Representation Learning of Knowledge Graphs with Hierarchical Types

Ruobing Xie,[1] **Zhiyuan Liu,**[1,2*] **Maosong Sun**[1,2]
[1] Department of Computer Science and Technology,
State Key Lab on Intelligent Technology and Systems,
National Lab for Information Science and Technology, Tsinghua University, Beijing, China
[2] Jiangsu Collaborative Innovation Center for Language Ability,
Jiangsu Normal University, Xuzhou 221009 China

Abstract

Representation learning of knowledge graphs aims to encode both entities and relations into a continuous low-dimensional vector space. Most existing methods only concentrate on learning representations with structured information located in triples, regardless of the rich information located in hierarchical types of entities, which could be collected in most knowledge graphs. In this paper, we propose a novel method named Type-embodied Knowledge Representation Learning (TKRL) to take advantages of hierarchical entity types. We suggest that entities should have multiple representations in different types. More specifically, we consider hierarchical types as projection matrices for entities, with two type encoders designed to model hierarchical structures. Meanwhile, type information is also utilized as relation-specific type constraints. We evaluate our models on two tasks including knowledge graph completion and triple classification, and further explore the performances on long-tail dataset. Experimental results show that our models significantly outperform all baselines on both tasks, especially with long-tail distribution. It indicates that our models are capable of capturing hierarchical type information which is significant when constructing representations of knowledge graphs. The source code of this paper can be obtained from https://github.com/thunlp/TKRL.

1 Introduction

Knowledge graphs (KGs) such as Freebase, DBpedia and YAGO provide effective structured information and have been crucial in information retrieval and question answering. A typical KG is usually represented as multi-relational data with enormous triple facts in the form of (*head entity*, `relation`, *tail entity*), abridged as (h, r, t).

As KG size increases, KG applications become more challenging due to data sparsity and computational inefficiency. To address these problems, representation learning (RL), which aims to project both entities and relations into a continuous low-dimensional semantic space, is blooming and widely utilized in knowledge completion, fusion and inference. It significantly improves the ability of KGs in cognition and reasoning [Bordes *et al.*, 2013; Yang *et al.*, 2014; Dong *et al.*, 2014; Neelakantan *et al.*, 2015].

Many methods have been proposed on RL for KGs, among which the translation-based models are simple and effective with the state-of-the-art performances. Unfortunately, most conventional methods merely focus on the structured information in triples, paying less attention to the rich information located in hierarchical types of entities. Fig. 1 shows a triple instance combined with part of its hierarchical types sampled from Freebase, in which the solid lines indicate the most significant roles that head and tail play in this triple.

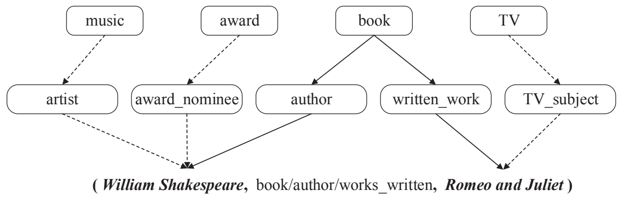

music
award
book
TV
artist
award_nominee
author
written_work
TV_subject
(William Shakespeare, book/author/works_written, Romeo and Juliet)

Figure 1: Example of entity hierarchical types in Freebase.

It is intuitive that entities possessing multiple types should have various representations in different scenarios. To take advantages of entity types, we propose a novel and effective RL method for KGs named Type-embodied Knowledge Representation Learning (TKRL). In TKRL, we follow the assumption in TransE [Bordes *et al.*, 2013], considering relations as translating operations between head and tail. For each triple (h, r, t), $\mathbf{h}$ and $\mathbf{t}$ are first projected to their corresponding type spaces in this relation as $\mathbf{h}_{rh}$ and $\mathbf{t}_{rt}$, according to the type-specific projection matrices constructed with two hierarchical type encoders. TKRL is then optimized by minimizing the energy function of $E(h, r, t) = ||\mathbf{h}_{rh} + \mathbf{r} - \mathbf{t}_{rt}||$. Moreover, type information is also considered as type constraints in both training and evaluation.

We evaluate the TKRL model on benchmark datasets of

*Corresponding author: Z. Liu (liuzy@tsinghua.edu.cn)

Freebase with knowledge graph completion and triple classification. Experimental results show that TKRL has significant and consistent improvements compared to all baselines on both tasks, especially with long-tail distribution, which confirms the capability of TKRL modeling KGs.

