is conducted by hTransM and TransA. The dataset adopts WN18, FB15K and FAMILY without loss of generality. To avoid randomness, the margin is the average margin of all triples in a iteration.

The process of margin variation is shown in Fig. 7, where the x-axis represents the number of SGD iterations in the training process, and the y-axis represents the value of the optimal margin. Note that since a negative sampling procedure is carried out when generating the optimal margin, there exists randomness in Fig. 7. As the number of triples in FAMILY is very small, the randomness of negative sampling on FAMILY is much more obvious than the other two datasets. There are three observations from Fig. 7 as follows.

First, it is verified that the optimal margin becomes larger with the iterations of SGD increasing, which implies that adaptively choosing margin is consistent with the intrinsic

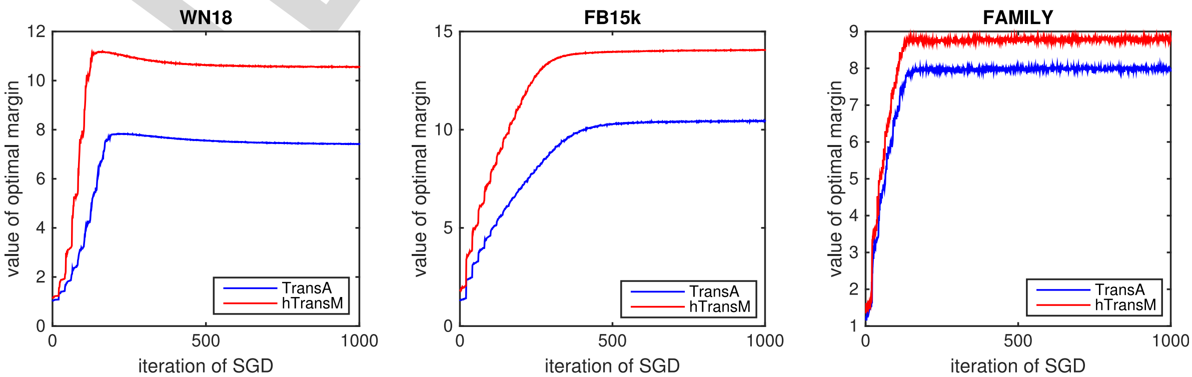


Fig. 7. The impact of the number of SGD iterations on the single-step margin of one entity on FAMILY, where x-axis stands for the number of sampling the triple in the training process, as well as the y-axis for the value of the margin.

characteristic of the margin-based knowledge graph embedding methods. It makes sense that the positive and negative examples have growing distance along with the optimization processing. Consequently, determining the optimal margin adaptively is of great assistance to the performance of link prediction.

Second, it can tell that the hierarchy-constrained margin of hTransM is larger than general margin of TransA, and increase more rapidly as well. It validates that hierarchy-constrained margin can be regarded as soft margin, and it is effective by integrating the hierarchical information into the predictive method.

Third, the value and the convergence of optimal margin vary from different datasets. The earlier the convergence is, the faster the value of margin increases, and there will be a small decrease if the convergence comes very early, such as WN18.

7 Conclusion

In this paper, we study the link prediction problem on knowledge graphs. To make full use of the hierarchical structures of knowledge graphs, this paper proposes a hierarchy-constrained link prediction method based on knowledge graph embedding, called hTransM. It determines the margin adaptively to achieve optimal predictive performance. The margin is modelled by discovering hierarchical structures automatically, and dividing them into the single-step hierarchical structures and multi-step hierarchical structures, which contributes to the optimal single-step margin and optimal multi-step margin. In addition, the methods could be scaled to other margin-based translation embedding methods, such as TransH, TransR, etc., on account of the effectiveness of the optimal margin.

Acknowledgments

This work is supported by National Grand Fundamental Research 973 Program of China (No. 2013CB329602, 2014CB340401), National Natural Science Foundation of China (No.61572469, 61402442, 61572473, 61303244, 61402022, 61572467), Beijing nova program (No.Z121101002512063), and Beijing Natural Science Foundation (No. 4154086).

References

[1] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, "Freebase: A collaboratively created graph database for structuring human knowledge," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2008, pp. 1247–1250.