2 Related Work

2.1 Translation-based Models

Recent years there are great advances in representation learning for knowledge graphs. TransE [Bordes *et al.*, 2013] projects both entities and relations into the same continuous low-dimensional vector space, interpreting relations as translating operations between head and tail entities. TransE assumes that the embedding of tail $\mathbf{t}$ should be in the neighbourhood of $\mathbf{h}+\mathbf{r}$. The energy function is defined as follows:

$$E(h,r,t) = ||\mathbf{h}+\mathbf{r}-\mathbf{t}||. \quad (1)$$

TransE is effective and efficient, while the simple translating operation may lead to problems when modeling more complicated relations like 1-to-N, N-to-1 and N-to-N.

To address this issue, TransH [Wang *et al.*, 2014b] interprets relations as translating operations on relation-specific hyperplanes, allowing entities to have different distances in different relations. TransR [Lin *et al.*, 2015b] directly models entities and relations in separate entity and relation spaces, projecting entities from entity space to relation-specific space when judging the distance between entities. TransD [Ji *et al.*, 2015] proposes dynamic mapping matrix constructed via both entity and relation, considering the diversity of entities as well as relations simultaneously. Besides, [Lin *et al.*, 2015a; Gu *et al.*, 2015] also encode multiple-step relation path information into KG representation learning. However, These models only concentrate on information in triples, ignoring rich information in entity types, which will be considered in TKRL model.

2.2 Multi-source Information Learning

Multi-source information like textual information and type information, considered as supplements for the structured information embedded in triples, is significant for RL in KGs. NTN [Socher *et al.*, 2013] encodes 3-way tensors into neural network and represents an entity as the average of word embeddings in its entity name. [Wang *et al.*, 2014a; Zhong *et al.*, 2015] encode both entities and words into a joint continuous vector space by alignment models using Wikipedia anchors, entity names or entity descriptions. DKRL [Xie *et al.*, 2016] proposes description-based representations for entities constructed from entity descriptions with CBOW or CNN, which is capable of modeling entities in zero-shot scenario.

Rich information located in hierarchical entity types is also significant for KGs, while it has just attracted attention. [Krompaß *et al.*, 2015] considers entity types as hard constraints in latent variable models for KGs. However, the type information is not explicitly encoded into KG representations, and their method doesn't consider the hierarchical structure of entity types. Moreover, hard constraints may have issues with noises and incompleteness in type information, which is pretty common in real-world KGs. Therefore, we propose the TKRL model to overcome these shortages. To the best of our knowledge, TKRL is the first method which explicitly encodes type information into multiple representations in KGs with the help of hierarchical structures.

3 Methodology

To utilize rich information located in entity types, we take type information into consideration when constructing projection matrices of entities, with two hierarchical type encoders modeling hierarchical structures. Moreover, type information is also utilized as relation-specific type constraints in both training and evaluation.

3.1 Hierarchical Type Structure

Hierarchical type information, which implies different roles an entity may play in different scenarios, is of great significance for representation learning in knowledge graphs. Most typical knowledge graphs (e.g. Freebase and DBpedia) possess their own entity type information or could collect it from large encyclopedias like Wikipedia through entity alignment. These types are usually constructed with hierarchical structures, in which different granularities of semantic concepts are considered as sub-types in different layers. Most entities have more than one hierarchical type. Fig. 1 shows a brief example of the hierarchical type structure.

Taking a hierarchical type c with k layers for instance, $c^{(i)}$ is the i-th sub-type of c. We consider the most precise sub-type to be the first layer and the most general sub-type to be the last layer, while each sub-type $c^{(i)}$ has only one parent sub-type $c^{(i+1)}$. Walking through the bottom-up path in hierarchical structure, we can get the representation of hierarchical type as $c = \{c^{(1)}, c^{(2)}, ..., c^{(k)}\}$.

3.2 Overall Architecture

Existing translation-based models perform well in knowledge graphs, but few of them make full use of the rich information located in entity types. TransE represents entities and relations in a low-dimensional vector space and interprets relations as translating operations between entities. However, TransE has issues when modeling N-to-1, 1-to-N and N-to-N relations, since each entity has only one representation in every scenario. For instance, *William Shakespeare* has a variety of types (e.g. *book/author*, *award/award_nominee* and *music/artist*) and shows different attributes under different types. We believe that every entity in different scenarios, as the reflections of itself from various angles, should have different representations.

To implement the multiple representations of entities in different types, we propose the TKRL model. More specifically, we set type-specific projection matrices M_c constructed from the hierarchical structures for each type c, and then represent both h and t under the projections of the specific types c_{rh} and c_{rt} which head and tail should belong to in this relation. The energy function is defined as follows:

$$E(h,r,t) = ||\mathbf{M}_{rh}\mathbf{h}+\mathbf{r}-\mathbf{M}_{rt}\mathbf{t}||, \quad (2)$$

in which $\mathbf{M}_{rh}$ and $\mathbf{M}_{rt}$ are different projection matrices for h and t. We propose two hierarchical type encoders to construct those projection matrices.

3.3 Hierarchical Type Encoders

To encode the hierarchical type information into representation learning, we first propose a general form of type encoder to construct projection matrices for each entity. Secondly, two advanced encoders are proposed to take advantages of the internal connections in hierarchical structures and the prior knowledge in relation-specific type information.

General Form of Type Encoder

Most entities in KGs have more than one type, which could be utilized as supplementary information when representing entities. We propose a general form of type encoder, in which the projection matrix $\mathbf{M}_e$ for entity e will be the weighted summation of all type matrices:

$$\mathbf{M}_e = \alpha_1 \mathbf{M}_{c_1} + \alpha_2 \mathbf{M}_{c_2} + \cdots + \alpha_n \mathbf{M}_{c_n}, \quad (3)$$

where n is the number of types entity e has, c_i is the i-th type e belongs to, $\mathbf{M}_{c_i}$ is the projection matrix of c_i, and α_i represents the corresponding weight for c_i. Those weights could be set according to the influence c_i has on e measured by some statistical features like type frequency. Through the general type encoder, all projection matrices for entity e will be the same in different scenarios.

However, entities should have different representations to emphasize attributes of more importance in different scenarios. Fortunately, the relation-specific type information in KGs, which provides the possible type(s) an entity may belong to in a specific relation, could help for multiple entity representations. To take advantage of this information, the projection matrix $\mathbf{M}_{rh}$ in a specific triple (h, r, t) will be:

$$\mathbf{M}_{rh} = \frac{\sum_{i=1}^{n} \alpha_i \mathbf{M}_{c_i}}{\sum_{i=1}^{n} \alpha_i}, \quad \alpha_i = \begin{cases} 1, & c_i \in C_{r_h} \\ 0, & c_i \notin C_{r_h} \end{cases} \quad (4)$$

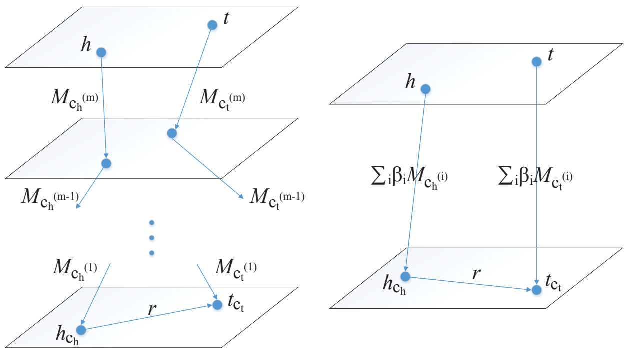
where C_{r_h} represents the type set of head in relation r given by the relation-specific type information. Projection matrices $\mathbf{M}_{rt}$ for entities in position of tail will be of the same form as those for entities in head. $\mathbf{M}_c$ is the projection matrix for type c, which could be constructed by the following two encoders.

Recursive Hierarchy Encoder

To further improve the representation of projection matrix $\mathbf{M}_c$ by mining the latent information located in hierarchical type structures, we propose the Recursive Hierarchy Encoder (RHE). Inspired by [Hu *et al.*, 2015], each sub-type (i.e. layer) in hierarchy is represented as a projection matrix with different granularities. During the projection process, entities (e.g. *William Shakespeare*) will be first mapped to the more general sub-type space (e.g. *book*) and then be sequentially mapped to the more precise sub-type space (e.g. *book/author*). The matrix $\mathbf{M}_c$ is designed as follows:

$$\mathbf{M}_c = \prod_{i=1}^{m} \mathbf{M}_{c^{(i)}} = \mathbf{M}_{c^{(1)}} \mathbf{M}_{c^{(2)}} \ldots \mathbf{M}_{c^{(m)}}, \quad (5)$$

in which m is the number of layers for type c in the hierarchical structure, while $\mathbf{M}_{c^{(i)}}$ represents the projection matrix of the i-th sub-type $c^{(i)}$.