[2] G. A. Miller, "Wordnet: A lexical database for English," *Commun. ACM*, vol. 38, no. 11, pp. 39–41, 1995.

[3] Y. Jia, Y. Wang, X. Cheng, X. Jin, and J. Guo, "Openkn: An open knowledge computational engine for network big data," in *Proc. IEEE/ACM Int. Conf. Adv. Soc. Netw. Anal. Mining*, 2014, pp. 657–664.

[4] W. Wu, H. Li, H. Wang, and K. Q. Zhu, "Probase: A probabilistic taxonomy for text understanding," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2012, pp. 481–492.

[5] D. Liu, Y. Wang, Y. Jia, J. Li, and Z. Yu, "Lsdh: A hashing approach for large-scale link prediction in microblogs," in *Proc. 28th AAAI Conf. Artif. Intell.*, 2014, pp. 3120–3121.

[6] Y.-T. Jia, Y.-Z. Wang, and X.-Q. Cheng, "Learning to predict links by integrating structure and interaction information in microblogs," *J. Comput. Sci. Technol.*, vol. 30, no. 4, pp. 829–842, 2015.

[7] H. Huang, J. Tang, L. Liu, J. Luo, and X. Fu, "Triadic closure pattern analysis and prediction in social networks," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 12, pp. 3374–3389, Dec. 2015.

[8] J. Zhang, Z. Fang, W. Chen, and J. Tang, "Diffusion of following links in microblogging networks," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 8, pp. 2093–2106, Aug. 2015.

[9] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," in *Proc. 26th Int. Conf. Neural Inf. Process. Syst.*, 2013, pp. 2787–2795.

[10] Y. Jia, Y. Wang, H. Lin, X. Jin, and X. Cheng, "Locally adaptive translation for knowledge graph embedding," in *Proc. 30th AAAI Conf. Artif. Intell.*, 2016, pp. 992–998.

[11] Y. Lin, Z. Liu, H. Luan, M. Sun, S. Rao, and S. Liu, "Modeling relation paths for representation learning of knowledge bases," in *Proc. 2015 Conf. Empirical Methods Natural Lang. Process.*, 2015, pp. 705–714.

[12] A. Bordes, N. Usunier, A. García-Durán, J. Weston, and O. Yakhnenko, "Irreflexive and hierarchical relations as translations," *ICML 2013 Workshop Struct. Learn.: Inferring Graphs Struct. Unstruct. Inputs*, 2013, p. 5.

[13] J. R. Quinlan and R. M. Cameron-Jones , "Foil: A midterm report," in *Proc. Conf. Mach. Learn.*, 1993, pp. 1–20.

[14] W. W. Cohen and C. D. Page, "Polynomial learnability and inductive logic programming: Methods and results," *New Generation Comput.*, vol. 13, no. 3/4, pp. 369–409, 1995.

[15] T. M. Mitchell, J. Betteridge, A. Carlson, E. Hruschka, and R. Wang, *Populating the Semantic Web by Macro-Reading Internet Text*. New York, NY, USA: Springer, 2009.

[16] S. Schoenmackers, O. Etzioni, D. S. Weld, and J. Davis, "Learning first-order horn clauses from web text," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2010, pp. 1088–1098.

[17] N. Lao, T. Mitchell, and W. W. Cohen, "Random walk inference and learning in a large scale knowledge base," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2011, pp. 529–539.

[18] N. Lao, A. Subramanya, F. Pereira, and W. W. Cohen, "Reading the web with learned syntactic-semantic inference rules," in *Proc. Joint Conf. Empirical Methods Natural Lang. Process. Comput. Natural Lang. Learn.*, 2012, pp. 1017–1026.

[19] Z. Zhao, Y. Jia, and Y. Wang, "Content-structural relation inference in knowledge base," in *Proc. 28th AAAI Conf. Artif. Intell.*, 2014, pp. 3154–3155.

[20] M. Li, Y. Jia, Y. Wang, Z. Zhao, and X. Cheng, "Predicting links and their building time: A path-based approach," in *Proc. 13th AAAI Conf. Artif. Intell.*, 2016, pp. 4228–4229.