(a) RHE (b) WHE

Figure 2: Hierarchical Type Encoders

Weighted Hierarchy Encoder

RHE proposes a recursive method of building hierarchical type projection matrices. However, different granularities of sub-types in hierarchical structures may vary in significance when mapping entities. In this case, we propose the Weighted Hierarchy Encoder (WHE) to consider weights in hierarchy. In WHE, sub-types are represented as projection matrices too. However, instead of using recursive operation to encode different granularities of sub-types, we sum up those projection matrices with different weights to represent the hierarchical type matrix $\mathbf{M}_c$ as follows:

$$\mathbf{M}_c = \sum_{i=1}^{m} \beta_i \mathbf{M}_{c^{(i)}} = \beta_1 \mathbf{M}_{c^{(1)}} + \cdots + \beta_m \mathbf{M}_{c^{(m)}}, \quad (6)$$

in which m is the number of layers in the hierarchical structure, $\mathbf{M}_{c^{(i)}}$ is the projection matrix of $c^{(i)}$, while β_i is the corresponding weight of $c^{(i)}$. We design a proportional-declined weighting strategy between $c^{(i)}$ and $c^{(i+1)}$ as follows:

$$\beta_i : \beta_{i+1} = (1 - \eta) : \eta, \quad \sum_{i=1}^{m} \beta_i = 1, \quad (7)$$

in which we set $\eta \in (0, 0.5)$. The strategy indicates that the more precise sub-type $c^{(i)}$ is, the higher weight β_i will be, thus the greater influence $c^{(i)}$ will have on $\mathbf{M}_c$.

3.4 Objective Formalization

We formalize a margin-based score function with negative sampling as objective for training:

$$L = \sum_{(h,r,t) \in T} \sum_{(h',r',t') \in T'} \max(\gamma + E(h, r, t) - E(h', r', t'), 0), \quad (8)$$

where $E(h, r, t)$ is the energy function score of positive triple and $E(h', r', t')$ is that of negative triple. $\gamma > 0$ is a hyper-parameter of margin. T' stands for the negative sampling set of T. Since there are no explicit negative triples in knowledge graphs, T' is constructed as follows:

$$\begin{aligned} T' = &\{(h', r, t) | h' \in E\} \cup \{(h, r, t') | t' \in E\} \\ &\cup \{(h, r', t) | r' \in R\}, \quad (h, r, t) \in T, \end{aligned} \quad (9)$$

in which the head or tail in a positive triple is randomly replaced by any other entity in E. Differed from [Bordes *et al.*, 2013; Lin *et al.*, 2015b], we also add relation replacements to negative sampling for better performances in relation prediction. Moreover, the new triples after replacements will not be considered as negative samples if they are already in T.

3.5 Type Information as Constraints

We use hierarchical type encoders to build type-specific projection matrices mapping entities to different semantic vector spaces in different scenarios. Besides we can use type information not only as projection matrices but also as constraints with the help of relation-specific type information. We propose two methods for using type information as constraints in both training and evaluation.

Type Constraints in Training

In training, we select negative entity samples from all entities in E with equal probability. In this case, entities sharing the same types tend to cluster and have similar representations, which actually becomes the main cause of errors in entity prediction. To solve this problem, we propose a method for negative sampling named Soft Type Constraint (STC). STC improves the probability of selecting entities which have the same types constrained by relation-specific type information, and thus broadens the distances between similar entities. Instead of setting hard constraints as in [Krompaß *et al.*, 2015] which may disturb the clustering of similar entities, we utilize soft constraints in which the probability of entities with the same types being selected is as follows:

$$P(e' \in E_c) = \frac{(k+1)|E_c|}{|E| + k|E_c|}, \quad k \in \mathbb{N}, \tag{10}$$

where c is the golden type for e in this triple, $E_c \in E$ is the entity set in which all entities have type c, with $|E_c|$ to be the number of entities in E_c. k is a hyper-parameter that a smaller k means a softer type constraint. It indicates that the probability of selecting entities in E_c is k times bigger than the probability of selecting entities which are not. With STC in negative sampling, we can balance the diversity as well as similarity between entities with the same types to get better performances.

Type Constraints in Evaluation

Besides type constraints in training, we can also utilize the information as type constraints in evaluation (TCE). It is intuitive that the head and tail entities should follow the type constraints provided by relation-specific type information. We can simply remove the candidates which don't follow the type constraints in evaluation. However, the performances of TCE are mostly based on the correctness and completeness of relation-specific type information, since type constraints could be incomplete or even wrong, which may lead to misses in prediction. In evaluation, we will show the power of TCE in auxiliary experiments.

3.6 Optimization and Implementation Details

The TKRL model can be stated as a parameter set $\theta = (\mathbf{E}, \mathbf{R}, \mathbf{M})$, in which $\mathbf{E}$ and $\mathbf{R}$ stand for the embeddings of entities and relations, and $\mathbf{M}$ stands for the projection matrices of all sub-types.

Optimization and Model Initialization

The TKRL model is optimized with mini-batch stochastic gradient descent (SGD), while chain rule is applied to update all parameters. The sub-type projection matrix set $\mathbf{M}$ could be initialized randomly or by identity matrix. Both $\mathbf{E}$ and $\mathbf{R}$ could be either initialized randomly or be pre-trained by existing translation-based models like TransE.

Implementation Details

We look through KGs to collect type instances for entities and relation-specific type information. To decrease the influence of incompleteness located in KGs, we also follow the local closed-world assumptions (LCWA) proposed in [Krompaß *et al.*, 2015] as a supplementary method. In LCWA, all entities appear in head (or tail) with the same relation should be allocated with the same type. Following [Wang *et al.*, 2014b], we implement two strategies (i.e. "unif" and "bern") for replacing head or tail with equal or different probabilities when selecting negative samples. For the consideration of efficiency, we employ a multi-thread version for learning.

4 Experiments

4.1 Datasets and Experiment Settings

Datasets

In this paper, we first use a typical knowledge graph FB15K [Bordes *et al.*, 2013] to evaluate our models on knowledge graph completion and triple classification. FB15K is a dataset extracted from Freebase [Bollacker *et al.*, 2008] which contains 14,951 entities, 1,345 relations and 592,213 triples in total. Following [Bordes *et al.*, 2013], we split those triples into train, validation and test set.

As for the type information, we collect all type instances for entities in FB15K located in *type/instance* field, as well as the relation-specific type information located in *rdf-schema#domain* and *rdf-schema#range* fields. We also filter the type instances which never appear in extracted relation-specific type information, since those types have little influence on relations in FB15K. After filtering, we have 571 types. All entities in FB15K have at least one hierarchical type and the average number of types is approximately 8.

Entities and relations in FB15K are limited that they should have at least 100 mentions in Freebase [Bordes *et al.*, 2013]. However, relations and entities in real-world KGs are much more sparse than FB15K due to the long tail. To further present the advantages of our models in real-world distribution, we construct FB15K+, which contains the same entities in FB15K and almost all relations between those entities. We only discard relations which just appear once, for they can't exist both in train and test set. The new-added 6,607 triples are split into train and test set where every relation is mentioned at least once in train set. FB15K+ have 806 types after filtering. The statistics of datasets are listed in Table 1.

Experiment Settings

In evaluation, we implement TransE and TransR for comparison. For TransE, we improve their dissimilarity measure with L_1-norm and replace relations as well as entities during negative sampling. We also use "bern" to replace head or tail with different probability following [Wang

Table 1: Statistics of datasets

Dataset	#Rel	#Ent	#Train	#Valid	#Test
FB15K	1,345	14,951	483,142	50,000	59,071
FB15K+	1,855	14,951	486,446	50,000	62,374

et al., 2014b]. For TransR, we directly use the released code given in [Lin *et al.*, 2015b] and utilize replacements of relations in negative sampling for better performances in relation prediction. Both TransE and TransR are trained with the best parameters reported in their papers. For other baselines including RESCAL [Nickel *et al.*, 2011; 2012], SE [Bordes *et al.*, 2011], SME [Bordes *et al.*, 2012; 2014] and LFM [Jenatton *et al.*, 2012], we directly use the results reported in [Lin *et al.*, 2015b].