[21] A. Neelakantan, B. Roth, and A. Mccallum, "Compositional vector space models for knowledge base completion," in *Proc. 53rd Ann. Meeting Assoc. Comput. Linguistics and 7th Int. Joint Conf. Natural Lang. Process.*, 2015, pp. 156–166.

[22] N. Lao and W. W. Cohen, "Relational retrieval using a combination of path-constrained random walks," *Mach. Learn.*, vol. 81, no. 1, pp. 53–67, 2010.

[23] M. Gardner, P. P. Talukdar, B. Kisiel, and T. Mitchell, "Improving learning and inference in a large knowledge-base using latent syntactic cues," *Americas*, vol. 70, no. 2, pp. 319–320, 2013.

[24] Z. Wang, J. Zhang, J. Feng, and Z. Chen, "Knowledge graph embedding by translating on hyperplanes," in *Proc. 28th AAAI Conf. Artif. Intell.*, 2014, pp. 1112–1119.

[25] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu, "Learning entity and relation embeddings for knowledge graph completion," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 2181–2187.

[26] G. Ji, S. He, L. Xu, K. Liu, and J. Zhao, "Knowledge graph embedding via dynamic mapping matrix," in *Proc. 53rd Annu. Meeting Assoc. Comput. Linguistics/7th Int. Joint Conf. Natural Lang. Process. (Vol. 1: Long Papers)*, 2015, pp. 687–696.

[27] M. Fan, Q. Zhou, E. Chang, and T. F. Zheng, "Transition-based knowledge graph embedding with relational mapping properties," in *Proc. 28th Pacific Asia Conf. Lang. Inf. Comput.*, 2014, pp. 328–337.

[28] A. Bordes, X. Glorot, J. Weston, and Y. Bengio, "Joint learning of words and meaning representations for open-text semantic parsing," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2012, pp. 127–135.

[29] A. Bordes, J. Weston, R. Collobert, and Y. Bengio, "Learning structured embeddings of knowledge bases," in *Proc. Conf. Artif. Intell.*, 2011, no. EPFL-CONF-192344, pp. 301–306.

[30] A. Bordes, X. Glorot, J. Weston, and Y. Bengio, "A semantic matching energy function for learning with multi-relational data," *Mach. Learn.*, vol. 94, no. 2, pp. 233–259, 2014.

[31] R. Jenatton, N. L. Roux, A. Bordes, and G. R. Obozinski, "A latent factor model for highly multi-relational data," in *Proc. 25th Int. Conf. Neural Inf. Process. Syst.*, 2012, pp. 3167–3175.

[32] I. Sutskever, J. B. Tenenbaum, and R. R. Salakhutdinov, "Modelling relational data using bayesian clustered tensor factorization," in *Proc. Neural Inf. Process. Syst.*, 2009, pp. 1821–1828.

[33] K. Q. Weinberger and O. Chapelle, "Large margin taxonomy embedding for document categorization," in *Proc. Neural Inf. Process. Syst.*, 2009, pp. 1737–1744.

[34] S. Bengio, J. Weston, and D. Grangier, "Label embedding trees for large multi-class tasks," in *Proc. Neural Inf. Process. Syst.*, 2010, pp. 163–171.

[35] R. Xie, Z. Liu, and M. Sun, "Representation learning of knowledge graphs with hierarchical types," in *Proc. 25th Int. Joint Conf. Artif. Intell.*, 2016, pp. 2965–2971.

[36] C. Wang, J. Han, Y. Jia, J. Tang, D. Zhang, Y. Yu, and J. Guo, "Mining advisor-advisee relationships from research publication networks," in *Proc. 16th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2010, pp. 203–212.

[37] Y.-Y. Liu, J.-J. Slotine, and A.-L. Barabási, "Control centrality and hierarchical structure in complex networks," *Plos One*, vol. 7, no. 9, 2012, Art. no. e44459.

[38] E. Jacobson and S. E. Seashore, "Communication practices in complex organizations," *J. Soc. Issues*, vol. 7, no. 3, pp. 28–40, 1951.

[39] P. Eades, Q.-W. Feng, and X. Lin, "Straight-line drawing algorithms for hierarchical graphs and clustered graphs," in *Proc. Int. Symp. Graph Drawing*, 1996, pp. 113–128.