We train TKRL model with mini-batch SGD. As for parameters, we select the batch size B among {20, 240, 1200, 4800}, and margin γ among {0.5, 1.0, 1.5, 2.0}. We also set the dimensions of entity and relation to be the same n, and all projection matrices are set to be $n \times n$. For learning rate λ, we could select a fixed rate following [Bordes *et al.*, 2013; Lin *et al.*, 2015b] or design a flexible learning rate which will descend through iteration. For WHE, we select the descending weight η between sub-types among $\{0.1, 0.15, 0.2\}$. For k in type constraints, we select among $\{5, 10, 15\}$. The optimal configurations of our models are: $B = 4800$, $\gamma = 1.0$, $\eta = 0.1$, $k = 10$, with λ designed by a linear-declined strategy which ranges from 0.0025 to 0.0001. TKRL and TransR are trained with entities and relations initialized by pre-trained TransE (unif) model. For a fair comparison, all models are trained under the same dimension $n = 50$.

4.2 Knowledge Graph Completion

Evaluation Protocal

Knowledge graph completion aims to complete a triple (h, r, t) when one of h, r, t is missing, which is used in [Bordes *et al.*, 2011; 2012; 2013]. Two measures are considered as our evaluation metrics: (1) mean rank of correct entities or relations; (2) proportion of correct answers ranked in top 10 (for entities) or top 1 (for relations). We also follow the two evaluation settings named "raw" and "filter".

We conduct our evaluation on FB15K and divide the task into two sub-tasks: entity prediction and relation prediction. For a fair comparison, evaluation conditions are the same for all models. We also evaluate on the method of type constraint in evaluation (TCE) and a new dataset with long-tail distribution as auxiliary experiments.

Entity Prediction

The results of entity prediction are shown in Table 2. From the results we observe that: (1) Both RHE and WHE significantly outperform all baselines in mean rank and Hits@10. It indicates that the hierarchical type information, which is successfully encoded into entity and relation embeddings, could improve the representation learning of knowledge graphs. (2) WHE+STC achieves the best performance with approximately 6.2% improvement compared to TransR in Hits@10, and such improvement provided by Soft Type Constraint (STC) can also be found in RHE. It is because that STC increases the probability of entities with the same types as the golden one being selected during negative sampling, which widens the distances between entities sharing the same types, and thus lowers the errors caused by those similar entities. However, STC has side effects that it may result in higher mean rank, since some wrong-predicted instances with extremely high rank will significantly increase mean rank. (3) Type information, either in form of projection matrices or type constraints, could provide significant supplements for RL in KGs.

Table 2: Evaluation results on entity prediction

Metric	Mean Rank		Hits@10(%)	
	Raw	Filter	Raw	Filter
RESCAL	828	683	28.4	44.1
SE	273	162	28.8	39.8
SME (linear)	274	154	30.7	40.8
SME (bilinear)	284	158	31.3	41.3
LFM	283	164	26.0	33.1
TransE	238	143	46.4	62.1
TransR	199	77	47.2	67.2
TKRL (RHE)	**184**	**68**	49.2	69.4
TKRL (WHE)	186	**68**	49.2	69.6
TKRL (RHE+STC)	202	89	**50.4**	73.1
TKRL (WHE+STC)	202	87	50.3	**73.4**

Relation Prediction

The results of relation prediction are shown in Table 3. We implement two typical models including TransE and TransR as baselines. From Table 3 we observe that: (1) Both RHE and WHE significantly outperform TransE and TransR in mean rank and Hits@10, and RHE achieves the best performance. It proves that RHE is better in relation prediction while WHE is better in entity prediction. (2) STC lowers the performances on relation prediction since wider distances between entities with the same types may confuse the entity clustering. In spite of this, all models with STC still outperform TransE, which indicates the positive effects of type information as constraints.

Table 3: Evaluation results on relation prediction

Metric	Mean Rank		Hits@1(%)	
	Raw	Filter	Raw	Filter
TransE	2.79	2.43	68.4	87.2
TransR	2.49	2.09	70.2	91.6
TKRL (RHE)	**2.12**	**1.73**	**71.1**	**92.8**
TKRL (WHE)	2.22	1.83	70.8	92.5
TKRL (RHE+STC)	2.38	1.97	68.7	90.7
TKRL (WHE+STC)	2.47	2.07	68.3	90.6

Type Constraints in Evaluation

Type constraints in training have been proved to be effective, while type constraints in evaluation (TCE) could be utilized to achieve even better performances in entity prediction, on

Table 4: Evaluation results on long-tail distribution

Relation Frequency	Test Number	Hits@10 for Entity (%)			Hits@1 for Relation (%)		
		TransE	TransR	TKRL (WHE)	TransE	TransR	TKRL (WHE)
$f_r <= 10$	1,444	28.0	32.4 (+4.4)	38.1 (**+10.1**)	13.2	17.0 (+3.8)	21.5 (**+8.3**)
$f_r <= 100$	4,763	49.9	54.5 (+4.6)	57.9 (**+8.0**)	45.7	50.5 (+4.8)	54.3 (**+8.6**)
$f_r <= 1000$	18,296	66.1	69.1 (+3.0)	71.6 (**+5.5**)	70.9	75.4 (+4.5)	77.8 (**+6.9**)
$total$	62,374	61.9	67.2 (+5.3)	69.2 (**+7.3**)	80.4	88.8 (+8.4)	89.7 (**+9.3**)

condition that the relation-specific type information is relatively complete. For a fair comparison, we implement baselines with the helps of both STC and TCE. Results in Table 5 show that: (1) All models have better performances with TCE compared to those corresponding results without TCE shown in Table 2, and the improvements will be more significant when combined with STC. It is because that TCE removes the candidates which don't follow the type constraints, while STC sharpens the differences between similar entities. (2) TKRL models outperform all baselines even when compared with the enhanced versions with STC, which implies the significance of hierarchical type encoders.

Table 5: Evaluation results on entity prediction with TCE

Metric	Mean Rank		Hits@10(%)	
	Raw	Filter	Raw	Filter
TransE+TCE	212	116	46.9	63.4
TransR+TCE	182	60	47.9	68.6
TransE+STC+TCE	203	104	49.8	69.9
TransR+STC+TCE	185	63	48.5	71.7
TKRL (RHE+STC+TCE)	**169**	56	**51.4**	75.4
TKRL (WHE+STC+TCE)	170	**55**	51.3	**75.6**

Knowledge Graph Completion with Long Tail

Representation learning of real-world KGs suffers from the long-tail distribution. We construct FB15K+, which contains almost all relations between entities in FB15K as well as the corresponding triples, to simulate the distribution in real-world KGs. From Table 4 we can observe that: (1) WHE significantly and consistently outperforms TransE and TransR in all conditions even without STC. (2) WHE achieves 5.8% and 4.5% improvements on entity and relation prediction compared to TransR with $f_r <= 10$, while it achieves 2.0% and 0.9% improvements with all triples. It demonstrates that TKRL takes advantages over TransR especially with low-frequency relations, and thus is more robust when modeling KGs with real-world distribution.

4.3 Triple Classification

Triple classification aims to confirm whether a triple (h, r, t) is correct or not. This binary classification task has been explored in [Socher *et al.*, 2013; Wang *et al.*, 2014b; Lin *et al.*, 2015b] for evaluation.

Evaluation Protocal

We evaluate this task on FB15K. Since FB15K has no explicit negative triples, we construct the negative triples following the same protocol used in [Socher *et al.*, 2013]. The classification strategy is conducted as follows: We set different relation-specific thresholds δ_r for each relation. For a triple (h, r, t), if the dissimilarity score of $E(h, r, t)$ is below δ_r, the triple is then predicted to be positive and otherwise negative. The relation-specific thresholds δ_r are optimized by maximizing the classification accuracies in all triples with the corresponding r on the validation set.

Results

Evaluation results on triple classification are shown in Table 6. From Table 6 we observe that: (1) TKRL models outperform all baselines, and WHE+STC achieves the best performance, which confirms the advantages TKRL has over baselines in triple classification. (2) STC improves the performances of both RHE and WHE, which indicates that sharpening the dissimilarity between entities with the same types is significantly helpful for triple classification.

Table 6: Evaluation results on triple classification

Methods	Accuracy(%)
TransE	85.7
TransR	86.4
TKRL (RHE)	86.9
TKRL (WHE)	87.1
TKRL (RHE+STC)	88.4
TKRL (WHE+STC)	**88.5**

5 Conclusion and Future Work

In this paper, we propose TKRL model for representation learning of knowledge graphs with hierarchical types. We consider type information as projection matrices for entities, which are constructed with two hierarchical type encoders. Moreover, type information is also regarded as constraints in training and evaluation. In experiments, we evaluate our models on two tasks including knowledge graph completion and triple classification. Experimental results show that type information is significant in both tasks especially with long-tail distribution, and TKRL model is capable of encoding hierarchical type information into KG representations.