[40] K.-K. Yan, G. Fang, N. Bhardwaj, R. P. Alexander, and M. Gerstein, "Comparing genomes to computer operating systems in terms of the topology and evolution of their regulatory control networks," *Proc. Nat. Acad. Sci.*, vol. 107, no. 20, pp. 9186–9191, 2010.

[41] P. Healy and N. S. Nikolov, "Hierarchical drawing algorithms," *Handbook of Graph Drawing and Visualization*, pp. 409–454, 2013.

[42] M. Li, Y. Jia, Y. Wang, J. Li, and X. Cheng, "Hierarchy-based link prediction in knowledge graphs," in *Proc. 25th Int. Conf. Companion World Wide Web*, 2016, pp. 77–78.

[43] V. Vapnik, *The Nature of Statistical Learning Theory*. New York, NY, USA: Springer, 2013.

[44] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proc. 5th Annu. Workshop Comput. Learning Theory*, 1992, pp. 144–152.

[45] K. Toutanova, V. Lin, W. T. Yih, H. Poon, and C. Quirk, "Compositional learning of embeddings for relation paths in knowledge base and text," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics*, 2016, pp. 1434–1444.

[46] F. Wu, J. Song, Y. Yang, X. Li, Z. M. Zhang, and Y. Zhuang, "Structured embedding via pairwise relations and long-range interactions in knowledge base," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 1663–1670.

[47] A. García-Durán, A. Bordes, and N. Usunier, "Composing relationships with translations," in *Proc. 2015 Conf. Empirical Methods Natural Lang. Process.*, 2015, pp. 286–290.

[48] O. Bousquet and A. Elisseeff, "Stability and generalization," *J. Mach. Learn. Res.*, vol. 2, pp. 499–526, 2002.

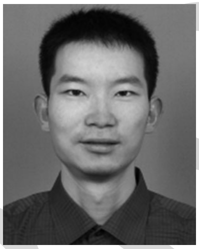
[49] M. Nickel, L. Rosasco, and T. Poggio, "Holographic embeddings of knowledge graphs," in *Proc. 13th AAAI Conf. Artif. Intell.*, 2016, pp. 1955–1961.

[50] A. Garcıa-Durán, A. Bordes, and N. Usunier, "Effective blending of two and threeway interactions for modeling multi-relational data," in *Proc. 2014th Eur. Conf. Mach. Learn. Knowl. Discovery Databases*, 2014, pp. 434–449.

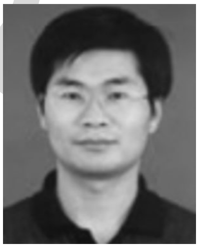
Manling Li is with the Institute of Computing Technology, Chinese Academy of Sciences. Her main research interests include knowledge graph, data mining, and natural language process, etc.

Yuanzhuo Wang received the PhD in computer science. He is a professor with the Institute of Computing Technology, Chinese Academy of Sciences. His current research interests include social computing, and open knowledge network, etc. So far he has published more than 140 papers. He is a senior member of China Computer Federation and member of the IEEE.

Denghui Zhang is with the Institute of Computing Technology, Chinese Academy of Sciences. His main research interests include knowledge graph, natural language process, and parallel computing, etc.



Yantao Jia received the PhD degree in mathematics. He is an associate professor with the Institute of Computing Technology, Chinese Academy of Sciences. His main research interests include open knowledge network, social computing, and combinatorial algorithms, etc. He is a member of the IEEE.



Xueqi Cheng is a professor with the Institute of Computing Technology, Chinese Academy of Sciences. His main research interests include network science, web search and data mining, big data processing and distributed computing architecture, and so on. He has published more than 100 publications in prestigious journals and conferences, including the *IEEE Transactions on Information Theory*, the *IEEE Transactions on Knowledge and Data Engineering*, the *Journal of Statistics Mechanics: Theory and Experiment*, the *Physical Review E.*, ACM SIGIR, WWW, ACM CIKM, WSDM, IJCAI, ICDM, and so on. He has received the Best Paper Award in CIKM (2011) and the Best Student Paper Award in SIGIR (2012). He is a member of the IEEE.