We will explore the following research directions in future: (1) TKRL model only considers type information into representation learning of KGs, while there is rich information in the form of images and texts which could be integrated to our model. We will explore the advantages of those rich information in future. (2) More hierarchical type structures such as Wikipedia categories could be introduced to bring in deeper hierarchical information, while the hierarchical type encoders could be further improved with more sophisticated algorithms designed for hierarchical structures.

Acknowledgments

This work is supported by the 973 Program (No. 2014CB340501), the National Natural Science Foundation of China (NSFC No. 61133012 and 61572273) and the Tsinghua University Initiative Scientific Research Program (20151080406).

References

[Bollacker *et al.*, 2008] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of KDD*, pages 1247–1250, 2008.

[Bordes *et al.*, 2011] Antoine Bordes, Jason Weston, Ronan Collobert, and Yoshua Bengio. Learning structured embeddings of knowledge bases. In *Proceedings of AAAI*, pages 301–306, 2011.

[Bordes *et al.*, 2012] Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. Joint learning of words and meaning representations for open-text semantic parsing. In *Proceedings of AISTATS*, pages 127–135, 2012.

[Bordes *et al.*, 2013] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Proceedings of NIPS*, pages 2787–2795, 2013.

[Bordes *et al.*, 2014] Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. A semantic matching energy function for learning with multi-relational data. *Machine Learning*, 94(2):233–259, 2014.

[Dong *et al.*, 2014] Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *Proceedings of KDD*, pages 601–610, 2014.

[Gu *et al.*, 2015] Kelvin Gu, John Miller, and Percy Liang. Traversing knowledge graphs in vector space. In *Proceedings of EMNLP*, pages 318–327, 2015.

[Hu *et al.*, 2015] Zhiting Hu, Poyao Huang, Yuntian Deng, Yingkai Gao, and Eric P Xing. Entity hierarchy embedding. In *Proceedings of ACL*, volume 1, pages 1292–1300, 2015.

[Jenatton *et al.*, 2012] Rodolphe Jenatton, Nicolas L Roux, Antoine Bordes, and Guillaume R Obozinski. A latent factor model for highly multi-relational data. In *Proceedings of NIPS*, pages 3167–3175, 2012.

[Ji *et al.*, 2015] Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. Knowledge graph embedding via dynamic mapping matrix. In *Proceedings of ACL*, pages 687–696, 2015.

[Krompaß *et al.*, 2015] Denis Krompaß, Stephan Baier, and Volker Tresp. Type-constrained representation learning in knowledge graphs. In *Proceedings of ISWC*, pages 640–655. 2015.

[Lin *et al.*, 2015a] Yankai Lin, Zhiyuan Liu, Huanbo Luan, Maosong Sun, Siwei Rao, and Song Liu. Modeling relation paths for representation learning of knowledge bases. In *Proceedings of EMNLP*, pages 705–714, 2015.

[Lin *et al.*, 2015b] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of AAAI*, 2015.

[Neelakantan *et al.*, 2015] Arvind Neelakantan, Benjamin Roth, and Andrew McCallum. Compositional vector space models for knowledge base completion. *Proceedings of EMNLP*, 2015.

[Nickel *et al.*, 2011] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A three-way model for collective learning on multi-relational data. In *Proceedings of ICML*, pages 809–816, 2011.

[Nickel *et al.*, 2012] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. Factorizing yago: scalable machine learning for linked data. In *Proceedings of WWW*, pages 271–280, 2012.

[Socher *et al.*, 2013] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. Reasoning with neural tensor networks for knowledge base completion. In *Proceedings of NIPS*, pages 926–934, 2013.

[Wang *et al.*, 2014a] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph and text jointly embedding. In *Proceedings of EMNLP*, pages 1591–1601, 2014.

[Wang *et al.*, 2014b] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of AAAI*, pages 1112–1119, 2014.

[Xie *et al.*, 2016] Ruobing Xie, Zhiyuan Liu, Jia Jia, Huanbo Luan, and Maosong Sun. Representation learning of knowledge graphs with entity descriptions. In *Proceedings of AAAI*, 2016.

[Yang *et al.*, 2014] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. *Proceedings of ICLR*, 2014.

[Zhong *et al.*, 2015] Huaping Zhong, Jianwen Zhang, Zhen Wang, Hai Wan, and Zheng Chen. Aligning knowledge and text embeddings by entity descriptions. In *Proceedings of EMNLP*, pages 267–272, 2015.